Journal of Networks

ISSN 1796-2056
Volume 5, Number 1, January 2010

Contents

Special Issue: Recent Advances in Network and Parallel Computing
Guest Editors: Jian Cao and Xin Wang

Guest Editorial
Jian Cao and Xin Wang

SPECIAL ISSUE PAPERS

Continuance Parallel Computation Grid Composed of Multi-Clusters
Qingkui Chen, Haifeng Wang, and Wei Wang

FAPP: A New Fairness Algorithm for Priority Process Mutual Exclusion in Distributed Systems
Sukendu Kanrar and Nabendu Chaki

LBLS: A Locality Bounded Hashing-Based Location Service
Ruonan Rao, Shuyin Liang, and Jinyuan You

LDB: Localization with Directional Beacons for Sparse 3D Underwater Acoustic Sensor Networks
Hanjiang Luo, Zhongwen Guo, Wei Dong, Feng Hong, and Yiyang Zhao

Map Synchronization and Alternatives Optimization for Firefighters Cooperative Decision Support in Ad Hoc Networks
Yanwei Chen, Demin Li, Chenwen Wang, and Jiacun Wang

Relay Aided Wireless Multicast Utilizing Network Coding: Outage Behaviour and Diversity Gain
Chen Zhi, Chen Wei, Pingyi Fan, and Khaled Ben Letaief

An Incentive Mechanism for Tree-based Live Media Streaming Service
Shuang Yang and Xin Wang

A Programmable Architecture for Layered Multimedia Streams in IPv6 Networks
Brendan McAllister, Alan J. Marshall, and Roger F. Woods

Novel Stream Cipher Using SCAN and Variable Ordered Recursive CA Substitutions and Its DSP+FPGA Implementation
Rong-Jian Chen, Jui-Lin Lai, and Shi-Jinn Horng

A Novel Data Mining-Based Method for Alert Reduction and Analysis
Fu Xiao, Shi Jin, and Xie Li

REGULAR PAPERS

Measuring the Botnet using the Second Character of Bots
Zhitang Li, Jun Hu, Zhengbing Hu, Bingbing Wang, Liang Tang, and Xin Yi
Control Strategy of Scheduling Invalidation of Data Broadcasting
Wen-bin Hu, Zhengbing Hu, Fu-qiang Yang, Cun-lian Fan, Lei Wang, and Xin-long Tan

A Novel Iterative Multilateral Localization Algorithm for Wireless Sensor Networks
Shaoping Zhang, Guohui Li, Wei Wei, and Bing Yang

Analysis of Queuing Behaviors with Self-similar Traffic in Wireless Channels
Chang Liu, Xiaowei Qin, Sihai Zhang, and Wuyang Zhou
Special Issue on Recent Advances in Network and Parallel Computing

Guest Editorial

This special issue is associated with the International Conference on Network and Parallel Computing (NPC 2008), which was held in Shanghai, China in October 2008. The purpose of this special issue is to provide a channel to guarantee fast publication of extended versions of the high-quality conference papers on network and parallel computing in the Journal.

Papers for this special issue were solicited not only from participants presented at the NPC 2008 and Workshops, but also from authors with original high-quality contributions that have neither been published in nor submitted to any journals or refereed conferences. Papers were sought that encompassed interdisciplinary research in the sense of applying novel network and distributed computing techniques to unsolved problems, such as wireless, P2P, high-performance computing, networking coding and etc.

We received 21 papers from around the world and selected 10 to be included in the special issue after a thorough and rigorous review process. The presented papers span a number of topics and are assembled into 4 categories, i.e. parallel computing, wireless networks, steam media and security.

In recent years, many new types of parallel computing such as grid computing and cluster computing become the hot research topics. In “Continuance Parallel Computation Grid Composed of Multi-Clusters”, Q. K Chen introduces a Continuance Parallel Computation Grid (CPCG) architecture in the dynamic network environment. It makes use of multi-agent structure, fuzzy theory based control, self-learning method, data parallel computing and migration mechanism to design the CPCG model that can support the continuance data parallel computing. In “FAPP: A New Fairness Algorithm for Priority Process Mutual Exclusion in Distributed Systems”, S Kanrar et.al proposes a new token based Fairness Algorithm for Priority Processes (FAPP) that addresses both the issues and keeps the control message traffic reasonably low.

Wireless network has been widely applied in many domains. In “LDBLS: A Locality Bounded Hashing-Based Location Service”, R.N. Rao presents a new location service, named LDBLS (Locality Bounded Location Service) to solve the locality problem with the comparable least communication and storage cost. In “LDB: Localization with Directional Beacons for Sparse 3D Underwater Acoustic Sensor Networks”, H. J. Luo proposes a 3D localization scheme with directional beacons for Underwater Acoustic Sensor Networks (UWA-SNs). By utilizing an Autonomous Underwater Vehicle (AUV) as a mobile beacon sender, the AUV patrols over the 3D deployment volume with predefined trajectory. Theoretical analysis and simulation results show high localization accuracy. There are still many research issues to be solved in wireless network itself. In “Map Synchronization and Alternatives Optimization for Firefighters Cooperative Decision Support in Ad Hoc Networks”, Y.W. Chen et al presents an effective map synchronization scheme and a method to solve the problem of assistance alternative optimization of firefighter cooperation decision support in mobile ad-hoc networks. The proposed process will increase system effectiveness safety. In “Relay Aided Wireless Multicast Utilizing Network Coding: Outage Behaviour and Diversity Gain”, C. Zhi et al define cooperative multicast schemes to achieve higher spatial diversity. They developed a network coding based cooperative multicast scheme (NCBC), which exploits limited feedback. Simulation results demonstrated significant gains over the direct multicast transmission.

Steam media becomes a very important application of network. We have two papers focusing on this topic. In “An Incentive Mechanism for Tree-based Live Media Streaming Service”, S. Yang et al addresses the incentive problem in P2P live streaming, which very important when there are free-riders. They proposed a rotation-based incentive mechanism for overlay tree structure. The rotation plus reenter method is simple and effective. In “A Programmable Architecture for Layered Multimedia Streams in IPv6 Networks”, B. McAllister et al introduces programmable network functionality (PNF) into QoS-enabled IPv6 networks. They integrate FPGA-based hardware devices into intermediate network devices. Thus improves efficiency and flexibility in multimedia transmission.

Security is always an unavoidable issue in network and its application. In “Novel Stream Cipher Using SCAN and Variable Ordered Recursive CA Substitutions and Its DSP+FPGA Implementation”, R. J. Chen et al presents a new method of stream cipher using 2-D hybrid cellular automata (CA). The scanning pattern for the CA is generated using an image pre-processing language (SCAN) from a short set of primitives. The hardware implementation of DSP plus FPGA is feasible and helpful for image security. In “A Novel Data Mining-Based Method for Alert Reduction and Analysis”, X. Fu et al gives a novel data mining-based method for alert reduction and analysis. They build a real-time framework to filter false IDS alerts using outlier detection. The prototype enhances the efficiency for handling IDS alerts.

We believe that this Special Issue will contribute to enhancing knowledge in many diverse areas of the network and parallel computing. In addition, we also hope that the presented results will stimulate further research in these areas.

The editors wish to acknowledge the great efforts of many people who helped with this Special Issue. We thank the following reviewers for their tireless efforts in completing the reviews in a short period of time without compromising...
quality:


We also thank Mrs. George J. Sun, the Executive Editor-in-Chief of JNW for his continued encouragement, guidance and support in the preparation of this issue.

Guest Editors

Jian Cao
Department of computer science & technology, Shanghai Jiaotong University, Shanghai, China

Xin Wang
School of Computer Science, Fudan University, Shanghai, China

Jian Cao got his Ph.D degree from Nanjing University of Science & Technology in 2000. He is a professor of the department of computer science & technology, Shanghai Jiaotong University. His main research topics include network computing, cooperative information system and software engineering. He has published more than fifty papers in referred journals and conferences.

Xin Wang is currently an Associate Professor in the School of Computer Science at Fudan University. He received his B.S. degree in information theory and M.S. degree in communication and electronic systems from Xidian University, China, in 1994 and 1997, respectively. He received his Ph.D. degree in computer science from Shizuoka University, Japan, in 2002. His primary research interests include wireless networks, peer-to-peer networks, and network coding.
Continuance Parallel Computation Grid Composed of Multi-Clusters

Qingkui Chen
School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, China
Email:chenqingkui@tom.com

Haifeng Wang and Wei Wang
Business school, University of Shanghai for Science and Technology, Shanghai China
Email: gadfly7@126.com

Abstract—For supporting the grid computing in dynamic network environment composed of multi-clusters, a continuance parallel computation grid (CPCG) is proposed in this paper. A series of formal definitions, such as the CPCG architecture, the dynamic network environment (DNE), the management agent system, the independent computing agents (ICA) which support the traditional computing (TC), the cooperation computing team (CCT) which supports the data parallel computing (DPC), and their relations are given. Through DPC, TC, and the migration mechanism, the continuance data parallel computing (CDPC) was constructed. The dynamic learning method, the fuzzy partition technique for the logical computer cluster on which CCT runs, the stage checkpoint mechanism and the migration process are studied. CPCG computing process is described. The experiment results show that CPCG resolves effectively the problems of optimization use of resources in DNE. It can be fit for grid computing.

Index Terms—parallel computation, dynamic network, multi-clusters grid, continuance parallel computation

I. INTRODUCTION

With the rapid development of the information techniques and their popular applications, the demand for the high performance processing devices is becoming more and more vehement. Nowadays, the numbers of Intranet composed of many computer clusters are quickly increasing, and a great deal of cheap personal computers are distributed everywhere, but the using rate of their resources is very low[1, 2]. Mining and adopting these idle resources, we can get alot of large-scale high performance computation, storage and communication resources which are not special. However, the heterogeneous, dynamic and unstable characteristic of these resources brings the huge obstacle for us. The grid [3] techniques and multiagents [4, 5] can become the main approaches to use effectively these resources, and the multiagents have already become the feasible solutions for grid computing. There are many successful examples [6, 7] of applications which are in conjunction with the automatic multiagent system. On the other hand, the researches of parallel computing based on grid are development quickly, and there are a lot of research results [8-12] today. The data parallel computing (DPC)[13] is an important computation mode in traditional parallel environments, such as the computer cluster. Because the dynamic network environment (DNE), which is composed of many computer-clusters connected by Intranet, is dynamical, the DPC computing can’t be completed in one computing phase, and it need many computing phases to complete. Due to the computation resources for DPC may be single computer or a computer cluster, DPC mode have to degenerate into the traditional serial computing (TC) mode. Therefore, the continuance data parallel computing (CDPC) in DNE means that the computing task is executed mainly by the DPC mode and sometimes by TC mode and it need many computing phases to complete. How to use the computation resources in DNE to support the CDPC is a very interest work. The key techniques are as follows : (1)How to partition the logical computational unit for DPC; (2) How to classify the DPC task by their resource demands; (3) How to set the checkpoint [10] [12] [14] [15] [16]; (4) How to increase the intelligence of computing agents by self-learning[17,18];(5)How to migrate the computational tasks [11] [19-22].

This paper introduced a Continuance Parallel Computation Grid (CPCG) in DNE. Building an open grid computing environment; using of the idle computational resources of DNE; adopting the multiagent, the fuzzy theory [23], the self-learning methods, DPC, TC and the migration mechanism, we designed the CPCG model that can support the continuance data parallel computing. The experimental results show that CPCG can increase the percentage of the using resources in DNE and it can improve response time of computation-intensive tasks.

II ARCHITECTURE OF MCG

CPCG includes two parts: one is DNE that is the personal computers for grid computing, and DNE is composed of many computer clusters connected through LAN and Intranet, and all the computers is not special; other is the agent system, and we call it as GCG (Global Computing Group). GCG is composed of a management...
agent system (MAS) and a lot of computing agents. The architecture is presented in figure 1.

A. DNE

Definition.1. Computing node (CN) is defined as CN (id, CT, AS, RSV, st), where id is the identifier of CN; Let CT be the type of computing node (definition 10); Let AS be set of agents running on CN; RSV(it denotes resource support vector, its power of its CPU denotes as \( r_{cpu} \), power of its memory storage denotes as \( r_{mem} \), power of its hard-disk denotes as \( r_{disk} \), and power of its network adapter denotes as \( r_{net} \) ). Let st \( \in \{ \) “Local”, “Idle”, “Running” \} be its state. “Local” represents that computer is undertaking local tasks, “Idle” means computer of being vacant. “Running” denotes that computer is performing tasks for Grid.

Definition.2. Computer cluster (CC) is defined as CC (Master, CS), where Master is the main computer of CC; Let CS = \{CN_1, CN_2, ..., CN_n\} be the set of all computing nodes in computer cluster.

Definition.3. Logical computer cluster (LCC) is defined as LCC (id, LCS, B, CC), where id denotes the identifier of LCC; Let LCS be the set of computing nodes of LCC; CC is the computer cluster which comprises LCC. Network bandwidth of LCC denotes as B.

So, the dynamic network environment (DNE) can be defined as DNE (Master, CCS, N, R), where Master denotes the main computer of DNE; CCS is the set of all computer clusters in DNE; Let N be its network set; R denotes the connection rules.

B. Management Agents System (MAS)

The main functions of MAS are the computation resource management, the DNE monitoring and the task scheduler. MAS involves three parts. The first part is the global control agent (GA) which should be responsible for management of CPCG. The second part, called as local agent (LA), is the agent for managing single computer cluster. The last part is the agents for managing the computing nodes, which are called as the node Agents (NA), and each computing node has a NA. MAS’ structure is the tree-level-ring-tree (TLRT) and it is shown in figure 1.

The main functions of GA are as follows: (1) Control and manage DNE; (2) Receive and dispatch the computing tasks; (3) control and monitor CDPC process; (4) Construct and remove CCT for DPC; (5) construct the LCC for DPC;

The main functions of LA are as follows: (1) Control all the computing nodes in its cluster; (2) Calculate the idle resources of cluster, and report them to GA; (3) Monitor the states all computing nodes and node agents;

The main functions of NA are as follows: (1) Control the computing node to join or to disjoin the DNE in dynamic; (2) Calculate the idle resources, and report them to LA; (3) Monitor the states and events in CN, and make the response adjustments; (4) Control the computing agents (ICA or CCA) to complete the computing task.

C. Computing Agents and GCG

There are mainly two types of computing mode in CPCG to support CDPC: the traditional serial computing (TC) based on single computer and data parallel computing (DPC) based on logical computer cluster. In order to support the two computing modes, the cooperation computing team (CCT) and the independent computing agent (ICA) must be introduced firstly.

Definition.4. Computing agent (CA) is defined as CA (id, PRG, BDI, KS, CE), where id is the identifier of CA; Let PRG be the executable program set of CA; BDI is the description of its BDI; Knowledge set denotes as KS; Configuration environment denotes as CE. CA is the basic element to execute computation task.

Definition.5. Independent computing agent (ICA). If one CA could complete the task independently, we call it as the independent computing agent (ICA).

Definition.6. Cooperative computing agent (CCA). If one CA couldn’t complete the task independently. So it must cooperate with others. Then we call it as the cooperative computing agent (CCA).

Definition.7. Cooperation computing team (CCT) is defined as CCT (id, Am, CAS, BDI, CKS, CCE, LCC), where id is the identifier of CCT; There exists main control agent denoted as Am which is the master of MASTER-SLAVER parallel mode in CCT.CAS is the set of all cooperative computing agents (CCA) in CCT.BDI is the description of its BDI; Knowledge set denotes as CKS. Configuration environment denotes as CCE. LCC is a logical computer cluster, on which CCT runs.

So, global computing group (GCG) is defined as GCG (id, MAS, ICAS, CCTS, GKS, GCE), where id is the identifier of GCG; MAS gives an indication of the management agent system of GCG; ICAS is indicative of the set of ICA which GCG includes; CCTS denotes as the set of CCT which GCG includes; GKS is an indication of knowledge set. GCE indicates its configuration environment. The relations between DNE and GCG are presented in figure 1.

D. CDPC

Definition.8. Task (TSK) is defined as TSK (id, RDV, DAT, AS, St), where id is the identifier of TSK; RDV(cpu, mem, disk, net) is its resource demand vector; DAT denotes as the data set; AS is an indication of the set of agents to calculate TSK. St represents its state. There exist six states of TSK which are “Committed”, “ready”, “Suspended”, “Migrating”, “Running” and “Completed”. The State graph is presented in Figure 2.
Obviously, there is no quiescent environment in DNE. The computing task, which will run for a long time, can not be completed in one computing phase. Then it must be migrated during the computing process. So, computing task may span many computing phases to be completed. Therefore, Beyond doubt, there exist different computing mode (ICA or CCT) in the whole period.

**Definition 9. Continuance Dependent Computing (CDPC)** is defined as a computing phase sequence \( \{ C_S_i \}, C_S_i \) is the ith computing phase, and the computing mode of \( C_S_i \) is ICA or CCT; If the computing mode is CCT, the CCT (LCC) scale may be different in different computing phases. All tasks of CPGC are to be executed in the form of CDPC. The continuance means that the suspended time of tasks is as short as possible.

### III CONSTRUCTION OF LOGICAL CLUSTER

When D4 receives the information from all computing nodes in DNE, the logical computer cluster will be constructed for DPC mode. Here let \( PCS=\{ CN_i \mid \{ id, CT_i, AS_i, RSV_i, st_i \} | 1 \leq i \leq n \} \) be the set of all idle computing nodes in DNE. Actually, as different computer node may provide different resources, we must classify the computer nodes to different power logical computer clusters according to their power. Therefore, the principle of classification is based on their power of CPU, Memory, Disk, and Net adapter. For solving this problem, we should firstly discrete the power data. And fuzzy relation theory is employed to represent the data.

#### A. Resource Matrix

Suppose that \( MR= (r_{ij}) \) (1 \( \leq i \leq n, 1 \leq j \leq 4 \)) is the resource matrix in DNE. \( F_j \) denotes as CPU frequency of \( CN_i \), here it is measured in MHZ, \( M_i \) denotes as memory capacity of \( CN_i \), and it is measured in MB; \( \bar{R}_i \) indicates Disk speed of \( CN_i \), and it is measured in RPM; \( N_i \) indicates communication power of \( CN_i \), and it is measured in MBS; The resource matrix elements are determined by the real ability of computing nodes, and the calculation method is as follows:

\[
egin{align*}
    r_{i1} & = \left\lfloor \frac{F_j}{500\text{MHz}} \right\rfloor + 1; \\
    r_{i2} & = \left\lfloor \frac{M_j}{128\text{MB}} \right\rfloor + 1; \\
    r_{i3} & = \left\lfloor \frac{\bar{R}_j}{5400\text{RPM}} \right\rfloor + 1; \\
    r_{i4} & = \left\lfloor \frac{N_j}{10000\text{MBS}} \right\rfloor + 1;
\end{align*}
\]

\( r_{i4} \) not be completed in one computing phase. Then it must be migrated during the computing process. So, computing task may span many computing phases to be completed. Therefore, Beyond doubt, there exist different computing mode (ICA or CCT) in the whole period.

**Partition for logical computer cluster**

It is very important to divide different logical computer clusters. Then the LCC division algorithm as follows:

**Algorithm 3.1 Partition method for LCC.**

1. \( MR= (r_{ij}) \) (1 \( \leq i \leq n, 1 \leq j \leq 4 \)) is the resource matrix in DNE, where \( n \) is the numbers of all idle computing nodes; \( T \) is a DPC mode task.
2. Construct the fuzzy matrix \( FM= \{ f_{ij} \} \) (1 \( \leq i \leq n, 1 \leq j \leq n \)), where
   \[
   f_{ij} = 1 - \beta (x_i r_{ij} - r_{ij} + x_i) r_{ij} - r_{ij} + x_j - r_{ij} + x_i = r_{ij} - \beta x_i r_{ij} > 0 \quad (1 \leq k \leq 4); 
   \]
3. (3) build the fuzzy equivalence matrix;
4. Repeat do
   
   \( FT=FM \odot FM; \) // \( \odot \) is the operation theorem to take the maximum and minimum
   
   If \( FT=FM \) then goto (4);
   
   \( FM=FT; \)
   
   End do;
5. (4) Calculate the c-cut matrix \( FM_4 \);)
6. (5) Divide the computing nodes of \( PCS \) into several equivalence class \( LCC_1 \cup LCC_2 \cup \ldots LCC_4 \) by \( FM_4 \);)
7. (6) Choose a LCC for \( T \) according to its resource demand vector by algorithm 3.2.

**B. Choose LCC for DPC mode task**

Suppose that \( LCC = LCC_1 \cup LCC_2 \cup \ldots LCC_4 \) is the set of all logical computer clusters which are built through algorithm 3.1, and \( T \) is a DPC mode task.

**Algorithm 3.2. Choose LCC method for DPC mode task**

1. (1) Get the resource demand vector \( RDV= (w_1, w_2, w_3, w_4) \) of \( T \);
   
   \( \min = \infty \);
2. (2) While \( LCC \neq \emptyset \) do
   
   Get \( S \in LCCS \); /* \( S \) is a logical computer cluster */
   
   Calculate total resource vector \( (ar_{cpu}, ar_{mem}, ar_{disk}, ar_{net}) \) of all computing nodes of \( S \), it is as follows:
   
   \[
   \begin{align*}
   & \{ ar_{cpu} = \sum_{CN \in S} CN . RSV . r_{cpu} \}; \\
   & \{ ar_{mem} = \sum_{CN \in S} CN . RSV . r_{mem} \}; \\
   & \{ ar_{disk} = \sum_{CN \in S} CN . RSV . r_{disk} \}; \\
   & \{ ar_{net} = \sum_{CN \in S} CN . RSV . r_{net} \}; \\
\end{align*}
\]

\( y_1, y_2, y_3, y_4 \) as short as possible.
Construct the vector \(Y(y_1, y_2, y_3, y_4)\); 
\[ e = [y_1 - y_i] + [y_2 - y_i] + [y_3 - y_i] + [y_4 - y_i]; \]
if \(e < c\) then \{ \(LCC = S; \) \(e = e\) \};
\[ LCCS = LCCS - \{S\}; \]
\} // end while
(3) Choose \(LCC\) as the logical computer cluster for \(T\); (4) End.

C. Optimized LCC

Obviously, the \(LCC\) may span many different networks in \(DNE\), so \(LCC\) can be optimized through the network skip distance, network bandwidth, and the resource demand vector of \(T\). Suppose that \(NSET = \{N_i \text{ (bw)} | 1 \leq i \leq m\}\) is the network set in \(LCC\). \(bw_i\) indicates bandwidth of \(N_i\), and \(m\) indicates the numbers of network. \(RD\) \((w_j, w_k, w_s, w_d)\) indicates the resource demand vector of task \(T\). The process is described as follows:

(1) Construct network matrix \(NM = (s_{ij}) (1 \leq i \leq m, 1 \leq j \leq m)\) that is composed of the factors of the network skip distance and bandwidth. The construction method for \(NM\) is as follows:
\[ s_{ij} = 1; \]
\[ s_{i,j} = 1 - \gamma \cdot \text{distance}(N_i, N_j) + |bw_i - bw_j|, \text{ when } i \neq j, \]
and \(0 < \gamma < 1;\)
When the network skip distance between \(N_i\) and \(N_j\) is 1, distance \((N_i, N_j) = 1;\)
When the network skip distance between \(N_i\) and \(N_j\) is 2, distance \((N_i, N_j) = 3;\)
When the network skip distance between \(N_i\) and \(N_j\) is 2, distance \((N_i, N_j) = 6;\)
When the bandwidth of \(N_i\) is 10MB, \(bw_i = 1;\)
When the bandwidth of \(N_i\) is 100MB, \(bw_i = 3;\)
When the bandwidth of \(N_i\) is 1000MB, \(bw_i = 6;\)
If \(N_i, bw_i = 1\) and \((N_j, bw_j = 3\) or \(N_j, bw_j = 6\) then distance \((N_i, N_j) = 6;\)
If \(N_i, bw_i = 3\) and \((N_j, bw_j = 1\) or \(N_j, bw_j = 6\) then distance \((N_i, N_j) = 6;\)
If \(N_i, bw_i = 6\) and \((N_j, bw_j = 1\) or \(N_j, bw_j = 3\) then distance \((N_i, N_j) = 6;\)
(2) Build the fuzzy equivalence matrix:
Repeat do
\[ FT = NM \odot NM; \] // \(\odot\) is the operation theorem to take the maximum and minimum
If \(FT = NM\) then goto (4); 
\[ NM = FT; \]
End do;
(3) calculate the \(c\)-cut matrix \(FM\) by \(w_e\) of resource demand vector of \(T\); 
(4) Divide the computing nodes of \(LCC\) into several equivalence class
\[ SLCCS = SLCC_1 \cup SLCC_2 \cup ... \cup SLCC_n \] by \(FM\); 
(5) \(SLCCS\) is the set of optimized \(LCC\) according to the network factors.

IV DESCRIPTION OF AGENT LEARNING MODEL

Because of the difference resources which \(CC, CN,\) and the network provided in \(DNE\), their types must be considered. These types are described as follows:

Definition 10. Computing node type (CNT) can be defined by \(RSV\) \((r_{fus}, r_{num}, f_{dis}, f_{run})\) of the computing node. According to the real conditions of each CNT in \(DNE\), the CNT types can be divided into the computing node type set \(CSTS = \{CT_1, CT_2, ..., CT_l\}\).

Definition 11. Network type (NT) is defined as \(NT\) \((B, PRT)\), where \(B\) denotes as the network bandwidth of \(CC\); \(PRT\) indicates network protocols of \(CC\). According to the real condition of networks, network types can be divided into the network type set \(NTS\) = \(\{NT_1, NT_2, ..., NT_m\}\).

The agent rules are described as follows:

(1) Basic rule (br) is defined as \(br\) \((id, rul, MRS)\), where \(id\) is its identifier; \(rul\) is the description of \(br\); \(MRS\) is the meta-rules set for revising \(br\). The basic rule set (BRS) is the set of all basic rules that GCG includes.

(2) Dynamic rule (dr) is defined as \(dr\) \((ct, nt, br, rul, w, sta, life)\), where \(ct \in CSTS, nt \in NTS, br \in BRS; \) \(rul\) is the formalization description of \(dr\); \(w\) is the is its weight value; and \(sta\) is its state, and \(sta \in \{\text"Naive}, \text"Trainable", \text"Stable\}; \) \(\text"Naive\) denotes that the \(dr\) is a new rule; \(\text"Trainable\) denotes that the \(dr\) is revising rule; \(\text"Stable\) denotes that the \(dr\) is a mature rule; \(\text"Life\) is the its life value; 

(3) If \(dr\) is a dynamic rule and \(dr.w > \text"MaxWeight\), which \(\text"MaxWeight\) is a constant in GCG, we call \(dr\) as the static rule (sr), its state is \"Static\";

(4) If \(dr\) is a dynamic rule and \(dr.w < \text"MinWeight\), which \(\text"MinWeight\) is a constant in GCG, we call \(dr\) as castoff rule (cr). Its state is \"Castoff\".

The state graph of rules is presented in figure 3.

The dynamic knowledge is the set of all the dynamic rules in GCG. The static knowledge is the set of all static rules. The basic knowledge can be earned by passive learning. To adapt the variety of computing resources, the dynamic rules must be generated at the start of the computing and be revised during the TSK computing process. Therefore, reinforcement learning can be adopted in the revising mechanism. Resources utilization rate for TSK is very important reinforcement factors.

Suppose that \(Y_1\) is the castoff threshold, and \(Y_2\) is the mature threshold; \(Q\) \((urt)\) denotes as the reinforcement function, and \(Q\) \((urt) > 0.urt\) indicates resources utilization rate; \(\text"MaxWeight\) is the maximum of the rule weight, and \(\text"MinWeight\) is the minimum of the rule weight, and let \(\text"MinWeight < Y_1 < Y_2 < \text"MaxWeight\); \(\text"MaxLife\) be the maximum of life value. The revising process is as follows:

(1) Suppose that a computing agent \(CA\) adopted a dynamic rule \(dr\) of CA.KS,
(2) \( dr\cdot life + + ; \) //increase the value of life
(3) wait for the \( urt \) from MAS;
(4) If \( urt > 0 \) then \( dr\cdot w = dr\cdot w + Q(urt); \) //increase weight
    If \( urt < 0 \) then \( dr\cdot w = - Q(urt) \); //decrease weight
(5) If \( dr\cdot w > MaxWeight \) then \( dr\cdot w = MaxWeight \);
    If \( dr\cdot w < MinWeight \) then \( dr\cdot w = MinWeight \);
(6) If \( dr\cdot w = Y \) and \( dr\cdot life > MaxLife \) then \( dr\cdot sta = \text{"Castoff"}; \)
//Castoff rule
    If \( Y < dr\cdot w < MaxWeight \) then \( dr\cdot sta = \text{"Stable"}; \)
//Stable rule
    If \( dr\cdot w >= MaxWeight \) then \( dr\cdot sta = \text{"Static"}; \) //Static rule
    If \( Y < dr\cdot w < Y \) then \( dr\cdot sta = \text{"Trainable"}; \) //Trainable rule
    If \( MinWeight < dr\cdot w < Y \) then \( dr\cdot sta = \text{"Naive"}. \)

V Process of CPCG

The CPCG computing process is as follows:
(1) The user agent commits the tasks to GCC, and their states is “Committed”;
(2) GA of GCC gets a task TSK that state is “Ready”;
(3) GA does the initialization works for TSK;
    Set \( C_{phase} = 0; \) // computing phase counter
    GA decides the computing mode for TSK by the conditions of DNE;
(4) GA allots a computing device PE (that is a CN or a LCC) for TSK; if this computing phase for TSK is DPC mode, GA get a LCC (That is constructed by GA through fuzzy partition method) for TSK; if this computing phase for TSK is TC mode, GA get a single computing node CN for TSK, Set \( C_{phase} = C_{phase} + 1; \)
(5) LA and N/A, which PE (CN or LCC) includes, constructed computing Agents (ICA or CCT) for TSK;
(6) All computing agents (ICA or CCT) for TSK construct the dynamic knowledge;
(7) All computing Agents calculate the TSK in cooperative, and they construct the stage checkpoint (Section 6.1) during the computing process, and they start the rule revising functions to revise the dynamic knowledge;
(8) If the TSK must be migrated, GA starts the migration process (Section 6.2 and 6.3) and then does
    \( \text{Set } C_{phase} = C_{phase} + 1; \) // next computing phase
    Go step (6) to continue the next computing phase ;
(9) If TSK be finished, GA receives the TSK from all N/A of PE and save the new knowledge into its knowledge base.

VI MIGRATION PROCESS

A. Checkpoint for DPC

In order to support the CDPC, we discuss the effective checkpoint mechanism. The communication checkpoint mechanism avoids the additional spending of coordinated checkpoint. The stage checkpoint can be described through XML language.

B. Migration Strategies

CDPC process involves a series of migrations, and the migration process can be classified as four sorts: (1)ICA \( \rightarrow \text{ICA}; \) (2)ICA \( \rightarrow \text{CCT}; \) (3) \( \text{CCT} \rightarrow \text{ICA}; \) (4) \( \text{CCT} \rightarrow \text{CCT}. \) The former three sorts are very simple, but the last one must be discussed in detail.

Now suppose that \( \text{CCT} \) runs on a logical cluster \( \text{LCC} \) and \( \text{CCT} \in \text{MIG} \cup \text{NIMG}, \) where \( \text{MIG} \) is the set of the computing agents that will be migrated; \( \text{NIMG} \) indicates the set of the non-migration computing agents;
(1) If $\text{NMIG} \neq \emptyset$, all CCA$s$ of CCT will be migrated into a new logical cluster NLCC, and the NLCC is optimized, so the migration process is simple; (2) If $\text{NMIG}$ is not $\emptyset$, all CCA$s$ of MIG will be migrated into a new logical cluster NLCC, and CCT will run on two logical computer clusters NLCC and RLCC (RLCC is the subset of LCC which NMIG runs). NLCC $\cup$ RLCC could not be optimized, so we construct the migration functions:

$$
\text{MF} = \frac{Z1 \times \text{TotalPower}(\text{NLCC}) \times \text{MBw}(\text{RLCC}, \text{NLCC})}{Z2 \times \text{Dist}(\text{RLCC}, \text{NLCC})}
$$

Where $\text{TotalPower}(\text{NLCC})$ indicates total computing power of NLCC; $\text{Dist}(\text{RLCC}, \text{NLCC})$ denotes their difference of $\text{RSV}$; $\text{MBw}(\text{RLCC}, \text{NLCC})$ denotes network skip distance between RLCC and NLCC; $\text{MBw}(\text{RLCC}, \text{NLCC})$ gives the indicative of minimal network bandwidth which the path from RLCC to NLCC includes; $Z1$ and $Z2$ are the coefficient.

If $\text{MF} > Y4$ ($Y4$ denotes as migration threshold), the migration will be executed; if $\text{MF} < Y4$, the sub-tasks from the computing agents of MIG will be retractible, and the CCT (NMIG) will run on RLCC.

C Migration Process
We present the process that ICA to ICA migrates and CCT to CCT migrates.

Algorithm 6.1. Migration of ICA to ICA

An ICA running on the computing node $CN_{i}$ must be migrated into the computing node $CN_{j}$, so the migration and learning process is presented as follows:
(1) ICA commits its $\text{CE}$ and $\text{KS}$ to $\text{NA}$ in $CN_{i}$, and $\text{NA}$ in $CN_{j}$ saves the $\text{KS}$ and $\text{CE}$;
(2) ICA asks for a new computing node $CN_{j}$ from GA, and it consults with $\text{NA}$ of $CN_{j}$, and migrates into $CN_{j}$;
(3) ICA gets the current knowledge about $CN_{j}$, and the other agents from $\text{NA}$ of $CN_{j}$;
(4) ICA and $\text{NA}$ of $CN_{j}$ learn each other, and they resolve the conflicts, and they refresh their $\text{KS}$ and $\text{CE}$;
(5) $CN_{i}$ and $CN_{j}$ report their $\text{KS}$ and $\text{CE}$ into LA and GA;
(6) ICA continues the next computing phase and Learning process on $CN_{j}$ at the newest stage checkpoint.

Algorithm 6.2. Migration of CCT to CCT
Suppose that CCT(id, Am, CAS, BDI, CKS, CCE) is a cooperation computing team running on the logical computer cluster LCC, and $\text{CAS} = \text{MIG} \cup \text{NMIG}$, where MIG indicates the set of the computing agents that will be migrated; NMIG indicates the set of the non-migration computing agents. The migration process involves three sub-algorithms.

Algorithm 6.2.1

While $\text{NMIG} \neq \emptyset$ and $\text{Am} \in \text{NMIG}$, the migration process is as follows:

(1) All the computing agents of MIG migrate into their new computing nodes severally, and learn the new knowledge according to the alogrithm 6.1;
(2) For all CCA $\in \text{MIG}$ do (3)(4)(5);
(3) CCA learns the cooperation knowledge from Am of NMIG, and consults to resolve the conflicts;
(4) If the conflict consultation was failure, CCT will retract the task $t$ from CCA and expel CCA: /*$\text{MIG} = \text{MIG} \setminus \{\text{CCA}\}$*/
(5) If the conflict consultation is successful, then $\{\text{CCA}$ refreshes its $\text{KS}; \text{NMIG} = \text{MIG} \cup \{\text{CCA}\}; \text{MIG} = \text{MIG} \setminus \{\text{CCA}\}\}$;
(6) CCT continues for executing the next computing phase at the newest stage checkpoint.

Algorithm 6.2.2
While $\text{NMIG} \neq \emptyset$ and $\text{Am} \notin \text{NMIG}$, the migration process is as follows:

(1) All the computing agents in NMIG cooperate to elect a new main control agent Anm from NMIG, Am submits the cooperation knowledge about CCT to Anm;
(2) Do the algorithms 6.2.1;

Algorithm 6.2.3
While $\text{NMIG} = \emptyset$, the migration process is as follows:

(1) Am migrates into the new computing node according to algorithm 6.1, and notices GA;
(2) Am broadcasts its new conditions to all members of CAS of CCT; The other members of CAS receive the messages from Am, and they do the migration by algorithm 6.1, and they submits the learning results to Am;
(3) After all the cooperation computing agents finish the migration process, CAS cooperate to produce the new CKS by Am control ;
(4) CCT continues the next computing phase at the newest stage checkpoint.

VII EXPERIMENTS
We have performed a number of experiments to verify the CGCP model in the DNE, which is composed of 24 computers and 3 physical computer-clusters connected by Intranet. All the computers are classified as four groups depending on the types of CPU, memory, disk, and net adapter. These groups are $\text{RSV}$ (3000MHZ, 512MB, 7200RPM, 100M), $\text{RSV}$ (2600MHZ, 256MB, 7200RPM, 100M), $\text{RSV}$ (1600MHZ, 256MB, 5400RPM, 1000M), and $\text{RSV}$ (1200MHZ, 128MB, 5400RPM, 100M). The operating systems of the computers are the Windows series or Linux. For the performance evaluation, there are three sets of continuance data parallel computing tasks which are the matrix operations, the linear programming and JOIN operation that is a most important operation of parallel relation in the parallel relation database. We programmed the DPC edition and TC edition for these operations in order to support CDPC. The Matrix operation $\text{RDV}$ is (0.3, 0.3, 0.2, 0.2), and the linear programming $\text{RDV}$ is (0.5, 0.3, 0.1, 0.1), and JOIN
operation RDN is (0.2, 0.3, 0.25, 0.25). The intranet clock is synchronous through GTS protocol. The initialization basic rules include 24 rules for ICA and 7 rules for CCT, and the parameter values are as follows: MaxLife=43200(s), Yf=15, Yr=80, Yp=0.3, MaxWeight=100 and MinWeight=0.

The experimentation includes seven times, and each time has 12 hours, and the total amount is 84 hours. The tests adopt a random function to choose some tasks (the matrix operation, the linear programming, and JOIN operation) in each time. In order to make the tasks to migrate as far as possible in the DNE, We make use of the random migration function RandMigration() to form the migration strategies during the test processes. Through the average values of the test information, we observe the test results of CPCG. The experiment results are as follows:

**Experiment1**. As shown in table 1, we can see the distribution characteristic that LCC spans many networks. Obviously, JOIN operation depended on the network bandwidth greatly.

<table>
<thead>
<tr>
<th>Network</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numbers that LCC spans operations</td>
<td>Matrix operation</td>
<td>0.81</td>
<td>0.19</td>
</tr>
<tr>
<td>Liner programming</td>
<td>0.12</td>
<td>0.78</td>
<td>0.10</td>
</tr>
<tr>
<td>JOIN operation</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Experiment2**. We tested the effective scales of the logical computer cluster (LCC) for three types of DPC. The test results are presented in Table2.

<table>
<thead>
<tr>
<th>DPC</th>
<th>Effective scale</th>
<th>Liner programming</th>
<th>JOIN operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix operation</td>
<td>About 8</td>
<td>About 10</td>
<td>About 5</td>
</tr>
</tbody>
</table>

**Experiment3**. Fig 5 presents the comparison the distribution of dynamic rules, which are generated during the learning process. The solid line denotes the distribution of dynamic rules that their state is always “Naive” during their life period. The dotted line denotes the distribution of the “Trainable” dynamic rules that their state had become the “Stable” during their life period. It can be seen from the figure that the efficiency of this learning model is increased gradually along with computing process.

**Experiment4**. As shown in fig 5, the thick solid line denotes the average resources utilization rate of DNE. No doubt, CPCG can raise resources utilization rate of DNE consumedly.

**Experiment5**. Table 3 presents the comparison the average time rates of computing mode (TC, DPC) for three kind CDPC computing task. The test result show that JOIN operations depend on its large data and its migration cost is very high; the others can fit for this migration model.

<table>
<thead>
<tr>
<th>Operations</th>
<th>The time rate of DPC</th>
<th>The time rate of TC</th>
<th>Suspended time rate</th>
<th>The time rate of migration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix operation</td>
<td>0.66</td>
<td>0.15</td>
<td>0.04</td>
<td>0.15</td>
</tr>
<tr>
<td>Liner programming</td>
<td>0.77</td>
<td>0.12</td>
<td>0.02</td>
<td>0.09</td>
</tr>
<tr>
<td>JOIN operation</td>
<td>0.35</td>
<td>0.00</td>
<td>0.37</td>
<td>0.28</td>
</tr>
</tbody>
</table>

**VI CONCLUSIONS**

Because of the heterogeneous resources, the different power of computers, and the migration of computing task, the effective use of resources is very difficult for the grid computing in the dynamic network environment. Through the techniques of ICA, CCT, the dynamic learning and the logical computer cluster partition based on fuzzy theory, the stage checkpoint, and cooperation migration, CPCG can support CDPC. These approaches can raise the resources utilization rate in DNE. It can fit for the grid computing in DNE.

**ACKNOWLEDGMENT**

We would like to thank the support of National Nature Science Foundation of China (No.60573108), the Innovation Program of Shanghai Municipal Education Commission (No. 08ZZ76, 07ZZ92), and Shanghai leading academic discipline project (S30501).

**REFERENCES**


© 2010 ACADEMY PUBLISHER


Qingkui Chen was born in 1966. He is a Professor and Ph.D supervisor of Scholl of Optical-Electrical and Computer Engineering at University of Shanghai for Science and Technology (USST), Shanghai, P. R. China. He is the Vice Dean of School of Optical-Electrical and Computer Engineering. Qingkui Chen is a senior member of China Computer Federation in China. His research interests include network computing, parallel computing, parallel database and computer network.He is the head of many programs which were supported by the Natural Science Foundation of China (NSFC) and the Shanghai Natural Science Foundation of China. Prof. Chen also served as program committee member of IFIPInternational Conference on Network and parallel Computing (NPC) and as the Technical Program Committee Co-Chairs of ICISE.

Haifeng Wang was born is 1976. He is a Ph.D.Candidate of Business school at University of Shanghai for Science and Technology, P. R. China. His current major research interests are in the areas of parallel computing, network computing.

Wei Wang was born is 1983. He is a postgraduate of Scholl of Optical-Electrical and Computer Engineering at University of Shanghai for Science and Technology (USST), Shanghai, P. R. China. His research interests include network computing, parallel computing.
FAPP: A New Fairness Algorithm for Priority Process Mutual Exclusion in Distributed Systems

Sukendu Kanrar
Narasinha Dutt College, Howrah, India
Email: sukhren2003@yahoo.co.in

Nabendu Chaki
University of Calcutta, 92 A. P. C. Road, Kolkata, India
Email: nabendu@ieee.org

Abstract — In this work, we have proposed a new token based Fairness Algorithm for Priority Processes (FAPP) that addresses both the issues and keeps the control message traffic reasonably low. One major limitation of the token based mutual exclusion algorithms for distributed environment like Raymond's well-known work on inverted-tree topology lies in the lack of fairness. Another aspect of the problem is in handling the prioritized processes. In one of our earlier works, both fairness and priority have been addressed in proposing an algorithm MRA-P. However, MRA-P suffered from some major shortcomings like lack of liveness, high message complexity, etc. The proposed FAPP algorithm, in spite of considering priority of processes, ensures liveness in terms of token requests from low priority processes. Formal verification on FAPP justify properties like correctness of the algorithm, low message complexity, and fairness in token allocation.

Index Terms — Mutual exclusion, Priority, Fairness, Correctness, Token based algorithm, Inverted tree topology.

I. INTRODUCTION

Distributed mutual exclusion algorithms are either token-based or non token-based. In token-based mutual exclusion algorithms, a unique token exists in the system and only the holder of the token can access the CS. As we study the existing token based approaches, it has been observed that the fairness aspect in terms of responding to the token request in a FCFS order, is often not addressed in many of the otherwise well appreciated solutions.

Priority, on the other hand, may be associated with each process, and a process is allowed to enter into its CS only when no higher priority process is pending.

The authors have worked on performance analysis of various logical interconnection topologies for quite some time [2, 3]. The tree-structure has been selected as the logical topology for its simple, yet highly flexible connection topology. Our approach uses a token-based model working on a logical tree structure, which is dynamically modified. The proposed algorithm is a token-based one. It allocates the token to the requesting process with the highest priority. Equal-priority processes are served following the FCFS order to ensure fairness of allocation.

Earlier, we have proposed a token-based mutual exclusion algorithm MRA-P with the same object [1]. The earlier work, however, suffered from high message complexity and relatively complex routing mechanism. Besides, the MRA-P algorithm lacks in ensuring liveness of token requests, particularly from the low-priority processes. The "progress towards critical section is a liveness property. Even if there is no deadlock, the progress is violated if there exists at least one infinite behavior, in which a process remains outside its critical section" [15].

However, in the proposed FAPP (Fairness Algorithm for Priority Processes) algorithm that has been introduced in this paper, we have successfully addressed the two together. The proposed FAPP is a completely new algorithm that follows a simpler routing, maintains lesser information in each node and has significantly lower message complexity comparing MRA-P.

II. CONTEMPORARY WORKS

A number of solutions have been proposed for prioritized mutual exclusion in a distributed system. Some of these approaches are suitable for real-time application but impose a higher message passing overhead. In Raymond's [5] token based mutual exclusion algorithm, requests are sent over a static spanning tree of the network, toward the token holder. This logical topology is similar to that of Housn and Trehel's approach [7]. Suzuki and Kasami proposed a Broadcast Algorithm [8], in which each site Si, keeps an array of integers RNij[1..N], where RNij is the largest sequence number received so far from SJ. The Performance of Suzuki-Kazami algorithm, 0 or N messages per critical section executed and synchronization delay = 0 or T. Based on Suzuki-Kazami algorithm, Singhal proposed a heuristically aided algorithm that uses state information to more accurately guess the location of the token [11]. An algorithm, based on a dynamic tree structure, was proposed by Trehal and Naimi [12]. The initial Naimi-Trehel algorithm takes into consideration another pointer next, useful when a node requests the critical section before the previous requester leaves the critical section. A variation of it has been presented in [13].

Nontoken-based mutual exclusion algorithms like Lamport’s [9] exchange massages to determine which
process can access the CS next. Lamport’s algorithm is permission based and requires 3*(N-I) massages to provide mutual exclusion. Permission based algorithm proposed by Ricart and Agrawala, reduced the number of required massages to 2*(N-I) messages per critical section entry [4]. Mueller [6] introduced a prioritized token-based mutual exclusion algorithm. In many of the solutions the fairness criterion is considered to be equivalent to the priority issue, i.e. a job that arrives early is defined as a higher priority job [14].

In [1], we have introduced a token-based mutual exclusion algorithm, called MRA-P, to maintain fairness amongst processes. The earlier work, however, used too many control messages across the network. In section III, we have presented a brief discussion on MRA-P for the sake of completeness of this paper.

III. MRA-P ALGORITHM

The MRA-P algorithm offers a solution for prioritized mutual exclusion in a distributed system. An additional data structure is maintained in all the nodes as introduced below.

Priority queue (PQ): Every process Pi maintains a priority queue PQi. A process Pi wishing to enter the critical section, sends a request for the token along with LDQj and its priority status on its outgoing edge. When a process Pm receives a token request from another process Pn, there are three possibilities.

1. The priority queue of Pm is empty: Pm enters the priority of Pn in PQm.
2. The priority queue of Pm is not empty and priorities of the two processes are not the same: Pm enters both the priorities in PQm with the higher priority entered first.
3. The priority queue of Pm is not empty and priorities of the two processes are same: Pm adds the priority of Pn in its PQm.

The MRA-P algorithm to enter CS considering the priority of processes as proposed in [1] is given below:

Begin
Step 1: A Process Pi wishing to enter the CS,
   1.1 Pi enters its id i in the LDQi and priority in PQi;
   1.2 Pi sends a token request {i} on its out-edge along with LDQj and the priority of Pj;
Step 2: When a process Pm ≠ Phold receives a token request from process Pn,
   2.1 Pm adds priority in PQm in ascending order;
   2.2 According to priority, Pm adds the ids from LDQn into LDQm followed by its own id, i.e., m;
   2.3 The token requesting process id i is added to the RQ;
Step 4: On completing the execution of CS, Pm=Phold performs the following
   4.1 Pm scans the first id k from RQ;
   4.2 Pm extracts entries starting from k to the first occurrence of m from LDQn and removes the first element of PQm;
   4.3 reverse the extracted sequence of ids;
   4.4 reverse the directed edge from Pn to Pm, where n is the id that immediately follows m;
   4.5 Phold = Pn;
   4.6 Pass RQ to Pn from Pm along with the token;
   4.7 If LDQm is not empty then
       4.7.1 Pm places a token request to Pn along with the reduced LDQm;
       4.7.2 The id of LDQn is added to the head of LDQn followed by n;
Endif.
Step 5: The newly designated root Pn = Phold performs the following
   5.1 If the first id of RQ is n, then
       5.1.1 remove the first id of RQ i.e., n;
       5.1.2 remove first element of PQ;
       5.1.3 entry n is removed from the top of LDQn;
   5.1.4 Pn enters its CS.;
   else
       5.2.1 scan and extract the first id k from LDQn;
       5.2.2 extract entries of LDQn from k to n;
       5.2.3 reverse the directed edge from Pn to Pk, if the extracted sequence is of length 2;
       5.2.4 hand over the token and RQ to Pm;
Endif.
End.

IV. LIMITATIONS OF MRA-P AND BEYOND

Before we discuss the limitations of the MRA-P algorithm, let us be precise on the parameters that are being considered to evaluate the performances of the mutual exclusion algorithms under consideration.

A. Fairness, Priority and Liveness

The terminologies that we have been using in the paper are defined in this section. The definitions below clearly mark the scope for each keyword in the context of our paper.

- Fairness: The definition of fairness that we follow here implies that if the token-holder Phold receives a token request from some arbitrary process A ahead of the same from some other process B, then the algorithm ensures that after Phold releases the token, process A gets it before process B.
- Priority of a process: We assume that the processes have some pre-assigned priorities. This implies that
if the token request from a higher priority process A is pending, then the token must not be allotted to processes having priority lower than that of A. The priority is quantified by putting a pre-assigned integer priority number for each process. A higher priority process comes with a great priority number.

- Priority-based fairness: Taking into account the priority aspect, we therefore revise the above definition of fairness with a stronger definition below.

The revised definition of priority-based fairness implies that the token must be allocated to some process A such that among all the processes having priority equal to that of A, the token request from A has reached $P_{hold}$ ahead of others and there is no pending token request from any other process B having priority higher than that of A.

- Liveness: The liveness property implies that any process A that requests for the token must get it eventually.

- Safety: A mutual exclusion algorithm is safe if it ensures that no two processes would enter the respective critical sections simultaneously.

- Correctness: The correctness of a control algorithm is a combination of the liveness along with safety. A control algorithm is correct if it confirms to both liveness and safety aspects.

### B. Dynamic Process Priority

The above definitions of priority-based fairness and liveness are directly in conflict with one another. Let’s assume that a process A with a low priority requests for token first. After A’s request, other processes B, C, ..., N all with priorities higher than that of A places there requests for token. In order to ensure the priority-based fairness defined above, all the processes B, C, ..., N would receive the token ahead of A. In fact, process A would not ever get the token if there is some pending token request from a higher priority process, irrespective of its time of request. In this situation the property of liveness is violated. This exactly had been the major limitation of one of our earlier algorithm MRA-P [1].

In our attempt to propose a solution that would strike the balance between priority-based fairness and liveness as have been defined above, we proposed to dynamically upgrade priorities of processes waiting in the queue for the token after placing the token request. The initial proposal had been somewhat like the following.

**Rule 1:** When a node A finds a token request from another process B with priority $r$ then for all such process C in $PL_{A}$ whose priority $t$ is less than $r$, the process priority of C is increased by 1 to $t + 1$. If priority of process C, is expressed $p(C)$ then,

$$\forall C \in PL_{A} \land p(C) < p(B) \Rightarrow p(C) := p(C) + 1$$

The list in A is re-constructed with this increased process priority for C. The motivation behind this is to elevate the priority of a process C every time some higher priority process B supersedes C in the request queue. This would make sure that even a token request from the otherwise lowest priority process would eventually be satisfied as the priority itself changes dynamically.

**Rule 2:** After a process gets the token it enters the critical section. As it comes out of the critical section, the priority of a process is reset to its original value.

### V. FAPP: Fairness Algorithm for Priority Processes

The proposed algorithm uses a request queue for each participating process. These are stored and maintained locally using the FAPP.

#### A. The Data Structure

We propose to maintain locally request queues for each process A. The initial priorities of the token requesting processes would be stored in the sorted request queues. The proposed algorithm describes how the priority values in these queues are revised and re-positioned to take care of the fairness, priorities as well as liveness of the solution. This would help towards efficient routing of the token and requests for the token from the participating nodes. The data structure has been introduced below.

Request Queue (PL): An arbitrary process A maintains a process request queue (PLA) as detailed below:

$$PL_{A} = \{[<R, R_{P}>], [<S, S_{P}>], [<T, T_{P}>], \ldots \}$$

where $R, S, T, \ldots$ are the direct descendents of A and $R_{P}, S_{P}, T_{P}, \ldots$ are the priorities of the respective descendant processes in a non-increasing sequence, i.e., $R_{P} \geq S_{P} \geq T_{P}$.

Any arbitrary process A with priority value $A_{P}$ wishing to enter its CS, makes an entry $<A, A_{P}>$ in the appropriate position in the sorted sequence of its own request queue $PL_{A}$. Again if the node A receives a token request from one of its descendents, say $B$, with priority $B_{P}$, then entry $<R, R_{P}>$ is inserted in the appropriate position of $PL_{A}$.

The actual number of entries in the request queue maintained at any process depends on the number of token requests from the descendents processes. The PL for some node A may be NULL if no token request is generated or passed to the respective node. On the other hand, the request queue for node A may have at most $(n+1)$ entries in it if A has a total n number of descendents, each of which and A itself, placing a request for the token. The entries in the request queues maintained locally will always be sorted in the non-increasing order based on the priorities of the token requesting processes.

#### B. The Description of FAPP

The proposed fairness algorithm for priority processes (FAPP) assumes an inverted tree structure as the logical topology where competing processes form vertices of the tree. The edges in an inverted tree are directed from child to parent node. Each node $P_{r}$, other than the root node, has exactly one out-edge $(P_{r}, P_{t})$, where $P_{t}$ is the parent of $P_{r}$ in the tree. The processes enter into the respective
critical section in a mutually exclusive manner. A token is shared by these processes. The process $P_{\text{hold}}$ that is holding the token at any instance is in the root of the tree. The root changes each time the token is transferred to some other process. The new node that gets the token is designated as $P_{\text{hold}}$ and forms the new root of the tree.

Step 1: When a process $M$ with priority $\{r\}$ wants to enter the critical section (CS)

insert $<M, r>$ in $PL_M[]$;  
/* According to priority $r$, it adds the tuple $<M, r>$ into $PL_M$ */
send token_request $<M, r>$;

Step 2: When a process $S \neq P_{\text{hold}}$ receives a token request $<K, r>$ and priority $\{r\}$ from another process $K$,

if ($r > t$) 
/* Priority of $K$, i.e., $r$, is higher than the priority $t$ for some process $C$ whose token request is sent through $S$. Priority value $t$ is increased by 1 i.e., $t = t+1$. */
endif
insert $<K, r>$ in $PL_S[]$ in the sorted order of descending priority;  
/* Tuple $<K, r>$ is inserted into $PL_S$ according to priority */
send token_request($S, r$);  
/* $S$ sends token request only when it receives request with higher priority or after update priority */

Step 3: When a process $J = P_{\text{hold}}$ receives a token request $<K, r>$ and priority $\{r\}$ from another process $K$,

if ($k \in PL_J[]$)  
/* there is a pending request from $P_k$. replace old priority of $K$ with $r$; */
the new request from $P_k$ must have a higher priority [step 2] */
update $PL_J[]$;  
/* revise $PL_J$ in accordance with decreased priority */
if ($r > t$)
endif
insert $<K, r>$ in $PL_S[]$ in the sorted order of descending priority;  
/* According to priority, it add the tuple $<K, r>$ into $PL_S$; update $PL_S$ in descending order of entries */
send token_request($S, r$);

Step 4: On completing the execution of a CS, $E_j = P_{\text{hold}}$ performs the following

remove the first process tuple $<M, m>$ from $PL_E$;  
/* $m$ is priority of process $M$ */
send token($M, m$);
/* $M$ would be the new $P_{\text{hold}}$ - the token is passed to $M$ from $E_j$ */
if ($PL_E[]$ ≠null)
send dummy token_request($M, x$);
/* $E_j$ places a token request to $M$ along with the highest priority that $E_j$ receives. */
endif

Step 5: The newly designated root process $M = P_{\text{hold}}$ that receives token $(G, r)$ performs the following

remove the first tuple $<G, r>$ from $PL_M$;  
/* $r$ is priority of process $G$ */
if ($G = M$)
/* after executing CS, process works with its original initial priority */
enter CS;
else
send token($G, r$);  
/* $G$ would be the new $P_{\text{hold}}$ - the token is passed to $G$ from $M$ */
endif

The run-time complexity of the FAPP algorithm is same as that of the Raymond’s algorithm. A discussion on the characteristics of the algorithm along with the formal proofs is presented in section VII.

VI. ILLUSTRATION OF THE PROPOSED FAPP ALGORITHM

The example under consideration deals with six processes A, B, C, D, E, F, G, H and K with the priority values 1, 5, 3, 2, 2, 4, 3, 5 and 4 respectively. We further assume that the priority 5 is highest priority of the process. Process G places the first request for the token with the root node A = $P_{\text{hold}}$. Next, B, C, and K processes have placed the requests for token in that order.

As per step 1 of the proposed algorithm, $PL_G$ stores $<G, 3>$. The first token request is then propagated to D along with the id of the requesting node, i.e., G in this case, and priority of the requesting process 3. D on receipt of the request from G, puts the content of G in $PL_D$, thus $PL_D$ = $<G, 3>$. D now places a token request along with its own id and priority of process G i.e. 3, again, on its out edge. A = $P_{\text{hold}}$ now receives the request and puts $<D, 3>$ in $PL_A$.

Let’s now consider that, at this point of time B issues the 2nd token request to enter CS. B has a priority of 5. So its puts its own process id and priority i.e., $<B, 5>$ in $PL_B$ and send the token request along with its own id and priority on its outgoing edge. A receives the request and following step 3 of the algorithm, it modifies $PL_A$ to $<B, 5>$; $<D, 4>$. Process A first increase the priority of D which originated by G, by one, then arranges the requests. Similarly Process C and process K issues the third and forth token request respectively. PL lists of every node shown in figure 1. Now process H sent fifth request using $<H, 5>$ to F. Process F updates $PL_F$ as $<K, 5>$, $<H, 5>$ . Then process D updates as $<G, 5>$, $<F, 5>$. Process D sent the modified tuple $<D, 5>$ to A. After receiving request from D, Process A Updates $PL_A$ as $<B, 5>$, $<D, 5>$ and
Let’s assume that now, A comes out of its CS and following Step 4, the first tuple from PL_A is extracted and token is handed over to B. Now, PL_A=<D, 5>, <C, 3> . As PL_A is not empty, a dummy request from A is sent to B along with the highest priority, which Process A receives i.e., token request as (A, 5). The token reaches B and at this point, PL_B=<B,5>, <A, 5>. Process B extracted <B, 5>. The condition for step 5.1 matches and therefore B enters the CS after updating PL_B=<A, 5>. Therefore, as B comes out of the CS, the entries from PL_B is extracted once again and the token is handed over to A and leaving PL_B=φ and Once again, A becomes the root of the inverted tree. Therefore using step 4 and 5, process G, K, H and C enters in CS respectively. When C enters the CS after removing its tuple from PL_C, all the lists are all emptied now. The token stays with C = Phold even after the process comes out of the CS and till any other request is generated in the system.

Table I is a tabular representation that illustrates how the content of the PLs maintained at different nodes change in successive iterations as explained above. The token requests from G, B, C, K and H are placed in step 1, step 2, step 3, step 4 and step 5 respectively. The process B enters CS in step 6. Process G enters CS in step 7 and K enters CS in step 9 and Process C enters CS in step 10. These are marked by placing a * before the PL entries for the process in the CS, in the appropriate cell of the table. The # symbol indicates that a new token request is issued. The <0> in the cells of the tables are used to indicate null lists. In the case study, the token is finally left with C as know other request for the token has been issued.

**VII. PERFORMANCE ANALYSIS FOR FAPP ALGORITHM**

We will analyze the performance of our algorithms using following performance parameters: message complexity per CS request, Fairness, Correctness of the FAPP algorithm, Time complexity of the FAPP Algorithm, synchronization delay and maximum concurrency for proposed FAPP algorithm.

**A. Message Complexity**

The control messages for the FAPP algorithm may be placed in two categories. The request control messages flow from the token requesting node towards the root node. These control messages also carry the priority information of the requesting node. The other type of control messages are referred as token transfer control message. These messages move from the root node Phold towards the token requesting node after Phold comes out of its own critical section. One can make a quantitative estimation for the number of control messages per critical section access by separately computing the request control messages and the token transfer control messages. When a node Pt makes a request for the token, its id {i} and the priority value, say {p} both moves up to its parent node Pj. The node Pj inserts {i} in PLj and {p} in PRLj before sending a fresh request {j} along with the priority value {p} to its own parent. The relaying of token request from originating node Pi may move right up to the root node Phold.

The process, may however, stop at any intermediate node Pk, if there is already a pending request for Pj whose priority is higher than that of Pt.

**Table I.**

<table>
<thead>
<tr>
<th>P id</th>
<th>Step 1</th>
<th>Step 2</th>
<th>Step 3</th>
<th>Step 4</th>
<th>Step 5</th>
<th>Step 6</th>
<th>Step 7</th>
<th>Step 8</th>
<th>Step 9</th>
<th>Step 10</th>
<th>Step 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>&lt;D,3&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;C,3&gt;</td>
<td>&lt;C,3&gt;</td>
<td>&lt;C,3&gt;</td>
<td>&lt;C,3&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>B</td>
<td>#&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;B,5&gt;</td>
<td>&lt;A,5&gt;</td>
<td>&lt;A,5&gt;</td>
<td>&lt;A,5&gt;</td>
<td>&lt;A,5&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>C</td>
<td>#&lt;C,3&gt;</td>
<td>&lt;C,3&gt;</td>
<td>&lt;A,3&gt;</td>
<td>&lt;A,3&gt;</td>
<td>&lt;A,3&gt;</td>
<td>&lt;A,3&gt;</td>
<td>&lt;0&gt;</td>
<td>&lt;0&gt;</td>
<td>&lt;0&gt;</td>
<td>&lt;0&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>D</td>
<td>#&lt;G,3&gt;</td>
<td>&lt;G,3&gt;</td>
<td>&lt;G,3&gt;</td>
<td>&lt;G,3&gt;</td>
<td>&lt;G,3&gt;</td>
<td>&lt;G,3&gt;</td>
<td>&lt;F,3&gt;</td>
<td>&lt;F,3&gt;</td>
<td>&lt;F,3&gt;</td>
<td>&lt;F,3&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>E</td>
<td>#&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>F</td>
<td>#&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>H</td>
<td>#&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;H,5&gt;</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>K</td>
<td>#&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;K,4&gt;</td>
<td>&lt;0&gt;</td>
</tr>
</tbody>
</table>
Thus, the maximum number of request control messages for each token request from $P_i$ could be $L$, where $L$ is the level of node $P_i$ in the tree. The following lemmas are defined and proved to give an estimate of the number of request control messages that carry the request path information from the request originating node, say, $P_i$.

**Lemma 1:** The number of request control messages along with the request paths that may be triggered by an initial token request from any node $P_i$ lies in the interval $[0, (H-1)]$, where $H$ is the height of the tree.

**Proof:** The total number of nodes on the unique path from the requesting node $P_i$ to root node $P_{\text{hold}}$ is $L+1$, where $L$ is the level of node $P_i$ in the inverted tree. In the worst case, the request control message from $P_i$ to its parent node along with the priority value of $P_i$ is relayed right up to the root node using the intermediate nodes. The total number of such messages will, therefore, be $L$. If $P_i$ happens to be a leaf node on one of the longest paths of the tree, then $L=H-1$ following the definition of height $H$ of the tree. The maximum number of request control message would thus be $L=(H-1)$.

It is trivial to prove that in an extreme case, if there is already a pending request of higher priority from $P_j$ to $P_i$ then no request control message with request path is sent at all. Thus, the total number of request control messages along with request path that may be generated in the system triggered by an original token request from any node $P_i$, lies within the closed interval of $[0, (H-1)]$, where $H$ is the height of the tree.

**Lemma 2:** The maximum number of token transfer control message for each token request from any node in the inverted tree is $(H-1)$, where $H$ is the height of the tree.

**Proof:** The FAPP algorithm stores the request paths of the tokens using local queues in each node. When the root node, say $P_{\text{root}}$ comes out of its critical section, it intends to pass the token to node, say, $P_j$, which is on top of the $P_{\text{hold}}$ and sends the token from $P_m$ to $P_j$. This transfer is performed using the intermediate nodes from $P_m$ to $P_j$, where a transfer control message is issued to mobilize the token through each intermediate node.

If the level of node $P_i$ is $L$, then the total number of nodes on the unique path from $P_m$ to $P_i$ would be $L+1$, inclusive of both the source and the target nodes. The number of transfer control messages along the path would therefore be $L$. The value of $L$, is again $(H-1)$ for the extreme case when target node $P_i$ happens to be leaf node on one of the longest paths of the tree of height $H$. Thus the statement of Lemma 2 is proved.

**Theorem 1:** The maximum number of control messages per critical section access for FAPP using a N-node balanced binary tree is $O(\log N)$.

**Proof:** The total number of request control messages along with the priority value $[p]$ of the original token requesting node $P_j$ is $L$, where $L$ is the level of node $P_j$ in the tree. If $P_j$ happens to be a leaf node on one of the longest paths in the tree, then the number of request control messages for a original request from $P_j$ would be $(H-1)$, where $H$ is the height of the tree. Lemma 1 establishes that $0 \leq p \leq (H-1)$ of these control messages carry the request depending on whether any token request of equal or higher priority is pending or not for some intermediate node on the path.

Similarly, the total number of transfer control messages that are generated for the token request from $P_j$ is at most $(H-1)$. Thus, the total number of control messages for each access to critical section cannot exceed $2^*(H-1)$. If the underlying inverted tree is a balanced binary tree, then its $H$ would be $\log_2 N$, $N$ being the total number of nodes in the tree. Thus the total number of control messages using a N-node balanced binary tree is $2^*(H-1) = 2^*(\log_2 N-1) = O(\log N)$.

The order of message could even be smaller for trees with $m$ children per node, for $m>2$. The message complexity for such a topology would be $O(\log N)$.

**B. Fairness**

A mutual exclusion algorithm is said to provide fair mutual exclusion if the following definition of fairness holds.

**Lemma 3:** Algorithm FAPP provides fair mutual exclusion as per the definition of fairness.

**Proof:** Let's start with a reverse hypothesis. We assume that the FAPP violates the fairness criteria. In other words, let's assume that, even if the request from process $P_r$ reaches the root node first, an equal priority process $P_i$ gets the token ahead of $P_r$. The allocation of token initiates by picking up the first element from $P_{\text{hold}}$ and then by passing the token towards the requesting node using the intermediate nodes. Without any loss of generalization, let's further assume that $P_i$ and $P_r$ are direct descendents of $P_{\text{hold}}$. This assumption just simplifies the scenario as we can do away with the routing through the intermediate nodes. We know that $P_i$ is to get the token before $P_r$, and priority of both processes are same.

Therefore, the occurrence of $P_i$ must precede that of $P_r$ in $P_{\text{hold}}$. \text{ .......... Condition 1} \text{ On the other hand, for token requests from the equal priority processes, the entries in $P_{\text{hold}}$ are to be made following the order of arrival of request control messages. There could be a change in this order by insertion into appropriate place, only when request from a higher priority process reaches $P_{\text{hold}}$. Such insertion, following step 3 of FAPP, does not however alter the relative sequence of the existing entries in the $P_{\text{hold}}$. $P_i$ and $P_r$ are of equal priority and the token request from $P_i$ arrives before that from $P_r$. \text{ .......... Condition 2} \text{ The conditions 1 and 2 are in conflict. This proves that the converse of our initial hypothesis is correct. Therefore, the Fairness Algorithm for Priority Processes (FAPP) indeed, meets the fairness criteria.}

**C. Correctness of the FAPP Algorithm**

The correctness of a distributed control algorithm is defined as a collection of two separate characteristics,
safety and liveness. A process synchronization algorithm satisfies the safety property if when any process \( P_i \) is in its CS, no second process \( P_j \) is allowed to enter the CS. In other words, the algorithm is safe if the competing processes access the critical sections in a mutual exclusion. Liveness demands that any process that requests for the token must get it eventually.

**Safety:** Mutual exclusion allows at most one process from any of the nodes to enter a critical section (CS) at any given instance of time. FAPP is a token based process where all the competing processes share the same token. A process is allowed to enter into critical section only after it gets the token. This makes sure that no two processes are in the critical section simultaneously. The process \( \text{P}_{\text{hold}} \) that is holding the token at any instance can only enter its CS. \( \text{P}_{\text{hold}} \) is in the root of the tree. The root changes each time the token is transferred.

**Liveness:** Liveness and priority-based fairness are just two conflicting aspects to handle. If an algorithm prefers a higher priority process, then its almost obvious that a lower priority job may suffer from starvation. The primary focus of this paper had been to strike a balance between priority-based fairness and liveness. The primary purpose of this paper had been to strike a balance between priority-based fairness and liveness. The proposed rule 1 in section IV B and its implementation in the step 2 of the proposed FAPP algorithm allows dynamically upgrade for priorities of processes. The rule increases the priority of a process \( A \) every time some higher priority process \( B \) supersedes \( A \) by virtue of its higher priority value. The mechanism ensures that even a token request from the otherwise lowest priority process would eventually be satisfied as the priority itself changes dynamically. Thus liveness as defined in section IV A would be maintained for FAPP.

**Concurrent occupancy:** In the proposed algorithm, before a process \( P_m = \text{P}_{\text{hold}} \) starts execution in CS, it extract the first id of \( \text{PL}_{m} \) and \( \text{PRL}_{m} \). If the first id of \( \text{PL}_{m} \) is \( m \) and the first element of \( \text{PRL}_{m} \) is equal to the own priority of \( P_m \), then, replace the extract positions of \( \text{PL}_{m} \) and \( \text{PRL}_{m} \) by the id of the sender and the priority sent by it. This is done only if the PRL and PL of the token sender are not empty. Then \( P_m \) enters its CS. The requesting process enters in its CS upon receiving the token. Hence, it is proved that the proposed FAPP algorithm satisfies the concurrent occupancy property.

**D. Time complexity of the FAPP Algorithm**

The FAPP algorithm does iterations in two stages for each critical section access. As explained in the proof of theorem 1, a maximum number of \((H-1)\) iterations, \( H \) being the height of the tree, are made while a token request is propagated to the root node. Similarly, to pass the token to a requesting leaf node, the number of iterations involving steps 4 and 5 will be \( H-1 \). The overall time complexity, would therefore be \( 2*(H-1) \approx O(\log N), N \) being the number of nodes in a balanced binary tree.

**Theorem 2:** The maximum concurrency of the proposed FAPP algorithm is \( L \), where \( L \) is the maximum level of any node in the tree.

**Proof:** In the proposed algorithm, the number of control messages per critical section access by separately computing the request control messages and the token transfer control messages. When a node \( P_i \) makes a request for the token, its id \( \{i\} \) and the priority value, say \( \{p\} \) both moves up to its parent node \( P_j \). The node \( P_i \) inserts \( \{i\} \) in \( \text{PL}_{j} \) and \( \{p\} \) in \( \text{PRL}_{j} \), before sending a fresh request \( \{j\} \) along with the priority value \( \{p\} \) to its own parent. The relaying of token request from originating node \( P_i \) may move right up to the root node \( \text{P}_{\text{hold}} \). The process, may however, stop at any intermediate node \( P_j \), if there is already a pending request for \( P_j \) whose priority is higher than that of \( P_i \). Thus, the maximum number of request control messages for each token request from \( P_j \) could be \( L \), where \( L \) is the level of node \( P_j \) in the tree. Therefore, maximum concurrency of our algorithm is \( L \).

**VIII. CONCLUSION**

We have presented a fair mutual exclusion algorithm for distributed systems at large and mobile environment in particular in the proposed FAPP algorithm that uses asympotonic massaging passing. Fairness is defined in terms of satisfying requests for critical section access maintaining the order of arrival of requests for tokens in the root node of the inverted tree structure that forms the logical topology. The FAPP algorithm in comparison to the Raymond’s algorithm uses more control messages. But this is how it ensures the fairness amongst the priority processes competing for the token. The FAPP fetches the identity of the requesting node right up to the root or \( \text{P}_{\text{hold}} \) node. This requires a slightly higher number of control messages per token request. The proposed FAPP algorithm passes the token to the requesting process with highest priority. Besides, it also maintains the FCFS order in allocating token amongst equal priority jobs. While the proposed algorithm exchanges minimal information to select the next process that would enter critical sections, the overhead due to information that are maintained in the nodes is also very little. The most significant contribution of FAPP in comparison with MRA-P[1] is that FAPP maintains liveness for lowest priority process although higher priority processes are preferred by the algorithm.

**REFERENCES**


Sukhendu Kanrar is a faculty member in the Department of Computer Science, Narasinha Dutt College, Howrah. He has completed his MCA in 2004. He is also pursuing M. Tech. in Computer Science from the West Bengal University of Technology. His research interest is primarily in the area of design of Operating Systems for distributed environment.

Nabendu Chaki is a faculty member in the Department of Computer Science & Engineering, University of Calcutta, Kolkata, India. He received his Ph.D. in 2000 from Jadavpur University, India. Dr. Chaki has published more than 50 referred research papers and a couple of text books. His areas of research interests include distributed computing and software engineering. Dr. Chaki has also served as a Research Assistant Professor in the Ph.D. program in Software Engineering in U.S. Naval Postgraduate School, Monterey, CA. He is a visiting faculty member for many Universities including the University of Ca’ Foscari, Venice, Italy. Besides being in the editorial board for 4 Journals, Dr. Chaki has also served in the committees of several international conferences.
LBLS: A Locality Bounded Hashing-Based Location Service

Ruonan Rao, Shuying Liang and Jinyuan You
Department of Computer Science and Engineering
Shanghai Jiao Tong University Shanghai, China 200030
Email: {rao-ruonan,liangsy,you-jy}@cs.sjtu.edu.cn

Abstract—Geographic-based routing allows routing in mobile ad hoc networks (MANETs) yet avoiding the overhead for maintaining the topology changes in MANETs. A critical challenge in geographic routing protocols is the design of a scalable distributed location services that tracks the locations of mobile nodes in MANETs. Although a number of location services have been proposed, in typical works, the performance is not satisfactory when it comes to locality problem, which introduce high overhead in update and query operations, especially in a location service without hierarchy structure, the location information stored can potentially be far away from both the source and destination nodes, even when the source and destination nodes are close. In this paper, we present a new location service, named LBLS (Locality Bounded Location Service) to solve the locality problem with the comparable least communication and storage cost.

LBLS uses a double index hash function to map a node to a location in the network area called the virtual coordination of that node. Then, a novel method employed to divide the physical space into lattices. The publish and query algorithms are designed based on this division. In LBLS, when the distance between the source and destination nodes is $l$, the cost of query is $O(l^2)$. We define this property as $n^2$-locality bounded. LBLS is the location service that achieves this property with the least storage and network overhead. Both the analysis and experiment results are present in this paper concerned with the cost, the locality bounded property and the scalability.

I. INTRODUCTION

A mobile ad hoc network (MANET) is a network formed by a collection of mobile wireless nodes without any pervously deployed infrastructure. Due to the lack of infrastructure supporting, each node in MANET should act not only as a end system but also as a router at the same time.

A fundamental challenge in MANETs research is the design and implementation of scalable and robust routing protocols. The current routing protocols can be roughly divided into two categories: Topology-based routing and Geographic-based routing. The former one is based on the knowledge of the whole network’s topological information. The latter one is based on the knowledge of each node’s position information. In these protocols, each node acquires its position via GPS or other GPS-free positioning technologies. When the location information is available for other nodes in MANET, a source node can effectively deliver a packet to the destination either by greedy forwarding or other methods. Intuitively, compared to topological-based routing, geographic-based routing incurs less communication overhead. However, geographic-based routing also faces two challenges:

1) How to deliver a packet to destination when the position of destination node is known? This is called as the forwarding strategy.
2) How to let the source node know the position of the destination node? This is called as the location service.

The first one is almost solved by several proposed algorithms, especially GPSR [1] [2]. Whereas, there are some problems remaining in the second one, though many location service algorithms have been proposed. One of them is called locality problem, i.e. where the corresponding location information is stored can potentially be far away from both the source and destination nodes, even when the source and destination nodes are close, as a result it causes high overhead on update and query operations. This problem is more serious in a location service without any hierarchy architecture. We call such a location service a flat location service. Compared to hierarchy approach, a flat location service avoids the complexity for maintaining the hierarchy structure, but it introduces the locality problem. In this paper, we try to solve this problem. A location service algorithm that possess the property called locality bounded is proposed. A location service algorithm is locality bounded if the cost of query the position of the destination node is bounded by a function of the distance between the source and destination nodes.

LBLS is a flat hash based location service, which means that LBLS uses hash functions to determine where to store the location information of nodes, and there is no hierarchy in LBLS. A hash based approach uses a hash function or hash functions to map a node to a location or a region in the network area, then store the corresponding location information near or in such location or region. LBLS is designed to cooperate with GPSR to supporting geographic based routing. With a novel method to divide the geographic area into lattices, LBLS guarantees that when the distance between the source and destination nodes is $l$, the query takes at most $O(l^2)$ to finish. LBLS...
is the best one we have known that takes the least cost to achieve such a property. We give the proofs on this property in this paper. Then, simulation experiments are used to verify LBLS to demonstrate its.

This paper has 7 sections. It starts with an overview of related works in section II. In section III, we state the locality problem of location services in MANET. In section IV and section V, the details of LBLS and some mathematical analysis are given. The experiments results are present in section VI. Section VII is a short conclusion with discussion on future works.

II. RELATED WORKS

Many location service algorithms have been proposed. Surveys on some of the algorithms can be found in [3] [4] [5] and [6], [7]. Not all of them are hash based. Following are some algorithms that are related to this work.

GLS [5] is a location service with a hierarchy of square regions. A node belongs to only one square in each order in the hierarchy. It stores its location information on 3 nodes in each square containing it. If two nodes are close enough, they will be in the same square area with a low cost and need not travel a long distance to exchange the location information. However, the cost of maintain such a hierarchy structure is expensive. The work in [8] shows some similar results. In addition, GLS has not been proved that it has an upper bound of query cost.

GLS protocol proposed in [10] is a simple flat hash based location service. In the paper, the authors have mentioned the locality issues, and try to solve this problem by a method called \( \alpha \)-scaled region—the hash function only maps a node to a location only in a region called scaled location server region which is similar to the whole area and located in the center. The radio of the side length of the scaled location server region and that of the whole area is defined as the scaling factor \( \alpha \). Intuitively, this approach can reduce the cost of query in some situations, especially when the source and destination nodes are all near the center. However, it also may cause the cost of query increases. GHTLS do not possess the locality bounded property, either.

GHT [11] which is designed for data-centric storage in sensornets. One can consider location information as a specific data in sensornets, and augment that GHT also can be used as a location service. Although, GHTLS and our work share some characteristics with GHT, GHT cannot be effectively used as a location service in MANET. The design objectives in GHT are fundamentally different from location services. An analysis on GHT in MANET can be found in [12].

III. LOCALITY PROBLEM IN FLAT HASHING BASED LOCATION SERVICES

A location service is a service that provides the corresponding location information according the node's unique identifier. Its main functionality is supporting geographic based routing, when it also can be used to support location related applications. In geographic-based routing, when a source node wants to send a packet to a destination node. It must first acquire the location information of the destination node through the location service.

In a typical flat hash based location service, a hash function is used to map each nodes unique identifier to a home region. All nodes in the home region maintain the location information for the node and are responsible for replying the queries for that node. These nodes act as location servers. However, the cost of underlying routing will surprisingly high if the source and destination nodes are close, but the home region is far away. Intuitively, in such situations, the cost of geographic based routing may be worse than topological based routing, because generally, the cost of topological based routing is bounded. Thus, such a locality problem is important for the success of geographic routing. To formally evaluate this problem, following definition is given.

Definition 1 (Locality Bounded): A location service is called locality bounded, if the distance between the source node \( S \) and the destination node \( D \) is \( l \), and the cost of query the location of \( D \) by \( S \) is at most \( f(l) \). We call such a location service is \( f \)-Locality Bounded.

In the following sections, LBLS algorithm a location service possessing the \( l^2 \)-Locality Bounded is introduced. We give the proof on this property.

IV. LBLS:A HASH TABLE LOCALIZED

In this section, we present the details of LBLS. LBLS is build on the top of GPSR, and it cooperates with GPSR to provide geographic based routing. We first study some properties of GPSR, then some concepts of LBLS are introduced. A novel method that LBLS uses to divide the physical space is present, and then the publish and query algorithms are given.

A. GPSR

GPSR [1] [2] is a geographical based routing. All nodes know their own location and their neighbors locations. All packets contain the location information about their destinations. GPSR defines two distinct modes of routing: the greedy forwarding mode that moves packets progressively closer to the destination at each hop, and the perimeter forwarding mode that forwards packets where greedy forwarding fails.

The greedy forwarding rule is simple: a node \( x \) forwards a packet to its neighbor \( y \) that is closest to the destination \( D \), provided that \( y \) is closer to \( D \) than \( x \). When such a \( y \) does not exist, the greedy forwarding mode fails. Then GPSR enters perimeter mode, which forwards packets using the right-hand rule, which means the packet
B. Map to Physical Space

As stated before, a flat hash based location service will map a node’s identifier to a region named home region using a hash function. LBLS uses the same approach with a little modification. The hash function used in LBLS is a double index hash function \( H(N.id) = (h1(N.id), h2(N.id)) \) that maps node \( N \)’s identifier \( N.id \) to a coordination \((h1(N.id), h2(N.id))\) inside the area of the network. We call the associated coordination \((h1(N.id), h2(N.id))\) of node \( N \)’s virtual coordination. The definition is given in definition 2.

Definition 2 (Virtual coordination): A Node \( N \) with unique identifier \( N.id \), its virtual coordination is \( H(N.id) = (h1(N.id), h2(N.id)) \), where \( h1, h2 \) is hash functions.

The major difference among typical flat hash based location services and LBLS is that LBLS pushes the concept of home region to the extreme: the home region becomes a single point in this area.

C. Divide the Physical Space

With node \( N \)’s identifier \( N.id \), one can find the virtual coordination \( H(N.id) \). We define a parameter \( d \) named lattice length. After finding the virtual coordination, LBLS divides the physical space into lattices using the virtual coordination as the original point according to the following rules:

- Make circles \( C_i \) with radius \( r_i = d * i, i = 0, 1, \ldots \) with the same center \( H(N.id) \). We call these circles lattice circles.
- For a circle \( C_i \), \((i = 1, 2, \ldots)\), let \( j = \lfloor \log_2 i \rfloor \), then divide \( C_i \) into \( 2^{j+2} \) with equal angle, starting from the \( x \) axis. We denote the lines used to divide the circle as \( l_{m}^{2^{j+2}} \), \( m = 0, 1, \ldots, 2^j - 1 \), and call them lattice lines. These lines intersect the circles \( C_k \), \( k = 2^j, 2^j + 1, \ldots, i \) with points \( P_{m}^{2^{j+2}} \), which are named lattice points.
- For each 4 points, \( P_{m}^{2^{j+2}}, P_{m+1}^{2^{j+2}}, P_{m+2}^{2^{j+2}}, P_{m+3}^{2^{j+2}} \), \((m = 0, 1, \ldots, 2^j - 1, k = 2^j, 2^j + 1, \ldots, i - 1)\), form a lattice corner, dedicated by \( LC_{m}^{2^{j+2}} \).
- The area enclosed by lattice corner \( LC_{m}^{2^{j+2}} \) and the corresponding line and circle is called a lattice, denoted as \( L_{m}^{2^{j+2}} \).

Lattice is the most important concept in LBLS. Figure 1 gives three examples of lattice in LBLS. Figures 1(a), 1(b), 1(c) give the lattices \( L_{0}^{2}, L_{1}^{2}, L_{3}^{2} \), when the largest circles are \( C_1 \), \( C_2 \), \( C_4 \). The corresponding lattice corners are dedicated by ▲.

D. Location Servers Selection and Location Information Update

Unlike other typical flat hash based, LBLS stores the location information not only in the home region, but also stores on the way to the home region. For a node \( N \), assuming its current location is \( N.loc \), and its virtual coordination is \( H(N.id) \). Obviously, a short path that travels along the lattice lines exits between \( N.loc \) and \( H(N.id) \). The path can be found based on following steps:

1) First, determine the lattice \( L(N.loc) \) where \( N \) is based on \( N \)’s current location \( N.loc \):
   a) Calculate the distance \( \text{dist} \) between \( N.loc \) and \( H(N.id) \). Let \( k = \lfloor \text{dist}/d \rfloor \) and \( j = \lfloor \log_2 i \rfloor \).
   b) Calculate the angle \( \alpha \) from vector \( (1, 0) \) to vector \( (N.loc, H(N.id)) \) in anticlockwise direction(Figure 2 gives two examples.). Let \( m = \lfloor \frac{\alpha}{2\pi} \rfloor \).
   c) Then node \( N \) is located lattice \( L(k)^m \). Two corresponding lattice lines are \( L_{m}^{2^{j+2}} \cdot L_{m+1}^{2^{j+2}} \).

2) One of the lattice lines \( L_{m}^{2^{j+2}} \cdot L_{m+1}^{2^{j+2}} \) is closer to \( N \) than another. The location information will be sent along this line.

3) When the path comes to the circle \( C_{2^j-1} \), the lattice line chosen will terminate. However, it may not arrive the virtual coordination \( H(N.id) \). Then a new lattice line should be chosen. If the current lattice line is \( L_{m}^{2^{j+2}} \), the next lattice line to travel is chosen based on the following rule:
   - If \( m \) is an even number, the next lattice line is \( L_{m/2}^{2^{j-1}} \).
   - If \( m \) is an odd number, the next lattice line is \( L_{(m-1)/2}^{2^{j-1}} \).

This rule not only can be applied when the first lattice line is terminated, but also suitable for the next lattice line and next’s next lattice line.

The shortest path to the virtual coordination is found. Where the location information will be stored? In each

\(^1\)When \( i = 0 \), the circle becomes the point \( H(N.id) \).
\(^2\)Here, \( m+1 \) means \( m + 1 \mod 2^{j+2} \). In this paper, we use \( m \) to denote such situations when it is not ambiguous.
\(^3\)If the distances from \( N \) to the two lines are equal, \( L_{m}^{2^{j+2}} \) is chosen.
lattice line along the path found, there are several lattice points. These points are called server coordinations. The location servers are the nodes nearest to these server coordinations. The location information of $N$ will be stored on these nodes.

Figure 3 gives an example of the way LBLS used to disseminate the location information of node $N$. As shown in figure 3, node $N$ is located in lattice $L_{29}^2$, which is enclosed by lattice circle $C_6$ and $C_7$, lattice line $L_{29}^{32}$, $L_{29}^{30}$, $L_{29}^{20}$ is closer to $N$ than $L_{29}^{30}$, and then $L_{29}^{20}$ is chosen. When the packet arrives to the lattice point $P_{29}^3$, which terminates the lattice line $L_{29}^{32}$. A new lattice line should be chosen. As the rules present above, the next lattice line should be $L_{14}^{14}$. $L_{14}^{14}$ will terminate in lattice point $P_{14}^4$, and the next line is $L_2^2$. This procedure will continue till the packet is delivered to the location $H(N.id)$. The server coordinations in this example are $P_{29}^6$, $P_{29}^5$, $P_{29}^4$, $P_{14}^{14}$, $P_2^8$, $P_2^6(H(N.id))$.

Then another problem arises—how to deliver the packet to the nodes that are nearest to the server coordinations? We use GPSR routing to solve this problem. A packet that used to disseminate the location information will indicate where its destination is (the server coordination), rather than indicate which node its destination is. Thus, in a traditional network, such a packet will not be consumed by any node. GPSR will not let any node consume this packet either. Fortunately, some properties of GPSR on such a situation, where the packet does not indicate the destination node, help to decide when and who will consume the packet for location disseminating. These properties are given by propositions 1 and 2. The proofs given here are simplified. The details of the proofs on these propositions can be found in [13].

**Proposition 1**: Routing terminates in GPSR’s perimeter mode in LBLS.

**Proof**: Assuming the routing does not terminates in perimeter mode, then it must terminate in greedy mode or dose not terminate forever. Obviously, routing will terminate when all the possible paths are traveled, and all the possible paths is at most $n^2$. Then, the routing terminates in greed mode. When a packet is in greed mode, it means that a neighbor node that closer to the destination exits. However, if such a node exits, why the routing terminates? Obviously, it leads to contradiction.

**Proposition 2**: The location server is the node where the packet enters the last perimeter mode.

**Proof**: Assuming that the nodes entering the perimeter mode are $N_1,N_2,\ldots,N_i$, orderly. We proof proposition 2 in the following two situations:

1) $n = 1$. Obviously, $N_1$ is the only node, and no other nodes will be closer to destination than $N_1$ in the routing path. Otherwise, a path can deliver the packet closer to the destination exits. As the result of proposition 1 states, then the perimeter mode $N_1$ enters is not the last one, contradictorily.

2) $n > 1$. If $N_1$ is not the location server, then a node that is closer to destination than $N_1$ exists in the routing path. There are two situations:

   a) This node is not in $N_1,N_2,\ldots,N_i$, As the proof on the case where $n = 1$, it leads contradiction.

   b) This node is $N_k$, and $k \in 2,\ldots,i$. In such situation, GPSR will enter Greedy mode on node $N_{k-1}$. Also lead to contradiction.

Based on the two propositions above, a location server can be easily found during the routing process. The next question to answer is when should a node publish its location information into LBLS, and how?

When Node $N$ moves from location $x$ to location $y$, it performs the publish as following:

- If $LN(x) = LN(y)$, new location information is published to the same servers.
- Else, node $N$ first issues a packet to inform the servers that store the old location information to erase the information. Then it publishes the new location information.

Another parameter $d_n$ is defined to indicate the distance a node can move before it do a update. Generally speaking, when a node move from one lattice to another, it must perform an update. As a result, $d_n$ should not be larger than $d/2$, where $d$ is the lattice parameter defined above.

**E. Perform Query**

When node $S$ wants to know where $D$ is, it performs a query operation. The query operation performs in tow modes: extension mode and non-extension mode. In extension mode, the query will goes to the servers located
on the adjacent lattice lines, when in non-extension mode, the query is limited to current lattice lines. The query process try to cover the area that may intersect with the path where \( D \) puts its location information as soon as possible. The details of the query process is described bellow.

1) In step 1, \( S \) calculates which lattice of \( D \) it is in using the method given in previous section. We denote this lattice as \( L_{k}^{m} \). Node \( S \) first query the location servers whose location server coordination are \( P_{k}^{m} \) and \( P_{k}^{m+1} \) in extension mode (\( P_{k}^{m} \) left-extension and \( P_{k}^{m+1} \) right-extension).

2) In step \( i \) (\( i = 1, 2, 3, \ldots, k - 1 \)), location servers \( P_{k-1}^{m'} \) will get the query. It process the query based on the following situations:

- It knows where \( D \) is, then it answer this query, and informs other location server in the same iterative level to terminate this query.
- It dose not know the answer.
  - The query is in non-extension mode.
    * \( k - i + 1 \) = 2\(^{j}\), no future action is needed.
    * \( k - i + 1 \) = 2\(^{j}\) and \( m' \) is an odd number. Deliver a query to \( P_{k-1}^{m'-1} \) in left-extension mode.
    * \( k - i + 1 \) = 2\(^{j}\) and \( m' \) is an even number. Deliver a query to \( P_{k-1}^{m'-1} \) in left-extension mode and a query to \( P_{k-1}^{m'} \) in non-extension mode.
  - The query is in extension mode. We only discuss the left-extension case. The right-extension case is similar.
    * \( k - i + 1 \) = 2\(^{j}\) and \( m' \) is an odd number. Deliver a query to \( P_{k-1}^{m'-1} \) in left-extension mode.
    * \( k - i + 1 \) = 2\(^{j}\) and \( m' \) is an even number. Deliver a query to \( P_{k-1}^{m'-1} \) in left-extension mode and a query to \( P_{k-1}^{m'} \) in non-extension mode.

3) In step \( k \), the query arrives \( H(D.id) \). If the location server knows where \( D \) is, then it reply the answer, otherwise it will tell \( S \) the location of \( D \) is not known.

Figure 4 gives an example of query in LBLS. Node \( S \) wants to know where \( D \) is. It calculates the lattice of \( D \) where it is in. In the example, \( S \) is in \( L_{6}^{19} \), it deliver queries to \( P_{19}^{19} \) in left-extension mode and to \( P_{6}^{20} \) in right-extension mode. We trace the thick line in figure 4 to explain the query process in detail. In the first iterative step, \( P_{6}^{19} \), 6 is not the power of 2. \( P_{5}^{18} \) (left extension mode), \( P_{19}^{19} \) (non-extension mode) will get the query. Then, \( P_{4}^{17} \) (left extension mode), \( P_{2}^{18} \) (non-extension mode), \( P_{4}^{19} \) (non-extension mode). Here, \( 4 = 2^{2} \), and 17 is an odd number. Then, \( P_{4}^{17} \) deliver query to \( P_{5}^{19} \). The process will go on until a server replies or the query arrives \( H(D.id) \).

---

4We use the location server coordination to denote location server, when there is no ambigulessness.
As a result, the following criterions can naturally obtained:

\[
\frac{3\pi d^2}{8} > Dens\]
\[
d\sqrt{\frac{\pi^2}{4} + 1} < 5D_t
\]

B. On the cost of LBLS

In this section, we study the cost of LBLS, which includes the publish cost, the query cost, and storage cost. In all cases, we assume the number of nodes is \( n \), the nodes density is \( Dens \), and the lattice length is \( d \). In addition, \( r \) hops a packet needs to travel from a location server to the next adjacent one.

1) Publish Cost: The publish cost is the number of packets and the distance these packets will travel to publish the location information of each node in the networks into LBLS. Obviously, the publish cost is proportional to the distance between node \( N \)’ location and its virtual coordination. When the number of nodes is known, the longest distance between a node and its virtual coordination is \( O(\sqrt{N}) \), providing the density of nodes is a constant. Therefore, the upper bound of the publish cost is \( O(\sqrt{N}) \).

2) Query Cost and Locality Bounded Property: As stated in the beginning of this paper, the most remarkable feature of LBLS is LBLS is \( l^2 \) locality bounded. We give the proof here.

Theorem 1: LBLS is \( l^2 \) locality bounded.

To proof theorem 1, we first give two lemmas, stated in Lemma 1 and Lemma 2.

Lemma 1: In the \( i \)th step of query, query packets arrive the lattices that are \( i - 1 \) lattices far away for the lattice where the source node is located.

Lemma 2: The cost of query after \( i \)th step finishes is \( O(i^2) \).

These two lemma can easily be proofed by inducing on \( i \). The details of the proofs can be found in [12] and are omitted here.

Finally, we can proof Theorem 1.

Proof: We assume the source node which query the location of another node \( D \) is \( S \), and is located in lattice \( L^m_k \). The location information is found in the server whose server coordination is \( L^p_{k-i+1} \). There are two possible cases based on the mode the query arrives \( P^p_{k+i} \):

1) In non extension mode. From the description of the publish and query algorithms, obviously, we have that the \( D \) is located in \( L^m_{k+i} \) or \( L^m_{k+i+1} \). Because \( P^p_{k+i+1} \) is on the publish path of node \( D \), but \( P^p_{k+i+2} \) is not (otherwise, we will get the location information in \( P^p_{k+i+2} \). Thus, the \( P^p_{k+i+1} \) is the beginning of the publish path, and \( D \) is in \( L^p_{k+i+1} \) or \( L^p_{k+i+2} \). In this case, the distance between \( D \) and \( S \) is at least \( (i-1) \times d \), and the cost is \( O(i^2) \). It means LBLS is \( l^2 \)-locality bounded in such case.

2) In extension mode. In such case, the query path intersects the publish path of \( D \) in the first time.

In GLS the height of grid is set to \( h \), and in LLS the unit length is \( u \). \( N \) is the number of nodes in the network.

Trace on the possible publish path (see figure 3 as a reference), the shortest distance between \( S \) and \( D \) appears when \( S \) and \( D \) is between the same lattice circles, and the shortest distance is also about \( (i-1) \times d \). As stated in lemma 2, the cost is also \( O(i^2) \). Then, we can conclude that in the case where the corresponding location information is found in extension mode, the LBLS is also \( l^2 \)-locality bounded.

Finally, LBLS is \( l^2 \)-locality bounded either the query finishes in non-extension mode or in extension mode.

3) Storage Cost: The storage cost is that for each nodes, how many nodes will act as its location servers, i.e. its location information will be distributed to how many nodes. Obviously, the cost of storage is similar with the cost of publish, which is also with an upper bound \( O(\sqrt{N}) \).

4) Location Servers Maintain Cost: The cost of servers maintaining is the overhead used for migrate the location information a node currently holding to its new location server. Obviously, the frequency the maintaining activities occur depends on the speed nodes moves. However, when such situation happens, only very few packets are needed (at most time it should be \( O(1) \)), and the activity is limited to a small area (within the scope of one hop).

5) A brief Comparison: Table I gives a brief comparison among LBLS and other location service related in section II. For detailed analysis, please refer [12], [10] and [13].

From the table, we can conclude that only LLS and LBLS bear the property of locality bounded. Yet compared to LLS, LBLS is with less network and storage overhead.

VI. EXPERIMENTS

In this section, we present the experiments results on LBLS. The experiments are concerned with the performance of LBLS in static network, and in mobile network and with the issue of scalability.

A. Experiment Method and Metrics

We have implemented LBLS in ns2 [14]. We first port the GPSR codes [15] from ns2.26 to current version 2.28. Then we modify some GPSR code and implement LBLS above it. In all experiments, we use a 802.11 radio model.
TABLE II. Parameter values of GPSR used in our experiments

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPSR Beacon Interval</td>
<td>1s</td>
</tr>
<tr>
<td>GPSR Beacon Expiration</td>
<td>4.5</td>
</tr>
<tr>
<td>use_peri</td>
<td>1</td>
</tr>
<tr>
<td>use_planar_</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE III. Metrics used for evaluating LBLS

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_in</td>
<td>The number of packets used to publish location information. We call such packets PUT packets.</td>
</tr>
<tr>
<td>P_r</td>
<td>The average hops a PUT packet travels.</td>
</tr>
<tr>
<td>R_n</td>
<td>The number of packets used to erase location information. We call such packets REMOVE packets.</td>
</tr>
<tr>
<td>R_p</td>
<td>The average hops a REMOVE packet travels.</td>
</tr>
<tr>
<td>U_n</td>
<td>The number of packets used to migrate location information to another node. We call such packets UPDATE packets.</td>
</tr>
<tr>
<td>U_r</td>
<td>The average hops a UPDATE packet travels.</td>
</tr>
<tr>
<td>LN_{max}</td>
<td>The maximum number of location information entries a node stored.</td>
</tr>
<tr>
<td>LN_{ave}</td>
<td>The average number of location information entries a node stored.</td>
</tr>
<tr>
<td>RQS</td>
<td>The ratio of query success.</td>
</tr>
</tbody>
</table>

The following metrics are used to evaluate LBLS: (1) LBSLs overhead - The number of location service (LS) protocol packets transmitted, and the average hops packets travels. (2) The cost of storing the location information. (3) Query success ratio (QSR) – The ratio of the number of query replies received at the sources to the number of location queries initiated by the sources. The discrete metrics are listed in table III. The experiments on static and mobile networks run 300s, and the experiments on scalability run 150s.

All experiments are performed as follows:
- In the first a few second (30s), all nodes publish their location information into LBLS.
- Then, each node chooses a destination randomly and independently, and queries its location.

Concerned with static networks, the topology scenarios are generated using the following model:
1) Uniform and Random Network. If there are N nodes in the network, then we divide the whole area of the network into N parts with equal area. Each node is placed randomly in each divided part.

In each topology, we vary the node densities and the value of lattice parameter d. The node densities are varied among 5625m²/node, 10000m²/node, and 22500m²/node. d are varied among 100m, 150m, 200m, 250m, and 300m.

Concerned with mobile networks, we use random way point mobility model (RWP) for evaluating LBLS in an entity mobility model and reference point group mobility model (RPGM) for evaluating LBLS in a group mobility model. We use IMPORTANT [16] to generate the mobile scenarios. The values of parameters used in RWP and RPGM are listed in table IV. The number of nodes is 100, the density of nodes is fixed to 10000m²/node, the lattice length d is set to 300m, and d_m is set to 150m. In RWP we vary the speed. In RPGM we only observe the performance on a scenario preset.

Concerned with the scalability issue, we study the performance of LBLS in both static and mobile network by varying the number of nodes. The number of nodes are varying among 36, 64, 100, 144, 256. In static network, the uniform and random topology is chosen, when in mobile network, the RWP mobility model is chosen.

B. Results on Static Network

Table V gives the results of experiments on static networks. In static networks, no update or location server maintaining activity occurs. Based on the results show above, we can verify the analytical results in the previous section. For example, in the case where the density of nodes is 10000m² and d is 250m, the distance between a node N and its virtual coordination H(N.id) is among [0, 1000sqrt(2)]. If the hash function distributes uniformity, then the average distance is about 700m. The the average store cost should be 700/250 ≈ 3, and the experiment result is 3.3. A little higher than theory result, but in a reasonable range. Concerned with the locality bounded property. In the random destination chosen model we used, the distance between the source and destination nodes is also approximately 700m. Without locality, the cost for query should approximately equal to the cost of publish. In the result, the QSR is smaller than P_r, which means with the locality, the distance a query packet travels decreases.

C. Results on Mobile Network

1) Results on RWP: Figures 5(a), 5(c), 5(b), 5(d), 5(e), 5(f), 5(g), and 5(h) give the results of experiments on RWP.

The results obtained are:
- The QSR decreases when the speed increases. This is also a problem in all current available location services as shown in [10] and [8]. In mobility situations, this occurs when the destination node is reachable, but the links to location servers are
The performance in RPGM is better than RWP in the case with comparable parameters, expect that the $P_r$ is higher in RPGM than that in RWP. The better performance is due to consistent of nodes movement in RPGM. The higher $P_r$ is because in a group mobility mode, all nodes may be concentrated in a small area, where some virtual coordination are outside. Then a long routing path may be needed to find the corresponding location server.

2) Results on RPGM: The experiment results on RPGM are given in table VI.

The results of the experiments on the issue of scalability of LBLS are given in figure 6 and figure 7. Note that not all metrics defined in section VI-A are used to evaluate the scalability of LBLS. It is not only because the space limitation, but also because other metrics are not related to the issue of scalability.

From the results, we can safely conclude that LBLS show a good scalability in both static and mobile network. All metrics are increase almost linear with the scale of the network. Although a linear result is not a perfect scalable algorithm, compared to similar approach, LBLS demonstrates a better scalability over others. Please see the results in [10] as a reference.

D. Results on Scalability

The results of the experiments on the issue of scalability of LBLS are given in figure 6 and figure 7. Note that not all metrics defined in section VI-A are used to evaluate the scalability of LBLS. It is not only because the space limitation, but also because other metrics are not related to the issue of scalability.

From the results, we can safely conclude that LBLS show a good scalability in both static and mobile network. All metrics are increase almost linear with the scale of the network.

VII. Conclusion

In this paper we present a new location service named LBLS, which possesses the property of locality bounded. The most remarkable of LBLS is that LBLS is the best one of so many location services proposed incorporating locality with minimal cost introduced. LBLS employees a novel method to divide the physical space into lattices. The publish and query algorithms are based on this
division. LBLS is $l^2$ locality bounded, which means the cost of query is at most $l^2$, when the distance between the source and destination node is $l$. We give the analytical and simulation results in details in this paper.

Although LBLS gives promising results both on theoretical analysis and simulative experiments. Some issues are worth future studying.

The first one is on the low bound of the locality bounded property. In both LLS and LBLS, the result is $O(l^2)$. We doubt it is the best answer. If it is not, then what is the low bound of locality a location service can achieve? Intuitively, $O(l^2)$ may be the final answer. Think about a node know the destination is at most $l$ away, but it does not know the exact location, then it should reach all possible locations, which will result in $O(l^2)$.

The second one is about the LBLS itself. Compared to other similar approaches, LBLS needs more computing capability. In LBLS, the types of computing include computing on trigonometric function, manipulation on vectors and so on, all of which do not appear in other location services. Although the computing power of mobile devices are continuing increasing, reduce the complexity of LBLS is still worth future investigating.

**REFERENCES**


Ruonan Rao is currently an associate professor at department of computer science and engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include distributed computing and grid computing.

Shuying Liang received her ME degree in software engineering from Shanghai Jiao Tong University, Shanghai, China, in 2009. She is currently a phd student in electrical engineering and computer science in Washington State University, Pullman, USA. Her current research interests include distributed computing and pervasive computing.
LDB: Localization with Directional Beacons for Sparse 3D Underwater Acoustic Sensor Networks

Hanjiang Luo\textsuperscript{1}, Zhongwen Guo\textsuperscript{1}, Wei Dong\textsuperscript{2}, Feng Hong\textsuperscript{1}, Yiyang Zhao\textsuperscript{3},
\textsuperscript{1}Ocean University of China
\textsuperscript{2}Zhejiang University
\textsuperscript{3}Hong Kong University of Science and Technology
\{luohj, guozhw, hongfeng\}@ouc.edu.cn, dongw@zju.edu.cn, \{zhaoyy\}@cse.ust.hk

Abstract—In this paper, we propose a novel distributed localization scheme LDB, a 3D localization scheme with directional beacons for Underwater Acoustic Sensor Networks (UWA-SNs). LDB localizes sensor nodes using an Autonomous Underwater Vehicle (AUV) as a mobile beacon sender. Mounted with a directional transceiver which creates conical shaped directional acoustic beam, the AUV patrols over the 3D deployment volume with predefined trajectory sending beacons with constant interval towards the sensor nodes. By listening two or more beacons sent from the AUV, the nodes can localize themselves silently. Through theoretical analysis, we provide the upper bound of the estimation error of the scheme. We also evaluate the scheme by simulations and the results show that our scheme can achieve a high localization accuracy, even in sparse networks.

Index Terms—Localization, Underwater Acoustic Sensor Networks (UWA-SNs), Acoustic Networks, Underwater positioning, Mobile beacon

I. INTRODUCTION

Underwater Acoustic Sensor Networks (UWA-SNs) has recently been drawing much attention because of its potential applications ranging from oceanographic data collection, environment monitoring, structure monitoring, tactical surveillance to disaster prevention, etc [1], [2]. For these applications, localization is an indispensable part of UWA-SNs. For example, energy-efficient geo-routing schemes require location information for making routing decision, aquatic monitoring and underwater surveillance application demand high-precision localization [3], [4], [5].

Though localization is widely studied for terrestrial sensor networks [6], [7], [8], existing terrestrial techniques cannot be directly applied to UWA-SNs because of the following challenges.

First, anchor nodes are difficult to deploy at precise locations in underwater environments. To achieve 3D localization, localization schemes often need to deploy anchor nodes underwater and the precision of anchor deployment significantly affects the location accuracy of nodes directly. However, it is always difficult to deploy anchor nodes precisely at the sea floor, especially for deep ocean [9].

Second, underwater acoustic communication channel has unique characteristics, such as limited bandwidth capacity and limited battery power. The current available limit of bandwidth underwater is roughly 40km.kbps [10], which means that long-range communication over several kilometers only has a few kHz bandwidth, and only short-range communication over several tens of meters may have more than a hundred kHz of bandwidth [11]. Thus the available bandwidth is severely limited underwater which renders those localization schemes based on frequent message exchanging not appealing [12]. The underwater sensors are battery-powered and it is even harder to recharge or replace node batteries in underwater environments. Meanwhile, the energy consumption of acoustic modem in UWA-SNs is quiet different from those of typical radio transceivers, for the transmit power is often 100 times expensive than that of the receive mode [13], [14]. For example, the typical receive power of the WHOI micro-modem is about 80mW, but the transmit power is 10W [15]. Thus, silent localization, which features those schemes that un-localized sensor nodes can localize themselves just by passively listening beacons sent by anchors, has advantages in UWA-SNs [16].

Third, distance measurements underwater suffer from large errors, which makes most range-based localization schemes inaccurate. Range-based localization schemes often use Time of Arrival (TOA), Time Difference of Arrival (TDOA), Angle of Arrival (AOA), or Received Signal Strength Indicator (RSSI) to estimate their distances to other nodes [17], [18]. TOA or TDOA require precise time synchronization. However, precise time synchronization is hard to achieve in underwater environments [19]. Even if those schemes do not rely on time synchronization, they are vulnerable to the speed of sound which is affected...
underwater acoustic channel and may provide ambiguous results in underwater environments [21].

Fourth, due to economic considerations, UWA-SNs are typically sparse which makes most range-free localization schemes inaccurate. Most of the 3D localization schemes assume a dense network in order to achieve satisfied localization results. However, in underwater networks, node deployment is generally sparse [10].

In order to address the challenges listed above, we propose LDB, a 3D localization scheme using directional beacons for UWA-SNs. LDB has the following characteristics and advantages.

First, it is anchor-free by leveraging a moving AUV (autonomous underwater vehicle) [22]. We mount an acoustic directional transducer which uses multiple ceramic or piezoelectric elements to create conical shaped directional acoustic beam under the AUV to aid localization underwater [23].

Second, it is energy-efficient by eliminating inter-node communications because it does not need synchronization between nodes or need send messages to anchor nodes. In LDB, all nodes passively listen to beacons sent by the AUV.

Third, it is range-free, thus does not suffer from distance measurement errors. By using cheap pressure sensors, the depth of node can be directly determined [24]. The 2D position at a pre-determined depth can be determined by passively listening to beacons sent by the moving AUV (containing the AUV’s current position).

Fourth, it is accurate, even in sparse networks. The key insight to determine a 2D position at a pre-determined depth is that nodes (2D points) right beneath the AUV’s moving line have the largest inter contact time whereas nodes far apart the moving line have smaller inter contact time. By calculating the distance that the AUV have moved during the inter contact time, we can precisely determine the position of nodes. Moreover, we have proved that the localization errors are upper-bounded.

We evaluate our scheme by extensive simulations, the results show that LDB not only has a high localization accuracy but also works well even in sparse sensor networks.

The rest of this paper is structured as follows. Section II briefly reviews some related work. In Section III we describe our localization scheme in detail. We evaluate the scheme in Section IV. In Section V we conclude our paper and report our future work.

II. RELATED WORK

Underwater acoustic localization can be broadly classified into two categories: range-based schemes and range-free schemes [9], [25]. Range-based schemes first measure or estimate distances or angles between nodes using Time of Arrival (TOA), Time Difference of Arrival (TDOA), Angle of Arrival (AOA), or Received Signal Strength Indicator (RSSI) [26]. Then the schemes apply triangulation or multilateration to compute node’s positions. Range-free schemes do not use range measurement to estimate distances between nodes, they use network topology or the position of surrounding anchor nodes to locate nodes, which can be generally classified into hopcount-based and area-based schemes [21]. The advantages of range-free schemes lie in their simplicity and the low communication costs. In this section, we briefly overview some localization schemes related to our work. A more detailed survey is in [21].

An area-based range-free underwater positioning scheme (ALS) is proposed in [27]. ALS relies on varying the power levels of anchor nodes to divide the plane into many small sub-regions. The main drawback is that the scheme needs a central sever to compute node’s position, thus it is not a fully distributed localization scheme.

UPS is a silent positioning scheme which uses the time difference of arrivals to detect range differences from four anchor nodes [16]. UPS reduces the communication costs by using silent positioning and can work well in some scenarios. However, it needs four powerful anchor nodes covering the entire un-localized area. This assumption may not always hold and as we mentioned in Section I, precisely deploying anchor nodes will be difficult.

DNR was proposed in [28], with beacons sinking and rising in the water, the nodes can localize their position by passively listening to those beacons. DNR assumes that nodes are synchronized which increase the communication costs. SLMP was proposed for large scale underwater sensor networks [29]. By utilizing the predictable mobility patterns of underwater sensor nodes, SLMP can balance the tradeoff between localization accuracy, localization coverage and communication costs. Both DNR and SLMP need a certain percentage of anchor nodes to achieve satisfied localization accuracy. Though high ratio of anchor nodes improves the accuracy of localization, it is achieved at a higher costs, for the anchor nodes may be more expensive and deploying anchor nodes is more difficult. Meanwhile, it may lead to more energy costs in communication and degrade the network throughput.

Using projection technique, the proposed USP transforms 3D localization to 2D localization [9]. USP is
especially tailored to sparse underwater sensor networks. However, their evaluation shows that the ratio of nodes localized reaches only 45% when the average node degree is 10, and when the average node degree is under 6, the ratio of nodes localized is even under 10%. In [30], an AUV was used to aid node’s localization underwater. The AUV receives GPS signals while floating and then it dives into fixed depth following a predefined trajectory. When it patrols the field of sensors, it broadcasts beacon messages. The un-localized nodes send messages to AUV when they hear the beacons, which are used to measure the round trip propagation delay. Thus the scheme relies on the speed of sound to estimate the distance. Both USP and [30] are range-based schemes, thus they suffer from distance measurement errors.

We proposed a directional acoustic beacon based localization scheme UDB which assumes that nodes are deployed in two-dimensional underwater sensor networks [31]. In this paper, we extend our prior work to three-dimensional underwater sensor networks where sensor nodes float at different depths of water. LDB works well with sparse underwater networks, even with average degree is 1, such as a linear underwater sensor networks. Meanwhile, it is energy-efficient for underwater sensors, because the node can localize itself by passively receiving even only two beacons.

III. LDB LOCALIZATION SCHEME DESIGN

In this section, we describe LDB in detail. We first describe the network model, then we propose LDB scheme and at the end of this section we present the LDB algorithm.

A. Network Model and Assumptions

We consider three dimensional underwater acoustic networks that the sensor nodes are anchored to the ocean floor and equipped with floating buoy which pushes the sensor nodes upwards. As shown in Figure 2, the nodes can not move freely with the underwater currents, because they are restricted with elastic anchor chain’s pull force and buoy’s floating force in addition to current’s force, though the underwater acoustic sensors may move with the ocean currents if they are freely floating in the water [4], [32]. Thus, the node’s movement is restricted to a very small area around a center and they are relatively stationary nodes [10].

The sensor nodes are deployed at different depth to cover a given interested area and we assume that sensor node is equipped with cheap pressure sensor to determine its depth (h).

AUVs are part of underwater sensor networks and have relatively rich power to work continually underwater [10]. Its battery may be rechargeable and it recharges its battery at sub-sea docking stations [33]. It may also be solar-powered and recharges its battery when floating at sea surface [34]. AUV can move slowly underwater following a predefined trajectory. A typical REMUS-class AUVs can move at a speed of 1.5m/s with a propulsion power consumption of 15W, and with a propulsion power consumption of 110W, it can move at a speed of 2.9m/s [35]. In [22], an AUV is used to collect sensed data from the node mounted on the water floor. An AUV was used to aid node’s localization underwater in [30]. In this paper, we also use an AUV to aid localize nodes.

As shown in Figure 1, we mount a directional acoustic transceiver under AUV to send directional beacons underwater. Though multi-path and rapid time-variation may be common in many underwater acoustic channel, the directional acoustic transceiver uses the vertical channel in our scheme which is characterized by little or no time dispersion especially in deep-water [11]. Moreover, using the same power, directional acoustic transceiver, which are commonly used between ship and unmanned underwater vehicles (UUVs), can reach far more than omni-directional transceiver.

The AUV is also equipped with cheap pressure sensor, thus it knows its depth (h_A) in water. The AUV can receive GPS signals while floating at the surface and then it dives into fixed depth [36]. When AUV is patrolling over the field of sensors at a fixed depth following a predefined trajectory underwater, it sends beacons including its 3D positions (x,y,h_A) and the directional acoustic transceiver’s beam angle [23] which we defined as the beacon angle α.

B. Proposed LDB Scheme

The depth of a node can be directly measured by a cheap pressure sensor, which is denoted as h. The key challenge to position a node’s location is thus to determine its 2D position at the fixed depth h (the node is denoted as S).

From Figure 1, we observe that when AUV sends a directional beacon towards the sensor field, those nodes who have heard the beacon actually fall in the conical beacon beam and the beam forms different circles with different h. With a fixed depth h, the center of the circle is (x,y,h), and the radius of the circle is

\[ r = \tan(\alpha/2) \times \Delta h \]  

(1)

where \(\alpha\) is the angle of the conical beacon, \(\Delta h = |h_A - h|\). Thus, when AUV wanders at fixed depth of water sending beacons, we could map the nodes from 3D to 2D with those determined circles.
For example, when node $S$ falls in the circle of depth $h$ (or the circle covers $S$), $S$ can receive beacons from the AUV, and thus can roughly estimate its position as $(x,y,h)$. This basic scheme, however, would introduce a relatively large error, because $S$ can be actually at the border of the circle. To address this issue, we make that the AUV moves in a straight line above the 3D deployment volume and it sends beacons containing its current position, its fixed depth, and the angle of the beacon with a constant time interval $t$. Hence, node $S$ can receive a series of beacons.

We first define the first-heard beacon point and the last-heard beacon point in those series of beacons. As shown in Figure 3, the AUV’s movement trajectory is the broken line. Sensor node $S$ hears the beacons when AUV sends them at point $T_1 \sim T_9$ and we define those points as mobile anchor point. AUV sends a beacon at point $T_1$, and this beacon is the first beacon which the sensor $S$ hears, we define the point $T_1$ as the first-heard beacon point. Just as the same, AUV sends a beacon at point $T_6$ and this is the last beacon which can be heard by the sensor $S$, thus we define this point as the last-heard beacon point. The sensor node need not store all those six beacons to get the last-heard beacon point. When it received the first beacon sending by AUV at $T_1$, it stored the beacon point as the first-heard beacon point. Then when it received the second beacon sending by AUV at $T_2$, it stored the beacon point as the last-heard beacon point. After it has received the third beacon at $T_3$, it renewed the last-heard beacon point as $T_3$. Similarly, when the AUV is moving away from the node and the node does not receive any beacon, the last-heard beacon point becomes $T_6$.

In the following, we will use these beacons to localize nodes. We first consider the continuous case, i.e., $t \to 0$ (Section III-B.1); then we consider the practical discrete case, i.e., $t > 0$ (Section III-B.2).

1) Continuous Case: When AUV patrols at fixed depth of water following a straight line and sends beacons continuously, the positions of the nodes can be determined by the first-heard beacon point and the last-heard beacon point.

As shown in Figure 4, the AUV moves in a straight broken line paralleled with $x$-axis. We assume that node $S$ is positioned on the top side of the AUV’s current moving trajectory. $F$ is node $S$’s the first-heard beacon point and $L$ is node $S$’s the last-heard beacon point with corresponding coordinates $(x_1,y_1), (x_2,y_1)$ respectively. The distance between $S$ and $F$ or $L$ is $r$ calculated using Equation (1). The position of $S$ is thus calculated as,

$$
\begin{align*}
  x &= \frac{x_1 + x_2}{2} \\
  y &= y_1 + \sqrt{r^2 - \left(\frac{x_1 - x_2}{2}\right)^2}
\end{align*}
$$

The node’s position can be determined by the the first-heard beacon point and the last-heard beacon point, but we assume that the AUV sends beacons continuously. However, in practice, AUV sends beacons with time intervals because the node who hear the beacons must differentiate between those beacons. In next section, we will deal with this problem.

2) Discrete Case: Because AUV sends beacons discretely with time intervals, nodes in the same small area will share the same first-heard beacon point and last-heard beacon point. As shown in Figure 5, AUV sends beacons at point $T_1 \sim T_{10}$ with the same time intervals and in the 2D plane where the sensor node $S$ resides, four circles centered at point $T_1,T_2,T_8,T_9$ form a small area. In this area, if there is more than one sensor nodes reside in it, those sensor nodes will share the same first-heard beacon point and last-heard beacon point with sensor node $S$. Thus using Equation (2) to compute node’s position will lead to large errors.

In the following, we use the small area’s centroid to compute the positions of nodes and estimate the upper bound of errors; then we discuss the AUV’s routing pattern design and 3D full-coverage problems.

a) Compute node’s position: As shown in Figure 6, we define the mobile anchor point $F’$ before the first-heard beacon point $F$ as the prior-heard beacon point of $F$ and the mobile anchor point $L’$ post the last-heard beacon point $L$ as the post-heard beacon point of $L$. The distance between two adjacent mobile anchor points which we called adjacent-beacon distance is defined as $d$. If we include the beacon interval $t$ and the AUV
and discuss how to reduce the error.

Fig. 6. Compute intersection area centroid

speed $v$ in the beacon sending by AUV, then the node can calculate $d$ as $d = t \times v$. We also define the distance between $F$ and $L$ is $D$, because we assume AUV moves in a straight line paralleled with x-axis, thus $D = |FL| = |x_2 - x_1|$. We define the distance between $F'$ and $L'$ is $D'$, thus the relation between $D$ and $D'$ is $D' = |FL'| = |FL| + 2d = D + 2d$. Four circles centered at $F', F, L, L'$ with radius $r$ form the intersection area. With different $D'$, the intersection will have different shape as shown in Figure 6(a), 6(b), 6(c). As we mentioned above, the nodes within the same intersection area share the same first-heard beacon point and last-heard beacon point.

Next we calculate the positions of nodes using the small area’s centroid according to two different cases. Figure 6(a) is under case 1 where $|FL'| < 2r$ and the un-located node’s position is determined by $F', F, L, L'$. Though Figure 6(b), 6(c) have different shape, they are under the same case 2, because node’s position is determined by two points $F$ and $L$ under the condition $|FL'| \geq 2r$. We also estimate their corresponding maximum error and discuss how to reduce the error.

Case 1: $|FL'| < 2r$

As shown in Figure 6(a), we compute $S$’s coordinate $(x, y)$ as follows:

$$
\begin{align*}
    x &= \frac{x_1 + x_2}{2} \\
y &= y_1 + \sqrt{\frac{r^2 - (\frac{x_2 - x_1}{2})^2}{2}} + \sqrt{\frac{r^2 - (\frac{x_1 - x_2}{2})^2}{2}}
\end{align*}
$$

According to Equation (3), we calculate the maximum horizontal error $E_{max-h}$ and maximum vertical error $E_{max-v}$ below.

$$
E_{max-h} = \frac{x_1 + x_2}{2} - \frac{x_1' + x_2'}{2} = \frac{x_1 - x_1'}{2}
$$

$$
E_{max-v} = \sqrt{\frac{r^2 - (\frac{x_2 - x_1}{2})^2}{2}} - \sqrt{\frac{r^2 - (\frac{x_1 - x_2}{2})^2}{2}}
$$

Case 2: $|FL'| \geq 2r$

Figure 6(b) and 6(c) have different shape because they have different $d$. However, when computing the positions of the nodes in the intersection area, they have the same characters, and we compute $S$’s coordinate $(x, y)$ as follows:

$$
\begin{align*}
x &= \frac{x_1 + x_2}{2} \\
y &= y_1 + \frac{\sqrt{r^2 - (\frac{x_2 - x_1}{2})^2}}{2}
\end{align*}
$$

According to Equation (6), we calculate the maximum horizontal error $E_{max-h}$ and maximum vertical error below.

$$
E_{max-h} = \frac{x_1 + x_2}{2} - \frac{x_1' + x_2'}{2} = \frac{x_1 - x_1'}{2}
$$

$$
E_{max-v} = \frac{d}{2}
$$
and AUV’s speed will reduce the estimation errors. The maximum vertical error is mainly determined by the radius $r$, the beacon interval $t$ and AUV’s speed $v$. If we decrease the beacon interval and AUV’s speed, we will reduce the estimation errors. The maximum vertical error is mainly determined by the radius $r$, the beacon interval $t$ and AUV’s speed $v$.

b) The impact of the restrained movement of node: In the last section, we compute the node’s position with a assumption that the node is static. However, the node may have a slightly movement under strong underwater currents. In this section, we analyze the impact of such movement.

As shown in Figure 2, restricted with elastic anchor chain’s pull force and buoy’s floating force in addition to current’s force, the node’s movement is restricted to a very small circled area around a center. We denote the radius of the circle as $u$. Compared with the circle with radius $R$ which the node resides, the circle of the node’s restrained movement is very small. However, the movement will decrease the accuracy of localization algorithm. Because the node may move randomly with the randomly underwater currents, the deviation from the center will increase the maximum vertical error or maximum horizontal error with maximum $u$.

Thus, the maximum vertical error or maximum horizontal error will be:

$$E_{max-v} = \frac{\sqrt{r^2 - \left(\frac{S - s}{2}\right)^2}}{2}$$

$$E_{max-h} = \frac{d}{2} + u$$

(8)

(9)

Summary both case 1 and case 2, we know that two cases have the same maximum horizontal error $\frac{d}{2}$ which determined by beacon interval $t$ and AUV’s speed $v$. If we decrease the beacon interval and AUV’s speed, we will reduce the estimation errors. The maximum vertical error is mainly determined by the radius $r$, the beacon interval $t$ and AUV’s speed $v$.

c) Ambiguous Problem, AUV Movement Pattern Design and 3D Full-Coverage: We have computed the position of the node using the first-heard beacon point, the last-heard beacon point and the adjacent-beacon distance. However, there is a ambiguous problem. As shown in Figure 7, when AUV wandering along the broken line, we can not determine the position of node $S_1$, because if there is a node $S_2$, then they share the same first-heard beacon point $T_1$ and last-heard beacon point $T_2$. Thus, the right equation to compute the node’s position of Equation (3) and (6) are:

$$x = \frac{x_1 + x_2}{2}$$

$$y = \frac{y_1 \pm \sqrt{r^2 - \left(\frac{S - s}{2}\right)^2}}{2}$$

(11)

(12)

By designing the AUV’s movement routing pattern, we can solve the ambiguous problem. Meanwhile, we can cover the 3D sensor deployment volume with the aid of the AUV’s routing pattern. We discuss our routing pattern below.

We assume that Figure 7 is the 2D plane to which the top surface of the 3D sensor deployment volume belongs. We assume that the distance between the AUV and the plane is $H$, then the radius $R$ of the circle in Figure 7 is computed using Equation (1) as follows:

$$R = \tan(\alpha/2) \times H$$

(13)

where $\alpha$ is the beacon angle. As shown in Figure 7, if we can cover the top surface of the 3D deployment volume, we can cover the whole sensor deployment volume fully, because the circle below the surface will be larger than those in the 2D plane. We design the AUV’s movement routing pattern as shown in Figure 7, and assume the distance between two adjacent parallel routing lines is $D_r$, then the condition to cover the whole sensor deployment volume is:

$$D_r \leq R$$

(14)

To make LDB work, another important issue to ensure is that an arbitrary node can be covered at least by two

![Fig. 7. Ambiguous problem](image1)

![Fig. 8. The condition for coverage](image2)
circles when the AUV moves along a straight line. As mentioned above, it is sufficient to ensure the coverage of the top surface. As shown in Figure 8, to ensure that an arbitrary node to be covered at least by two circles when the AUV moves along a straight line, we require that:

\[ d \leq 2\sqrt{R^2 - D_2^2} \]  \hfill (15)

If the AUV follows the designed routing pattern, then the ambiguous problem can be resolved. That is if the nodes have been localized twice, then their positions will be determined, for the nodes stay between two parallel routing lines. Another simple way is that if AUV moves paralleled with x-axis from the bottom to the top as shown in Figure 7, then the nodes located for the first time will be positioned above the routing line, thus their positions are determined and we can still use Equation (3) and (6) to calculate the node’s position. Next, we show our algorithm below.

3) Distributed localization algorithm LDB: We present the algorithm in Algorithm 1. The AUV moves paralleled with x-axis from the bottom to the top as shown in Figure 7 with a constant speed v over the 3D sensor deployment space. The beacon packet sends by AUV including its current position \( x, y \), \( h_A \), the beacon angle \( \alpha \), the beacon interval \( t \), the AUV speed \( v \) and \( D_r \). At initialization phase, each node sets its measured depth information \( h \). A timer \( T \) is initialized to zero which used to count time. When the node receives a beacon successful, the timer \( T \) increases \( t \). We denote \( t_{\text{max}} \) as the maximum time which a node can hear the beacon continually. When \( T \) is greater than \( t_{\text{max}} \), the node will not receive any beacon because the AUV is moving out of its receiving range. As shown in Figure 9, node S’s first-heard beacon point is \( T_1 \) and last-heard beacon point is \( T_m \). The distance between these two points is \( 2r \). Because the speed of the AUV is \( v \), thus the maximum time which the node \( S \) can hear the beacons of the AUV is \( \frac{2r}{v} \).

When the node receives the first beacon from AUV successful, it stores \( x, y, h_A, \alpha, t, v \) and \( D_r \). Then it uses the data to compute the radius \( r \) of the circle in which it resides. It also sets the \( t_{\text{max}} \) and increases the timer \( T \). When the node receives the second beacon, it stores the AUV’s new position \( x, y \) and increases the timer \( T \). After that, it checks whether the difference between \( y_1 \) and \( y_2 \) is equal or greater than \( D_r \). If it is true, which means the AUV has moved to another paralleled line, it stops receiving

\begin{algorithm}[h]
\begin{algorithmic}[1]
\Procedure{Initialization}{}
\State \( h \leftarrow h \)
\State \( T \leftarrow 0 \)
\State \( \text{FirstRec} \leftarrow \text{FALSE} \)
\State \Return \( \text{TRUE} \)
\EndProcedure
\Procedure{ReceivedBeacons}{}
\If {\( \text{FirstRec} = \text{FALSE} \)}
\State \( \text{FirstRec} \leftarrow \text{TRUE} \)
\State \( \text{FirstReceivedB}() \)
\Else
\State \( \text{SecondReceivedB}() \)
\EndIf
\EndProcedure
\Procedure{FirstReceivedB}{}
\State \( x_1 \leftarrow x \)
\State \( y_1 \leftarrow y \)
\State \( t \leftarrow t \)
\State \( v \leftarrow v \)
\State \( \alpha \leftarrow \alpha \)
\State \( h_A \leftarrow h \)
\State \( D_r \leftarrow D_r \)
\State \( r \leftarrow \tan(\alpha/2) \times (h - h_A) \)
\State \( t_{\text{max}} \leftarrow \frac{2r}{v} + 1 \)
\State \( T = T + t \)
\State \Return \( \text{TRUE} \)
\EndProcedure
\Procedure{SecondReceivedB}{}
\State \( x_2 \leftarrow x \)
\State \( y_2 \leftarrow y \)
\State \( T = T + t \)
\If {\( |y_2 - y_1| \geq D_r \)}
\State \( \text{CalculateP}() \)
\EndIf
\EndProcedure
\Procedure{TimeCheck}{}
\If {\( T > t_{\text{max}} \)}
\State \( \text{CalculateP}() \)
\EndIf
\EndProcedure
\Procedure{CalculateP}{}
\State \( r \leftarrow \tan(\alpha/2) \times (h - h_A) \)
\State \( D_r' \leftarrow x_2 - x_1 + 2 \times t \times v \)
\If {\( D_r' < 2r \)}
\State \( \text{Use Equation(3) compute } x y \)
\Else
\State \( \text{Use Equation(6) compute } x y \)
\EndIf
\EndProcedure
\Procedure{TurnOffReceiver}{}
\EndProcedure
\end{algorithmic}
\end{algorithm}

© 2010 ACADEMY PUBLISHER
procedure and calculate its position using Equation (3) and (6) according to different $D'$. If this is not true, it then checks whether timer $T$ is larger than $t_{\text{max}}$. If timer $T$ is less than $t_{\text{max}}$, the node will wait another beacon to refresh its data until the timer $T$ is greater than $t_{\text{max}}$, then it stops receiving procedure and calculates its position. When the node has figured out its position, it may turn off its receiver temporarily to save energy.

The algorithm LDB is a distributed algorithm. Each node only needs to listen to beacons sending from the AUV and then compute their locations independently using Algorithm 1.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the LDB through extensive MATLAB simulations. The sensor nodes are randomly deployed in $1000m \times 1000m \times 100m$ 3D space. The AUV follows a pre-defined trajectory as shown in Figure 7 at fixed depth with a constant speed over the 3D deployment space, and the mounted transducer has the fixed acoustical radiating power to reach the farthest nodes that may be deployed on the bottom of the 3D space. We set the speed of AUV 1m/s ($\equiv 2$ knot). We set the radius of the node’s restrained movement at 0.5m. Our evaluation metrics are beacon intervals, directional beacon angles, localization accuracy. We use the ratio of localized nodes to measure the localization accuracy, which is defined as

$$R_{\text{LN}} = \frac{N_{\text{T}} - N_{\text{R}}}{N_{\text{T}}}$$

where $N_{\text{T}}$ is the total number of sensor nodes deployed in the 3D space and $N_{\text{R}}$ is the number of sensors by which their computed error exceeds a given threshold $T_{\text{E}}$. In our simulation, we set the node’s threshold $T_{\text{E}}$ at 6m.

A. Impact of Beacon Intervals

In Figure 10, we show the ratio of localized nodes for different beacon intervals. We vary the beacon interval from 1s to 15s. We set the angle of directional acoustic beam at $60^\circ$ and $D_v$ at 10m, 18m, 26m separately, in which 18m is calculated by Equation (14). The distance between AUV and the top surface of 3D space $H$ is at 20m.

From Figure 10, we observe that the ratio of localized nodes decreases as the beacon interval increases. This is because as the beacon interval increases, the adjacent-beacon distance $d$ increases which leads to the increment of the maximum horizontal error $E_{\text{max-h}}$, thus the ratio of localized nodes decreases. When we fix the beacon interval, smaller $D_v$ has higher ratio of localized nodes. The reason is that as $D_v$ becomes larger, more nodes can not receive enough beacons to calculate their positions. When the beacon intervals are less than 3s and $D_v$ satisfies Equation (14), the ratio of the localized nodes are all above 96%.

B. Impact of the Directional Beacon Angles

The angle of directional beacon is also a significant parameter, which influences the performance of our approach. Thus, we try to fix other parameters and evaluate the impact of the angle. Figure 11 illustrates the ratio of localized nodes with different angles. We set $H$ at 20m and $D_v$ is calculated by Equation (14). We vary the degree of the angle from $20^\circ$ to $90^\circ$ by increments of $10^\circ$. An increase in the degree of the angle results in a lower ratio of the localized nodes for different beacon intervals. This can be explained that as the angle becomes larger, the radius $r$ becomes larger, and the maximum vertical error $E_{\text{max-v}}$ also increases, thus the ratio of localized nodes decreases. From Figure 11, we also observe when the degree of the angle is below $60^\circ$ and the beacon intervals is below 3s, the ratio of the localized nodes are all above 92%.

C. Impact of the $H$

We evaluate the impact of the $H$ which is the vertical distance between AUV and top surface of the deployment volume. The angle of directional acoustic beam is set at $60^\circ$. We vary $H$ from 0m to 50m by increments of 10m. We choose three groups of beacon interval and $D_v$ to test

© 2010 ACADEMY PUBLISHER
D. The mean error and mean square deviation

We evaluate the mean error and mean square deviation of the algorithm. The angle of directional acoustic beam is set at 60° and \( D_v \) at 18m, in which 18m is calculated by Equation (14). The distance between AUV and the top surface of 3D space \( H \) is at 20m. We vary the beacon interval from 1s to 15s. As demonstrated in Figure 13, the mean error increases sharply as the beacon interval increases. This is because as the beacon interval increases, the adjacent-beacon distance \( d \) increases which leads to the increment of the maximum horizontal error \( E_{max-h} \), thus the mean error of nodes increases. We also observe that the corresponding mean square deviation also increases as the beacon interval increases, however, it increases a little slower compared with the mean error.

E. Impact of the Number of Sensor Nodes

From a sparse deployment to a dense deployment, we deploy 20 to 200 sensors in the same 3D space to evaluate our scheme. We fix other parameters, such as \( H \) at 20m and the angle of directional acoustic beam at 60°. The result of our simulations, demonstrated in Figure 14, evidences that the ratio of localized nodes does not change much when we deploy more sensor nodes in the volume. The reason is that our scheme does not depend on communication between sensor nodes, it only depends on the beacons sent by the AUV. The results also demonstrate that our scheme works well with sparse networks which has low average sensor node degree. We can imagine that LDB can localize those linear sensor networks whose sensor node degree is 1.

V. CONCLUSIONS AND FUTURE WORK

We proposed a novel distributed 3D localization scheme LDB, a range-free silent localization scheme which can be applied to both dense and sparse underwater sensor networks, especially for sparse networks with very low average node degree. Mounted with a directional
transceiver, an AUV patrols over the deployed 3D sensor space sending beacons towards the sensor deployment area. By passively listening the beacons sent from the AUV, the sensor nodes use the first-heard beacon point and the last-heard beacon point to localize their position. Our simulation shows that LDB has a high localization accuracy. The simulation also demonstrates that our scheme works well with sparse sensor networks.

For future work, we plan to apply LDB to real testbeds, where a directional acoustic transceiver is being constructed at our university. Meanwhile, we also plan to use AUV to localize 3D free drifting underwater sensor networks.

ACKNOWLEDGMENT

We thank Professor Lionel M.Ni, Yunhao Liu and the members in China 973 WSNs Joint Lab at ShangHai for discussions concerning this work, and the anonymous reviewers for their detailed comments.

REFERENCES

Hanjiang Luo is currently a Ph.D. candidate at Ocean University of China. He received BS degree and Master’s degree in computer science from Shandong University of Technology and Ocean University of China respectively. His research interests include wireless sensor networks and underwater acoustic sensor networks. He is a student member of the IEEE.

Zhongwen Guo received the B.S. (1987) degree in computer science and engineering from Tongji University, China, and the M.S. (1996) degree in applied mathematics from the Ocean University of China, and Ph.D. (2005) degree from the same university. He is currently a professor in the computer science at Ocean University of China, Qingdao. His main research interests are computer networks and distributed ocean information processing techniques, with special interest in routing protocols and algorithms for ad hoc, wireless sensor and underwater acoustic networks. He is a member of the IEEE.

Wei Dong received his BS degree in the College of Computer Science from Zhejiang University, and achieved all credits in the Advanced Class of Engineering Education (ACEE) of Chu Kechen Honors College from Zhejiang University in 2005. He is currently a Ph.D. student in the College of Computer Science of Zhejiang University, under supervision of Prof. Chun Chen. He is a student member of the IEEE.

Feng Hong received the BS degree from Computer Science Department in Ocean University Of China, and the Ph.D. from Computer Science and Engineering Department in Shanghai Jiaotong University. He is currently an assistant professor of the Computer Science and Technology Department in Ocean University of China, Qingdao. His research interests are sensor networks, peer-to-peer networks, distributed computing, marine data visualization. He is a member of the IEEE.

Yiyang Zhao received his B.Sc. degree from Tsinghua University in 1998 and Mphil degree from Institute of Electrical Engineering of CAS in 2001. After that, he began his Ph.D. study in Department of Computer Science and Engineering at HKUST. His research interests are generally in computer networks. In particular, he is now focusing on the localization sensing with RFID. He is also interested in wireless sensor networks.
Abstract—Firefighting is a dangerous profession, which often faces complex, variable, and uncertain situations. Only depending on conventional tools, cannot the firefighters cooperatively work effectively. We present a mobile fuzzy decision support application solutions for firefighter cooperation in ad hoc networks based on the decision support architecture proposed by William J. O’Brien. The application solutions, involve path navigation, danger reminder, rescue cooperation, and extinguishing cooperation. By these solutions, the firefighters can obtain complete situation information of the fire scene. To realize the application solutions, we discuss two main issues, map synchronization and alternative optimization. For an effective map synchronization scheme, we present synchronization table strategy. For alternative optimization, we give a selection process based on triangular fuzzy set. The realization of the decision support applications, and the interfaces’ patterns are discussed in the end.

Index Terms—Ad hoc Networks, firefighting, mobile decision support, map synchronization, alternative optimization

I. INTRODUCTION

As we know, fire disaster is huge destroy to our life. When it happens, how to deploy as quickly as we can and achieve effective cooperation is an important issue. With disaster happened, all kinds of communication patterns that rely on infrastructures maybe be useless. Like the latest Wenchuan Earthquake in China, all infrastructures were destroyed. Mobile ad hoc networks are decentralizing, self-organizing, and highly dynamic networks formed by a set of mobile hosts connected through wireless links, without requiring any infrastructure, that can be used for emergent communication. Facing plenty of information, another problem of succor in rescue is how to draw valid information for assistance. Mobile Decision Support Systems (MDSS) are good at it.

With development of ubiquitous compute, all kinds of MDSS are presented, like H-Vet [1], CampusGIS [2], CrossTalk [3], iTriage [4], PBST [5], MedNet[6], and WCSCW [7]. Most of these are designed to support traditional work, not for firefighting. According to the Xiaodong Jiang’s investigation[8] of firefighting, there are three main problems in firefighting scene: first, firefighters often have an incomplete picture of the situation; second, there are several weaknesses in the existing communication systems used by firefighters; third, firefighters operate in extremely harsh environments during a fire spreading.

Some decision support systems like MobileEmerGIS [9], GeoBIPs [10] are designed for firefighting. However the communication modes in the systems are all centralized, which are easily destroyed in firefighting scene. Siren [11] is a decentralized mobile decision support system for firefighting, but it based on Wi-Fi network, which can only support in a small scale and can’t obtain the whole situation of fire scene. The Cooperative Decision Support we provided is based on mobile Ad hoc Network, which not only offers local decision support but also can achieve global cooperation decision. Moreover, in view of firefighting is a dangerous profession that calls for quick decision in high-stress environments, we add some simple operations to increase its effectiveness and efficiency.

The rest of this paper is organized as follows. Section 2 we discuss the architecture of The Cooperative Decision Support and its four application solutions in detail. In section 3, we describe an effective map synchronization scheme for The Cooperative Decision Support. Then, we present a method for calculating the best assistance alternative and interpret the process of decision initiation in section 4. In section 5, we give some interfaces of the decision support system. Finally, we conclude the paper and discuss what future works will be going on.

II. ARCHITECTURE AND APPLICATION SOLUTIONS

The architecture for decision support in ad hoc networks presented by William J. O’Brien[12] is composed of three layers (from bottom to top): Sensor Communication Layer (SCL) for handling physical communication between devices; Data Processing Layer (DPL) for data processing and abstraction of sensor data...
from specific device; Decision Support Layer (DSL) for decision application.

In this section we describe four application solutions for firefighting in detail under the architecture. First application is Path Navigation, as we know, firefighters are always dispatched to unacquainted places, and once firefighters enter into fire scene they maybe easily lost their ways. Using PDA with GPS, firefighters can get the best path to destination through operating PDA. GPS will obtain firefighter’s location and algorithm like Dijkstra in DSL will calculate the optimum path to the place where he specified. In view of firefighter can’t be staring at PDA all the time, we can fix left and right denotation lamp on his or her safe cap to navigate. While need to turn left, left lamp is sparkling; similarly need to turn right, right lamp is sparkling; while on the opposite direction both are sparkling; while on the correct direction both are off. With such denotation, the lamps are to achieve navigation but don’t call for firefighter’s plenty attention. If encountering obstacles stop his or her way, through operating PDA, he or she will mark this point and broadcast this information to update other point’s fire scene map.

Second application is Danger Reminder, in fire scene, a lot of potential dangerous factors (e.g. close to explosive or toxic gas) will threaten firefighters’ life. It’s important to remind of the danger while closing to. Once firefighter walk into fire scene, PDA monitors the information collected by wireless-sensors. The danger remaining process is similar to information reminder described by William J. O’Brien[12]. If dangerous information is collected, PDA will broadcast it simultaneously.

It’s a usual instance that firefighters need assistances from other points, we design third application Rescue Cooperation to help us find out the best assistance alternative. First, we select an assistance scheme integrated in PDA. And PDA will get the full-scale situation. According to the information collected, it will calculate the best alternative. These will be discussed intensively in section 4. Then assistance request is broadcasted, which not only includes assistance request but also some useful information (e.g. the optimum route of arriving to destination) for assistance staff.

Fourth application is Extinguishing Cooperation, when firefighters reach burning area, wireless-sensors within SCL measure all kind of information to estimate the situation of fire, which accomplishes in DSL based on data integration method like neural network and fuzzy logic. According to the situation estimating, PDA will judge the fire-level and evaluate the grade of current resources satisfaction. If needed, PDA will give some advices. For complicated fire scene, a single firefighter can not manage the situation, and need group firefighter’s cooperation decision to control the dynamic situation. If group decision-makers confirm assistance request, the following process is just like Rescue Cooperation. If not needed, the grade can also be used for sending to assistance-request-point for calculating the best alternative.

III. MAP SYNCHRONIZATION

As the situation we described, firefighter need to get the whole picture of fire scene, “where is burning, where is danger, where is obstructed and so on”, which is the essential of decision making. So information is an important part in our scheme. But “how to obtain message, how to express message, how to storage message, how to exchange message”, a lot of questions are derivative. In our scheme every firefighter will be equipped a suit of wearable computer with all kind of sensors-enable. Something like burning message, danger message can be easily achieved by hardware. In the following paragraph, we will discuss how to achieve message expressing, storage, and exchanging, respectively.

A. Map Characters

Ad hoc networks are kinds of dynamic topology networks, which communication links are set up by transmitted power of hosts. Due to the limitation of host power, it is usual fact that firefighter may move out of the communication range of other firefighters, he becomes “communication-isolated” and no information can be transmitted between him and others. And because of complicate environment, some critical communication links may be interfered and the networks are partitioned into several sub-networks. In other words, due to firefighters’ moving and complicate environment, the whole network will be disconnected in one time or connected in the other time. As we described, every firefighter is an information collected point, some firefighters are composed of sub-networks, and every sub-network maintains only a part of the whole fire scene. Map data should be exchanged between the merged sub-networks to achieve consistent internal maps in the corresponding PDAs[13].

How to minimize the volumes of the map data exchanged between the sub-networks? Weihua Sheng[13] who researched on the way of multi-robots map detecting, presented a map synchronization scheme, which let each robot maintains a raw map table and a last-sequence-number table to achieve only exchanging latest increased data with each other. The raw map table consists of the map information discovered by each robot. The last-sequence-number table saves the last order of map data discovered by each robot. Once a new communication link established, if check out they aren’t in one sub-networks. They will send their last-sequence-number table to each other, and calculate what data should be exchanged with each other. Through this way, two robots can only transmit their new updated data to each other and achieve minimal data exchange. However this scheme still has some defects. For example, hosts in ad hoc networks usually have processing, storage, and battery limited, but the scheme request each host maintains a raw map, may result in redundant and overlap. This also may bring unnecessary data exchange, storage, and battery cost. Furthermore, scheme requires one more step to update local map from raw map. We present a
map synchronization scheme which will require less data exchange and storage.

B. Model Assumptions

For simplicity, fire scene is to be modeled as 2D occupancy grids. Each grid contains the message of coordinate, time-stamp, direction-passable, burning, and danger. Each firefighter’s PDA has mapping, localization and communication capabilities. The grid is considered as minimal unit what firefighter can move. And the sensing range is denoted by a circle of radius centered on the firefighter. When firefighter is traveling, it doesn’t carry out sensing and mapping. Each PDA is capable of localizing itself with respect to its own local map. The communication capability enables a PDA to directly talk to other PDA within the communication range, or to a remote PDA in the same sub-network through multi-hop communication[13].

C. Map Storage

In our scheme each firefighter maintains a local map table and a synchronization status table. The local map table stores the map information of current fire scene which is as shown in Table 1. Here, we transfer usual map into grid map, each grid contains a package of message, as Fig. 1.

Let \( M_{ij} = ((i,j), (E,S,W,N),(B,D)) \), indicates one message package of a grid, where \((i,j)\) is the coordinate of \(M_{ij}\), it can be any integrate value, \(t\) is the time stamp of detecting difference of \(M_{ij}\). We set \( t=0 \) before firefighters enter into scene, and sampling every one minute. \((E,S,W,N)\) is the direction passable to east, south, west and north, denoted by bool value respectively; \((B,D)\) is used to show \(M_{ij}\)'s burning and danger message, denoted by bool value respectively.

All grids of message package are initialized to be the state of original plan designed, for example \( M_{11} = \{(3,1,0),(1,0,1,0),(0,0)\} \).

The synchronization status table keeps each firefighter’s synchronization status with other one, is as following Table 2. Here, \( S_{ij}^{u} \) indicates synchronization status of the firefighter \( F_u \) in grid \( M_{ij} \) with others, can be represented like this, \( S_{ij}^{u} = \{S_{ij}^{u1}, S_{ij}^{u2}, \ldots, S_{ij}^{unu}\} \), the \( n \) is the ID of firefighter’s PDA in fire scene, \( S_{ij}^{vu} \) is a bool value means whether firefighter \( F_u \) in grid \( M_{ij} \) could synchronize with firefighter \( F_v \). For example, there are eight firefighters in fire scene, \( F_1, F_2, F_3, F_4 \) are in the same sub-networks. After one broadcasted by \( F_1 \) about a package message of \( M_{11}, M_{33}, M_{55}, M_{77} \) will be synchronization with \( F_1\), we can obtain \( S_{11}^{1} = \{0,1,0,1,0,1,0\} \).

C. Synchronization Algorithm

Now we discuss while a new communication link is established, how the two firefighters are to exchange their data. First, they will judge whether they were in one sub-networks, only the ones from different sub-networks need exchange data. As the scheme we described, for grid \( M_{ij} \)'s status, there are three possibilities between two PDAs. The first is that all are synchronization; second is that one show synchronization with the other but the other show no; third is that both show asynchronous with each other. For first possibility we needn’t exchange it, for second possibility only need the asynchronous one actively send the data to the other, for third possibility we need exchange their time-stamp to decide which one is latest then synchronize it. After information exchange, two PDAs will keep a synchronization map, and correspondence synchronization status bit will all be 1.

The step of the map synchronization algorithm on firefighter \( F_u \) is as follows:

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>LOCAL MAP TABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>M_{61}</td>
<td>M_{62}</td>
</tr>
<tr>
<td>M_{51}</td>
<td>M_{52}</td>
</tr>
<tr>
<td>M_{41}</td>
<td>M_{42}</td>
</tr>
<tr>
<td>M_{31}</td>
<td>M_{32}</td>
</tr>
<tr>
<td>M_{21}</td>
<td>M_{22}</td>
</tr>
<tr>
<td>M_{11}</td>
<td>M_{12}</td>
</tr>
</tbody>
</table>

![Grid map](Image)

**Fig.1. Grid map**
Firefighter $F_u$’s map-synchronization algorithm

1) if (any firefighter $F_v$ is reachable from this firefighter)
2) if ($F_v$ is already in the same sub-networks as this firefighter) return;
3) receive one grid $M_{ij}$’s synchronization status from $F_v$ and compare it, if (both show synchronization) goto 5), else if ($F_v$ show synchronization but $F_u$ no) require $F_v$ the data, else if ($F_v$ show synchronization but $F_u$ no) send $F_v$ the data, else if (both show asynchronously) $F_u$ sends the time-stamp then $F_v$ is to make decision of sending or requiring the data;
4) multicast synchronization data to other firefighters in one subnetworks, modify 1 for $M_{ij}$’s correspondence bit;
5) if (all grids are synchronization) goto end, else point to next grid and goto 3)

Utilizing the scheme described we can obtain the number of burning grids captured by sensors and integrate a fragment fire situation. But it isn’t the whole situation of fire which is need for decision. To obtain the whole situation, using fire-burning model to calculate the maximal number of burning grids accord current grids’ situation can be a way. So we can only get a bound of burning grids’ number, that it only is vague fire situation we can obtain. Following we will discuss how to use the data to select optimum assistance alternative, which three factors are taken into account, assistance distance, assistance staff, and assistance appliance. These factors’ estimation all depend on fire situation. So they are vague, that we employ triangular fuzzy set and its algorithm to solve it.

IV. ALTERNATIVES OPTIMIZATION

A. Process of Alternatives Optimization

The initiation of requesting assistance includes two steps, the first is broadcasting for collecting information, and the second is making out alternative which is the best satisfactory based on information of first step collected. We calculate the best alternative base on three criterions, Assistance-Distance (A-D), Assistance-Staffs (A-S), and Assistance-Appliances (A-A). PDA will integrate a lot of schemes which involve all kinds of preferences. For example, a scheme maybe describes like this “Need $*$ assistance-staffs as quickly as they can.” This scheme interpreted in three criterions are, assistance-distance most important, assistance-staffs important, and assistance-appliances less important. We can do some scheme settings like the reciprocal weights of these criterions, in prior. And $*$ is the number of assistance-staff can be given while using this scheme. When initiation-point determines the scheme and broadcasts it, the received-point will evaluate their grade of satisfaction of these criterions by the current information about fire-level, staffs, and appliances. For example, based on the current fire-level, PDA will measure the quantity of extinguishing agent to estimate possibility of appliances. After evaluation, results will be replied to initiation-point. Now initiation-point PDA obtains information involving the degree of suitability of received-point’s A-S and A-A. Get rid of low assistance-ability points, PDA calculate the distance to other assistance-point. Then PDA will produce all kinds of assistance-alternatives, and make out the best alternative based on three criterions.

B. An Example to Describe the Method of Alternatives Optimization

Assume such a situation, two firefighting group G1, G2 and two single firefighters F1, F2 conduct a firefighting and rescue task in a fire scene. G1 arrives one burning area and their PDAs calculate fire-level too high to extinguish by them accord the information collected. Then G1-leader’s PDA displays an advice and recommends a request-assistance scheme, waiting for G1-leader’s confirmation. Assume that G1-leader choose the scheme “Need 2 assistance-staffs as quickly as they can” which reciprocal weight of criterions’ settings given in Table 3.

<table>
<thead>
<tr>
<th>TABLE III. RECIPROCAL WEIGHT OF CRITERIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteria</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>A-D</td>
</tr>
<tr>
<td>A-S</td>
</tr>
<tr>
<td>A-A</td>
</tr>
</tbody>
</table>

This scheme is interpreted like this: for assistance distance, the shorter the better; for assistance staffs, two are the best fitness, one is narrow fitness; for assistance appliances (e.g. extinguishing agent), total quantity of extinguishing agent at least must be equal half of one complete jar, and if more than it is better. Three criterions described above denoted by triangular fuzzy number is A-D = (0, 0, 500), A-S = (1, 2, 2), and A-A = (0.5, 1, 2), which we had set in prior (Fig. 4).

The information was collected as Fig. 3 shown in section 5, G2 and F1 have assistance ability except F2. Received-point’s PDA evaluates he or his group’s possibility of one and two staff assistance. The possibility of appliance assistance is estimated by the grade of satisfying G1-leader’s best need (is 1 from (0.5, 1, 2)). These assumption values are shown in Table 4, here we mark the points with $m$ (e.g. F1 number is 1). These evaluations are complicated and uncertain that can’t be only determined by PDA, so it will pop-up a window displaying the results in linguistic value for decision-maker’s confirmation (Fig. 5). Assume received-points valuate the satisfaction of criterions as Table 4 shown.
Based on Table 4, G1-leader’s PDA gets rid of some invalid staffs (e.g. F2) and alternatives (e.g. Selecting one G2’s staff without assistance appliances as assistance staff), and produces four valid alternatives, 1A, 2A, 3A, and 4A. 1A is selecting F1 as assistance staff; 2A is selecting one G2’s staff as assistance staff; 3A is selecting two G2’s staffs as assistance staffs; 4A is selecting F1 and one G2’s staff as assistance staffs. These alternatives’ grade of satisfaction of three criterions is given in Table 5, where we assume the shortest distance of F1 from G1-leader is 100 meters and the shortest distance between G2-leader and G1-leader is 200 meters.

### TABLE IV
**RECEIVED-POINTS’ ASSISTANCE ABILITY**

<table>
<thead>
<tr>
<th>Criterions</th>
<th>Received-point (m)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>One staff assistance (v_m^{as(1)})</td>
<td>(1, 1, 1)</td>
<td>(0.8, 0.8, 1)</td>
</tr>
<tr>
<td>Two staffs assistance (v_m^{as(2)})</td>
<td>(0, 0, 0)</td>
<td>(0.3, 0.3, 0.4)</td>
</tr>
<tr>
<td>Appliance assistance (v_m^{ap})</td>
<td>(0.4, 0.4, 0.4)</td>
<td>(0.5, 0.5, 1)</td>
</tr>
</tbody>
</table>

Based on Table 4, G1-leader’s PDA gets rid of some invalid staffs (e.g. F2) and alternatives (e.g. Selecting one G2’s staff without assistance appliances as assistance staff), and produces four valid alternatives, \(A_1\), \(A_2\), \(A_3\), and \(A_4\). \(A_4\) is selecting F1 as assistance staff; \(A_2\) is selecting one G2’s staff as assistance staff; \(A_3\) is selecting two G2’s staffs as assistance staffs; \(A_4\) is selecting F1 and one G2’s staff as assistance staffs. These alternatives’ grade of satisfaction of three criterions is given in Table 5, where we assume the shortest distance of F1 from G1-leader is 100 meters and the shortest distance between G2-leader and G1-leader is 200 meters.

### TABLE V
**GRADES OF SATISFACTION OF CRITERIONS**

<table>
<thead>
<tr>
<th>Solution ((A_i))</th>
<th>(A-D) (r_{i}^{ad})</th>
<th>(A-S) (r_{i}^{as})</th>
<th>(A-A) (r_{i}^{aa})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A_1)</td>
<td>(0.8, 0.8, 0.8)</td>
<td>(0.5, 0.5, 0.5)</td>
<td>(0.4, 0.4, 0.4)</td>
</tr>
<tr>
<td>(A_2)</td>
<td>(0.6, 0.6, 0.6)</td>
<td>(0.4, 0.4, 0.5)</td>
<td>(0.5, 0.5, 1)</td>
</tr>
<tr>
<td>(A_3)</td>
<td>(0.6, 0.6, 0.6)</td>
<td>(0.3, 0.3, 0.4)</td>
<td>(0.5, 0.5, 1)</td>
</tr>
<tr>
<td>(A_4)</td>
<td>(0.6, 0.7, 0.6)</td>
<td>(0.8, 0.8, 1)</td>
<td>(0.9, 0.9, 1)</td>
</tr>
<tr>
<td>Ideal</td>
<td>(0.8, 0.8, 0.8)</td>
<td>(0.8, 0.8, 1)</td>
<td>(0.9, 0.9, 1)</td>
</tr>
<tr>
<td>Anti-ideal solution</td>
<td>(0.6, 0.6, 0.6)</td>
<td>(0.3, 0.3, 0.4)</td>
<td>(0.4, 0.4, 0.4)</td>
</tr>
</tbody>
</table>

Here, \(r_{i}^{ad}\), \(r_{i}^{as}\), and \(r_{i}^{aa}\) can be calculated respectively as following, \(D_i\) represents the shortest distance between assistance point and initiation point which can be calculated through algorithm like Dijkstra. \(D_{\text{max}}\) indicating the maximal \(D_i\) which can be considered to be assistance point, had been known when G1-leader chosen the scheme, is 500 learning from \(A-D = (0, 0, 500)\). And based on the equation \(r_{i}^{ad} = 1 - \frac{D_i}{D_{\text{max}}}\), we can calculate \(r_{i}^{ad}\), if the assistance staffs of alternative are from the same point. For example, \(A_4\)’s degree of suitability of \(A-D\), \(r_{1}^{ad} = 1 - \frac{100}{500} = 0.8\). If assistance staffs of \(A_i\) is from different points, without loss of generality we assume that the \(r_{s}^{ad} (s = 1, 2, ..., n)\) are those single points’ degrees of suitability of distance, then

\[
r_{i}^{ad} = \text{min}_s (r_{s}^{ad}), \frac{1}{n} (\text{max}_s r_{s}^{ad}) \text{ or } \text{max}_s (r_{s}^{ad}) ,
\]

for example

\[
r_{4}^{ad} = \frac{1}{2} (0.8 + 0.6), \max (0.8, 0.6) = (0.6, 0.7, 0.8).
\]

Let \(t\) assistance staffs’ degree of satisfying A-S be denoted by \(d^{as(t)}\), and the evaluation of staff’s assistance-ability from received point be denoted by \(v_m^{as(t)}\). We assume \(d^{as(1)} = (0.5, 0.5, 0.5)\) and \(d^{as(2)} = (1, 1, 1)\). Then \(r_{i}^{as} = d^{as(t)} \otimes v_m^{as(t)}\), for example,

\[
r_{2}^{as} = d^{as(1)} \otimes v_2^{as(1)} = (0.5, 0.5, 0.5) \otimes (0.8, 0.8, 1) = (0.4, 0.4, 0.5).
\]

If assistance appliance of alternative is from the same point, then \(r_{i}^{aa} = v_m^{aa}\), where \(m\) represents the point’s number. When assistance appliance of alternative is from different points, we calculate the result based on \(r_{i}^{aa} = v_1^{aa} \oplus v_2^{aa} \oplus ... \oplus v_n^{aa}\), here without loss of generality, we assume that \(v_s^{aa} (s = 1, 2, ..., n)\) are those single point’s valuation of their appliance assistance ability.

For Table 3, we transfer it into triangular fuzzy numbers, and then based on the graded mean integration representation method\cite{14}, the relative weights scale of each criteria measuring method presented by Kahraman et al.\cite{15}, and the geometric method proposed by Buckley\cite{16}, we obtain Table 6 as following.

### TABLE VI
**THE INTEGRATED FUZZY WEIGHT AND WEIGHT CRITERION**

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Integrated fuzzy weight</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-D</td>
<td>(0.2425, 0.418, 0.6491)</td>
<td>0.4273</td>
</tr>
<tr>
<td>A-S</td>
<td>(0.2425, 0.388, 0.6725)</td>
<td>0.4112</td>
</tr>
<tr>
<td>A-A</td>
<td>(0.1316, 0.194, 0.3001)</td>
<td>0.2013</td>
</tr>
</tbody>
</table>
Employ the concept of distance between two triangular fuzzy numbers, the distance of alternative \(i\) versus ideal and anti-ideal solutions, \(D_i^*, D_i^\sim\) [17] Table 5, and Table 6 we can obtain the distance between alternatives and ideal and anti-ideal solution described by Table 7 as following:

**Table VII**

<table>
<thead>
<tr>
<th>Alternative</th>
<th>(A_1)</th>
<th>(A_2)</th>
<th>(A_3)</th>
<th>(A_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(D_i^*)</td>
<td>0.1822</td>
<td>0.2076</td>
<td>0.2374</td>
<td>0.052</td>
</tr>
<tr>
<td>(D_i^\sim)</td>
<td>0.1131</td>
<td>0.0748</td>
<td>0.0624</td>
<td>0.2467</td>
</tr>
</tbody>
</table>

Employ the relative approximation value of alternative \(i\) versus the ideal solution, \(C_i^*\) [17] we obtain the Table 8 to describe the close index of alternative versus the ideal solution.

**Table VIII**

<table>
<thead>
<tr>
<th>Alternative</th>
<th>(A_1)</th>
<th>(A_2)</th>
<th>(A_3)</th>
<th>(A_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C_i^*)</td>
<td>0.4631</td>
<td>0.2649</td>
<td>0.2081</td>
<td>0.826</td>
</tr>
</tbody>
</table>

As three criterions we used are all positive, so we learn \(A_4\) is the best alternative accord the Table 8.

**V. INTERFACES**

In previous sections, we discussed the method of collecting data and making out the optimum alternative, and preference settings, corresponding to database, way-base, and model-base according the tradition definition of decision support system, respectively. Now we will give four interfaces to aid firefighters making decision in mobile ad hoc network.

Executing the solution we described, the main interfaces as shown in Fig. 2, display four usual buttons which deployed according emergency rate.

**Fig. 2 Main interface**

As three criterions we used are all positive, so we learn \(A_4\) is the best alternative accord the Table 8.

**Fig. 3 Information collected by G1**

Selecting the best alternatives depend on the information as in Fig. 3. The information is collected by G1-leader. Here, a rectangle, such as G1, G2, represents a group and a circle, such as F1, F2, stands for a single firefighter. The prefix "e" and "d" means enable and disable, respectively. For example, “d-G1” indicates G1 having no assistance ability, and “e-F1” indicates F1 having assistance ability.

**Fig. 4 Scheme settings**

Fig. 4 is the interface of scheme parameters settings, whose value can be set in prior. These values include reciprocal weight of criterions, and triangular fuzzy number generated by scheme according situation. For example “A-D is EI than A-S” means criteria A-D is equal important than A-S, and “A-S: (1, 2, 2)” means......
requiring at least one assistance staff, two should be better.

When PDA received assistance request, it will calculate the grade of possibility for assistance, then pop-up an advice window to give a recommendation, like in Fig. 5. There are five levels to describe evaluation about assistance ability, “VB, B, M, G, VG” denotes “very bad, bad, medium, good, very good”, respectively. And like r-B stands for PDA’s recommendation.

VI. CONCLUSIONS

In this paper, we give a solution for firefighter cooperation decision support in Ad hoc networks. Then, we present an effective map synchronization scheme and describe a method to solve the problem of assistance alternative optimization by triangular fuzzy set. The process we described will assist firefighter execute firefighting more effectively and also can increase their safety. But there are still some problems, such as how PDA estimates the situation according its information collected, and how to make this solution more intelligent for practice, etc. We are managing to deal with those problems.

ACKNOWLEDGMENT

This work is partially supported by NSFC granted number 70271001, China Postdoctoral Fund granted number 2002032191 and Fund of Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education.

REFERENCES

Yanwei Chen received the Master’s Degree in electronic and electrical engineering from Donghua University of Information Science and Technology, China, in 2009. His recent research interests include mobile decision and Ad hoc Networks. He published 4 research papers in journals and conferences.

Demin Li received the PhD in electronic and computer engineering from Nanjing University of Science and Technology, China, in 1998. He is currently a professor of the Department of Telecommunication and Electronic Engineering, College of Information Science and Technology, Donghua University, Shanghai, China. From March 2002 to June 2005, he was a research scientist of Nanjing University, China. His recent research interests include telecommunication system engineering, wireless mobile networking, mobile decision theory and mobile decision support systems. He published more than 60 research papers in journals and conferences. He is currently as Associate Chairman of the Circuits and Systems Committee in Shanghai.

Chenwen Wang is a graduate student major in electronic and electrical engineering in college of Information Science and Technology, Donghua University.

Jiacun Wang received the PhD in computer engineering from Nanjing University of Science and Technology (NUST), China, in 1991. He is currently an associate professor of the Software Engineering Department at Monmouth University, West Long Branch, New Jersey, USA. From 2001 to 2004, he was a member of scientific staff with Nortel Networks in Richardson, Texas. Prior to joining Nortel, he was a research associate of the School of Computer Science, Florida International University (FIU) at Miami. Prior to joining FIU, he was an associate professor at NUST. His research interests include software engineering, discrete event systems, formal methods, wireless networking, and real-time distributed systems. He authored Timed Petri Nets: Theory and Application (Norwell, MA: Kluwer, 1998), and published more than 50 research papers in journals and conferences. He is an Associate Editor of IEEE Transactions on Systems, Man and Cybernetics, Part C, and has served as a program committee member for many international conferences. Dr. Wang is a senior member of IEEE.
Relay Aided Wireless Multicast Utilizing Network Coding: Outage Behaviour and Diversity Gain

Chen Zhi*, Chen Wei*, Pingyi Fan†, and Khaled Ben Letaief ‡
Department of Electronic Engineering, Tsinghua University, Beijing, P.R.China
E-mails: {chenzhi06@mails,c-w05@mails, fpy@mail}.tsinghua.edu.cn
National Mobile Communications Research Laboratory, Southeast University, China
†Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong
E-mail: eekhaled@ece.ust.hk

Abstract—Broadcast nature of wireless networks can be exploited to provide a flexible transmission, especially in multicast service. The potential relay then is capable to participate in message forwarding. In this paper, we first presents a network coding based cooperative (NCBC) multicast scheme exploiting limited feedback, where the source transmits two separate signals to multiple destinations in two successive half slots. The relay may combine the signals if it received two signals correctly and forward it to destinations in the next half slot. The destinations, therefore, can recover signals either from direct transmission or the relay forwarding. The performance analysis on the developed NCBC multicast protocol is given in the viewpoint of physical layer, such as the outage probability and diversity order. It is demonstrated that the NCBC multicast scheme can work better than the source direct multicast in terms of outage probability. Meanwhile, the NCBC multicast scheme can achieve full diversity gain (diversity two for one relay case). Comparing with the known relay schemes, i.e., amplify-and-forward (AF) and selection decode-and-forward cooperation schemes, it shows that the NCBC multicast scheme achieves almost the same outage performance with relatively less bandwidth and energy consumption.

Index Terms—Multicast, NCBC, network coding, cooperation

I. INTRODUCTION

Multicast is widely applicable in communication networks and has been extensively investigated in [1]-[4]. It is a way to distribute information from a single transmitter to multiple intended receivers in a network. For instance, in a wireless sensor network, sensed information may be needed to multicast to neighboring nodes for information gathering. Many previous works mainly focused on how to improve multicast or broadcast efficiency while guaranteeing the reliability [1]-[10]. In this paper, we consider wireless multicast. It is known that wireless medium has the broadcast nature, which can be exploited so that the same information can be transmitted to all its intended destinations simultaneously without paying any other extra cost. Meanwhile, due to wireless intrinsic fading, channels between the source and the destinations may not be good enough all the time to guarantee reliable transmissions and channels of different users may be different for each specific realization. Efficient multicast schemes to achieve high performance and reliability are therefore required. There are many ways to guarantee reliability, like ARQ technique or redundancy channel coding, etc. In ARQ technique, if a packet is not received successfully, the transmitter will retransmit it. In multicast applications, information will be repeatedly retransmitted until all destinations receive it by using ARQ. Redundancy channel coding, another way to guarantee reliable information delivery, is well suitable to the applications that can tolerate packets errors to some extent. This paper focuses on how to improve throughput in terms of outage probability for a multicast network. In contrast to unicast scenario, in multicast networks, only a few destinations which lost a common packet may benefit from one retransmission. In this case, retransmission efficiency is low as some destinations still do not receive the retransmitted information from this retransmission. This motivates us to propose a multicast scheme utilizing network coding. One can observe that, different lost packets can be combined together in one packet, as long as these packets are lost by different destinations. In this way, more destinations can get benefit and recover its lost information previously in one retransmission. Transmission efficiency thus can be improved. Meanwhile, cooperation diversity is also intrinsic in wireless networks, so relaying is also exploited in this paper to further improve system performance in terms of outage probability.

A. Motivation

To illustrate the main idea of this paper, we first review previous progress in wireless communication. Recently, cooperative communication and network coding have attracted considerable attention, which motivates us to exploit them to improve multicast throughput. Below we review them briefly.

Corresponding Author is Prof. Pingyi Fan. Part of this paper was presented in IEEE HPN08.

© 2010 ACADEMY PUBLISHER
doi:10.4304/jnw.5.1.47-56
Relays are traditionally used to extend the communication coverage in a wireless network. However, in recent years, relay networks attract more attentions as new kinds of applications [11]-[29], node cooperation is necessary and can provide transmit diversity via virtual antenna arrays. The seminar work in [14] first introduced cooperation diversity. In these works the relay was better located between the source and the destination to join in communication. In [16], amplify-and-forward (AF) and decode-and-forward (DF) schemes were first introduced and demonstrated to achieve diversity. In AF scheme, the cooperation node overhears the information, amplifies and relays to the destination as a virtual antenna. Whereas in DF scheme, the relay first decodes the received message, and then re-encodes and forwards it to the destination. Transmit diversity can be thus achieved via the two methods, which can greatly improve system performance in terms of outage probability.

In a cooperative network, essentially, a relay or a cooperative user may provide a better channel when the source-destination channel is in deep fading hence the throughput can be improved via cooperation or relay. It was demonstrated that such improvement can be obtained for wireless unicast in the literature. It turns out that this technique can also be applied in wireless multicast networks to improve throughput. Since a typical relay is located among the source and multiple destinations, it can be used to forward information overheard previously to multiple destinations to achieve higher throughput and transmit diversity.

Besides cooperation, network coding is also an emerging topic since 2000. It is based on a simple basic idea stated in the landmark work [30]. The essence of it is to mix data in some intermediate nodes. The destination receiver can extract its own information from those received mixed data packet. In [30], network coding was shown to improve network throughput. For the multicast or broadcast case, appropriate mixing schemes at intermediate nodes was shown to achieve the max-flow-min-cut capacity in wire communications. Meanwhile, many recent works, have started to investigate wireless network coding [24] [4] [32]. For instance, in [4], the authors exploits network coding for wireless multicast throughput improvement.

### B. Related Work

In wireless networks, multicast and broadcast are well explored [1]-[10]. For relay network, it was first introduced by van der Meulen [11]. Cover and El Gamal in [7] focused on Gaussian noise relay channel. [16] proposed some efficient protocols to achieve transmit diversity and better outage behavior. [16] also introduced an incremental relaying protocol. In their protocol, if the source-destination SNR is sufficiently high, the relay does nothing as the message can be successfully decoded by the destination. Otherwise the relay amplify-and-forwards what correctly received to help the destination decode the message. Hunter in [22] introduced another cooperation, coded cooperation, where each partner transmits incremental redundancy bits instead of repeating received bits. Both Laneman and Hunter demonstrated that cooperation can achieve higher capacity and transmit diversity of two for two-user cooperation. Gamal, Madsen and others, extended cooperation analysis and discussed diversity-multiplexing tradeoff or resource allocation problems from an information theoretic point of view [17] [18] [28].

Besides cooperation mechanism, network coding is employed to improve throughput in wireless systems. A form of wireless network coding is given in [31] [32]. In these papers information exchange through an intermediate node can be performed efficiently using network coding. In [24], the authors investigated diversity gains offered by network coding in a wireless network with distributed antenna system (DAS). It was shown that DAS and network coding leads to lower outage probability at a lower cost. Additional gain can be obtained by implementing user cooperation. But it focused on uplink information transmission in cellular wireless network with DAS. Another work [31] considered multiple-access relay channel utilizing network coding. In [31], the relay combined the messages received from the two sources and forwarded them to the destination. At the destination, the message thus could be recovered from either direct transmission or relay aided network coding. It was shown that network coding based cooperation has better performance in terms of outage behavior. However, the idea in [31] is not directly applicable in multicast scenarios. In [4], they proposed a multicast protocol utilizing network coding for packet retransmission. It focused on multicast routing by utilizing random graph theory and cooperation was not considered. Moreover, outage behavior and diversity was not investigated in [4], which is the focus of this paper. There is another interesting work [34]. Instead of multi-hop transmission, many relays cooperated with the transmitter to improve the throughput. Some information thus should be exchanged among the cooperation relays for consistency and more network resources was required for this information exchange. In this paper, we will only investigate cooperation with only one cooperation node and analyze its performance of physical layer. [25] employed both cooperation and network coding to further improve system performance of wireless communication, but it focused on wireless unicast scenarios.

In this paper, we try to combine the relay cooperation and network coding techniques to propose a network coding based cooperative (NCBC) multicast scheme, which exploits limited feedback and requires only partial CSI. Another main contribution is that we also investigate NCBC protocol performance from the viewpoint of physical layer, including the outage probability, diversity order and spectral efficiency etc and made some comparisons with other known relay schemes.

The rest of this paper is organized as follows. Section II describes system model and details of a practical network coding based (NCBC) multicast protocol is also presented. In Section III, outage behavior, diversity order
and spectral efficiency for the developed NCBC multicast scheme are analyzed. In section IV, numerical results are presented to verify the performance improvement of this NCBC multicast scheme. Conclusions of this paper is drawn in the last Section.

II. SYSTEM MODEL

Let us consider the system model shown in Fig.1. It consists of 4 nodes: one source $s$, one relay $r$ and two destinations $d_1$ and $d_2$. Concretely, Fig.2 shows 4 different cases based on the physical topology of the four nodes. The first three sub-figures focus on symmetric case where two destinations enjoy source-destination channels with the same average fading gain. However, the asymmetric case where the two destinations have unequal source-destination channel is also depicted in Fig.2(d) for consideration. Meanwhile, Fig.2(a)-Fig.2(c) represent equal fading gain, poorer fading gain, higher fading gain of relay-destination channels compared with that of source-destination channels, respectively.

In practice, the source may usually select a relay which is located among source and destinations, where the R-D channels have relatively higher fading gain than corresponding S-D channels. However, to show spatial diversity gain obtained via relaying compared with direct multicast, all these three physical topologies will be considered.

For the 4-node relaying multicast network in Fig.1, the source node $s$, which potentially exploits $r$ as a relay, transmits the same signals to both $d_1$ and $d_2$. In this paper, all these nodes are not allowed to transmit and receive signals simultaneously, in other words, they work in half-duplex mode.

Meanwhile, all channels are assumed to be mutually independent and the path gains $h_{sr}, h_{si}$ and $h_{ri}(i = 1,2)$ are subject to corresponding fading gains of source-relay, source-destination and relay-destination channels, respectively.

In conventional cooperation schemes developed in [16], like amplify-and-forward(AF) and decode-and-forward(DF), the source transmits a signal $X_a$ in the first half slot, then the relay forwards the same symbol in
the second half slot using AF or DF scheme in TDMA mode. In this paper, however, we develop a network coding based cooperative multicast scheme, called NCBC multicast scheme, for the sake of exploiting network coding benefit. In the developed NCBC multicast scheme, the source transmits two different symbols like $X_a$ and $X_b$ in the first two half slots, then the relay may forward the combined symbol $X_a \oplus X_b$ to the destinations in another half-slot under some scenarios, as shown in Fig.3. Obviously, this scheme can save almost 1/4 time slots and energy compared with conventional cooperation scheme under appropriate conditions. In other words, spectral efficiency is improved.

For this network coding based cooperation(NCBC) multicast scheme, the channel in the first two half slots is modeled as

$$y_i(n) = h_{si}x(n) + n_i(n) \quad (i = 1, 2) \quad (1)$$
$$y_j(n) = h_{sj}x(n) + n_j(n) \quad (2)$$
$$y_i(n+1) = h_{si}x_i(n+1) + n_i(n+1) \quad (3)$$
$$y_j(n+1) = h_{sj}x_j(n+1) + n_j(n+1) \quad (4)$$

where $x_s(n), x_j(n+1)$ are the two successive signals transmitted by the source, respectively. Meanwhile, $y_i$ and $y_j$ are received signals by the destinations and the relay, respectively.

In the NCBC multicast scheme, if the relay received the two signals successfully and the two destinations lost different signals, the relay forwards combined signal of the two signals to the intended destinations, so the received signal at the destinations is given by

$$y_i(n+2) = h_{si}x_i(n+2) + n_i(n+2), \quad i = 1, 2 \quad (5)$$

where $x_i(n+2) = x_i(n) \oplus x_j(n+1)$ is the combined signal transmitted by the relay and $y_i$ is the received signal by $d_i(i = 1, 2)$. It can be seen that each destination can recover the signal via this network coded signal under some scenarios. For instance, if $d_1$ only failed to receive $x(n)$ and $d_2$ only failed to receive $x(n+1)$, $x(n+2) = x(n) \oplus x(n+1)$ is forwarded by the relay if it successfully received the two signals in the first two half slots. Since $x(n+1)$ is already known at $d_1$, $d_1$ can easily recover $x(n)$ as $x(n) = x(n+2) \oplus x(n+1) = x(n) \oplus (x(n) \oplus x(n+1))$. Similar operations can be taken by $d_2$ to recover $x(n+1)$.

However, in the developed NCBC multicast scheme, the combined signal is not allowed to be forwarded in the third half slot under some scenarios, instead original signals may be forwarded. For instance, if $d_1$ fails to receive the two signals, two half slots are needed to retransmit the two signals, respectively. In such a case, the NCBC multicast scheme consumes the same amount of time slots as the cooperative schemes developed in [16].

In formulas (1)-(5), $h_{ij}$ represents the fading gain of the corresponding channel, and $n_{ij}$ is the additive Gaussian white noise, where $i \in (s, r)$ and $j \in (r, 1, 2)$. We assume that $h_{ij}$ is mutually independent and the current fading coefficients are known only to the corresponding receivers, but not to the transmitters. The transmitters only know the statistics of fading gain of the transmission channels, but not the current fading level. In other words, partial CSI is assumed to be known throughout this paper. Statistically, $n_{ij}$ is modeled as a zero-mean mutually independent, complex Gaussian random variable with variance $N_0/2$ per dimension. For Rayleigh fading, path gains $h_{ij}$ are modeled as a zero-mean, independent, circularly symmetric complex Gaussian random variable with variances $E[|h_{ij}|^2] = \sigma^2_{ij}$. For simplicity, the average SNR for each channel is represented as $\Gamma_{ij} = \sigma^2_{ij} \gamma_{ij}^{0.5}$ [22].

### A. Description of Network Coding Based Cooperative Multicast Scheme

In this subsection, we describe the details of the network coding based cooperation(NCBC) multicast scheme that can be utilized in the wireless network for broadcast or multicast service. This scheme needs limited feedback to determine relaying actions. Due to exploitation of feedback, the NCBC multicast scheme, improves spectral efficiency like incremental relaying scheme in [16], as the relay is only utilized when necessary. In our developed NCBC multicast scheme, the radios employ repetition codes as we are more interested in cooperative network coding gain. Moreover, we focus on repetition coding for its low-complexity implementation.

As the NCBC multicast scheme requires limited feedback, it is somewhat similar to conventional ARQ, which requires the source to retransmit if the destination provides a negative acknowledgement via feedback about the transmitted signal. In the NCBC multicast scheme, however, the relay retransmits instead of the source as long as it successfully received the transmitted signal, if needed. Otherwise the source retransmits the signal. Because of this, feedback from the relay is required by the source to determine whether it or the relay needs to retransmit the signal, if retransmission is required by the destinations. Meanwhile, feedback from the destinations is also required by the source and the relay, like in ARQ. For instance, the destinations can inform success or failure for each signal by broadcasting a 2-bit feedback to the source and the relay in a very low rate, which is thus assumed to be reliably received by the source and the relay.

In the 2-bit feedback from the destinations, the first bit indicates the identity of the destination and the second one indicates whether the transmitted signal is received correctly or not by the corresponding destination. For example, "00" means that $d_1$ receives the signal correctly, whereas "01" means that $d_1$ fails to receive the signal. Similarly, "10" means that $d_2$ receives the signal correctly, whereas "11" means that $d_2$ fails to receive the signal. Including these 2-bit feedback from the destinations and feedback from the relay, retransmission actions can be determined.

It should be noted that there are indeed some possibilities that the relay cannot forward the combined signal,
such as the two destinations lose the same signal or one destination fails to receive two successive signals, etc. In these cases, the relay forwards the transmitted signal if it successfully receives this signal, otherwise the source needs to retransmit the signal. For clarity, different transmission possibilities is classified into 5 disjoint cases below.

Case 1: Both destinations successfully receive the two transmitted symbols. In this case, the relay does nothing and another two successive symbols can be transmitted by the source in the following time-slots. It is obvious that no outage event happens in this case.

Case 2: Only one destination fails to receive a symbol. In this case, the relay forwards the symbol in the third half time-slot, if it can successfully receive the symbol. Otherwise, the source needs to retransmit the symbol in the third half time-slot. The fourth half slot is thus saved compared with conventional cooperative schemes.

Case 3: Both destinations only failed to receive a common signal. In this case the relay forwards the common signal failed previously to the destinations in the third half slot, if the relay received it correctly. Otherwise, the source needs to retransmit the signal again in the third half slot. The fourth half slot is also saved in this case compared with conventional cooperation schemes.

Case 4: The two destinations failed to receive different signals. In this case, the relay forwards the combined signal via XOR operation to both destinations if the relay successfully received the two signals. Otherwise, the source needs to retransmit the combined signal. Note that each destinations can benefit from the combined symbol since another symbol is already known to each destination. This will clearly improve the transmission efficiency. It is also observed that, on average, only $3/2$ time slots are needed for information transmission and energy is saved compared with conventional cooperation scheme in this case.

Case 5: At least one destination failed to receive both $x(n)$ and $x(n+1)$. In this case, the relay can not forward the combined signal as it provides no information for the destination which failed to receive both signals. Under this scenario, two half slots are needed for forwarding the two signals, respectively. If the relay successfully received a signal, the relay will forward this received signal. Otherwise, the source needs to retransmit the signal.

B. Relay Selection for the NCBC Multicast Scheme

In wireless networks, there are many strategies to select a relay to help transmission in multicast protocols [34]. In the developed NCBC multicast scheme, however, the selected relay should be better located among the source and the destinations, although a randomly selected relay can also provide cooperative diversity. In fact, better relays can further reduce system outage probability. Accordingly, the better relay enjoys higher channel gains of the R-D channels compared with the S-D channels. That is why the relay is required to transmit information instead of the source after the relay received the signals successfully. In fact, relay can be selected by finding a good one with relatively high R-D and R-S channel gains.

We conclude this section by noting that this paper only focuses on a multicast network with one relay. Treating protocols that exploit more relays, along with their closed-form performance improvement, is beyond the scope of this paper.

III. PERFORMANCE ANALYSIS

In this section, we shall analyze the performance of the NCBC multicast scheme from the viewpoint of physical layer, including the outage probability, diversity order and spectral efficiency etc. We also consider the high-SNR case for the approximation of outage probability.

First of all, we shall introduce the definition of outage probability as follows.

By Shannon formula, the instantaneous channel capacity is $C(\gamma) = \frac{1}{2}\log_2(1 + \gamma)$ given that the instantaneous SNR is $\gamma$. The outage event happens if channel capacity $C(\gamma)$ is below a predefined threshold $R$. The corresponding outage probability is thus defined as $Pr[C < R]$ [22].

In a Rayleigh fading wireless environment, $\gamma$ is exponentially distributed with parameter $1/T$. Thus the outage probability is given by [22]

$$Pr[C < R] = Pr[\gamma < 2^{2R} - 1] = 1 - \exp(-\frac{2^{2R} + 1}{T})$$  \hspace{1cm} (6)

In the NCBC multicast scheme, for each transmission from a transmitter to a receiver in one half slot, the instantaneous channel capacity is given by

$$I(i, j) = \frac{1}{2}\log(1 + \frac{|h_{ij}|^2P}{N_0}) = \frac{1}{2}\log(1 + |h_{ij}|^2)[\text{SNR}]$$  \hspace{1cm} (7)

where $i \in (s, r)$ and $j \in (1, 2, r)$. For simplicity, we define $\text{SNR} = \frac{P}{N_0}$ which is the ratio of transmit power to background noise, or the received SNR without fading. The outage probability $Pr[I(i, j) < R]$ thus can be given by

$$p_{ij}^{out} = Pr[I(i, j) < R] = Pr[I(i, j) < R] = 1 - \exp(-\frac{(2^{2R} - 1)}{\text{SNR}})$$  \hspace{1cm} (8)

where $i \in (s, r)$ and $j \in (1, 2, r)$.

As the NCBC multicast scheme is adaptive, there are several possibilities of retransmission actions according to the reception scenarios at the relay and the two destinations, which is described above. We again express these possibilities briefly and analyze the outage probability for each case. So the outage probability can be calculated separately for different cases first and finally summed up.

Without loss of generality, we only consider the outage event of the signal $x(n)$ at $d_i$. The analysis also holds for other outage events. Below is the detailed analysis.

© 2010 ACADEMY PUBLISHER
Case 1: As the two signals are successfully received by the destinations in the first half slots, no outage happens in this case.

Case 2: In this case, only one signal is not successfully received for a destination. In this case, the corresponding outage event may happen if just $x(n)$ is not received successfully at $d_1$. Under this condition, outage event happens if $d_1$ again fails to receive the signal $x(n)$ forwarded in the third half slot. The relay forwards $x(n)$ if it successfully received $x(n)$. Otherwise the source needs to retransmit the signal. So the outage probability in this case is

$$(1-p_{s2})^2(1-p_{s1})^2 p_{s1}^2 p_{s2}^2 \cdot [p_{sr}^2 p_{s1} + (1-p_{sr})p_{r1}]$$

Case 3: In this case, if both destinations just fail to receive $x(n)$, the outage event for $x(n)$ at $d_1$ may happen. The outage event happens if $d_1$ fails to receive $x(n)$ in the third half slot. The relay forwards $x(n)$ if it successfully received $x(n)$ previously. Otherwise the source needs to retransmit $x(n)$. So the outage probability in this case is given by

$$(1-p_{s2})^2(1-p_{s1})^2 p_{s1} p_{s2} \cdot [p_{sr}^2 p_{s1} + (1-p_{sr})p_{r1}]$$

Case 4: In this case, if $d_1$ fails to receive $x(n)$ and $d_2$ fails to receive $x(n+1)$, the outage event may happen. Under this scenario, the combined signal $x(n) \oplus x(n+1)$ is retransmitted to the destinations. If the relay successfully received the two signals, it forwards the combined signal. Otherwise, the source needs to retransmit the combined signal. The outage probability in this case is

$$(1-p_{s1})^2(1-p_{s2})^2 p_{s1} p_{s2} \cdot [p_{sr}^2 p_{s1} + (1-p_{sr})p_{r1}]$$

Case 5: In this case, there are two scenarios that the corresponding outage may happen. The first one is that $d_1$ only loses $x(n)$ but $d_2$ loses both signals. The outage probability in this case thus is

$$p_{s1}^2(1-p_{s1})^2 p_{s2}^2 \cdot [p_{sr}^2 p_{s1} + (1-p_{sr})p_{r1}]$$

Since the five cases are disjoint, the outage probability for $x(n)$ at $d_1$ can be given by

$$p_{\text{out}}^1(x(n)) = (1-p_{s2})^2(1-p_{s1})^2 p_{s1} p_{s2} \cdot [p_{sr}^2 p_{s1} + (1-p_{sr})p_{r1}] + (1-p_{s2})^2 p_{s2} \cdot (1-p_{sr})p_{r1}$$

If the channels obey the Rayleigh fading, Eqn. (13) can be evaluated exactly by using Eqn. (6) and Eqn. (8).

A. Asymptotic Diversity Order Analysis

When SNR is sufficiently high, we have

$$p_{\text{out}}^1(x(n)) = 1 - \exp\left(-\frac{(2R-1)}{\sigma^2 \text{SNR}}\right)$$

$$≈ \frac{(2R-1)}{\sigma^2 \text{SNR}}$$

where

$$g = \frac{2R-1}{\text{SNR}}$$

Thus the outage probability in Eqn. (13) can be written as

$$p_{\text{out}}^1(x(n)) = g^2 \left(1 - \frac{g}{\sigma^2 s_1^2}\right) \cdot \beta_1 + g^3 \left(1 - \frac{g}{\sigma^2 s_2^2}\right) \cdot \beta_2 + g^3 \cdot \beta_3 + g^4 \left(1 - \frac{g}{\sigma^2 s_1^2}\right) \cdot \beta_4$$

where

$$\beta_1 = \frac{1}{\sigma^2 s_1^2} \left[ \frac{g}{\sigma^2 s_2^2} \cdot \frac{1}{\sigma^2 r_1^2} \right]$$

$$\beta_2 = \frac{1}{\sigma^2 s_1^2} \left[ \frac{g}{\sigma^2 s_2^2} \cdot \frac{1}{\sigma^2 r_1^2} \right]$$

$$\beta_3 = \frac{1}{\sigma^2 s_1^2} \left[ \frac{g}{\sigma^2 s_2^2} + \frac{1}{\sigma^2 r_1^2} \right]$$

$$\beta_4 = \frac{1}{\sigma^2 s_1^2} \left[ \frac{g}{\sigma^2 s_2^2} + \frac{1}{\sigma^2 r_1^2} \right]$$

It can be seen from Eqn. (16) that, as SNR → ∞, the outage probability behaves as $g^2 \propto \frac{1}{\text{SNR}^2}$. Since the smallest exponent of $\frac{1}{\text{SNR}^2}$ is two, the diversity order two of the NCBC multicast scheme can be achieved.

B. Spectral Efficiency of The NCBC Scheme

In this subsection, we investigate the spectral efficiency of the NCBC scheme. By analyzing the outage event and the required time slots presented above, we can evaluate the expected spectral efficiency by considering different possible cases. As an example, now let us consider the spectral efficiency for $d_1$. The expected spectral efficiency of the NCBC scheme thus can be given by Eqn. (17), where $R$ is the constant transmit rate (bit/s/Hz).

In fact, the spectral efficiency is obtained by summing up all the independent possible cases.
The NCBC multicast scheme achieves full diversity in terms of outage behavior. Monte-Carlo method is employed here to evaluate the outage probability for the NCBC multicast scheme. The outage probabilities for four different cases in Fig. 2 are compared. Without loss of generality, for the first three physical topologies, where both destinations have equal S-D channel fading gain on average, only outage probability for $d_1$ is examined. For the fourth case where each S-D channel has different channel fading gain, outage behaviors of different destinations are also presented.

In the simulation, all average channel fading gains are set to be unity, unless noted. We set $R = 1/2$ b/s/Hz to be the transmit rate. The values of these parameters are only examples to demonstrate system performance improvement in terms of outage behavior. Monte-Carlo method is employed here to evaluate the outage probability for the NCBC multicast scheme and other cooperative schemes in [16].

In Fig. 4, the outage probability of the NCBC multicast scheme under different physical topologies in Fig. 2(a)-Fig. 2(c) is compared with the direct multicast. It is shown that the NCBC multicast scheme achieves full diversity. One can also see that when the relay-destination channels have better channel gains than source-destination channels ($\sigma_{ri}^2 > 0.1\sigma_{si}^2, i = 1, 2$), the NCBC multicast scheme obtains about 15dB SNR gain compared to the direct multicast at the outage probability of $10^{-3}$. In particular, even when relay is located relatively far away from source and destinations, ($\sigma_{si}^2 = 10\sigma_{sr}^2 = 10\sigma_{ri}^2 = 1$), the NCBC multicast scheme still has over 5dB SNR gain improvement at the outage probability of $10^{-3}$, as it provides spatial diversity compared with the direct multicast.

Fig. 5 compares the outage probability of the NCBC multicast scheme with other known cooperative schemes, including AF and selection DF. It can be seen that the NCBC multicast scheme performs almost the same as selection decode-and-forward and gets roughly 5dB SNR gain over direct multicast at the outage probability of $10^{-2}$. These results demonstrated that the NCBC mul-

$$R_{NCBC} = \frac{2}{3} R \cdot (1 - p_{out}^2)^2 (1 - p_{s1}^2)$$

$$+ \frac{4}{3} R \cdot (1 - p_{s2}^2) (1 - p_{out}) \cdot p_{s1} \cdot [p_{sr} (1 - p_{out}) + (1 - p_{out}^2)(1 - p_{r1})]$$

$$+ \frac{4}{3} R \cdot (1 - p_{out}) (1 - p_{s1}^2) \cdot p_{s2} \cdot [p_{sr} (1 - p_{out}) + (1 - p_{out}^2)(1 - p_{r1})]$$

$$+ \frac{4}{3} R \cdot (1 - p_{out}) (1 - p_{s2}^2) \cdot p_{s1} \cdot p_{sr} \cdot [p_{sr} (1 - p_{out}) + (1 - p_{out}^2)(1 - p_{r1})]$$

$$+ R \cdot p_{s1}^2 (1 - p_{out})^2 \cdot [p_{sr} (1 - p_{out}) + (1 - p_{out}^2)(1 - p_{r1})]$$

$$+ R \cdot p_{s2}^2 (1 - p_{out})^2 \cdot [p_{sr} (1 - p_{out}) + (1 - p_{out}^2)(1 - p_{r1})]$$

(17)
destination channel case in Fig. 5 compares outage probability of the NCBC multicast scheme with various known cooperation schemes to achieve higher spatial diversity. Simulation results demonstrated that the NCBC multicast scheme can provide significant gains over the direct multicast transmission, especially in low-rate regime. We also demonstrated that this scheme achieves diversity order two with relatively less time resources and transmission power compared with the known cooperative schemes, such as amplify-and-forward (AF) and selection decode-and-forward.

It should be noted that only the single relay node in cooperative multicast network was considered in this paper. The extension to a multicast network with multiple relays is an interesting topic in wireless ad hoc network. Some further discussion on the developed NCBC multicast will be presented in [33].

ACKNOWLEDGMENTS

This work was partially supported by NSFC/RGC Joint Research Scheme No.60831160524 and by the open research fund of National Mobile Communications Research Laboratory, Southeast University, China.

REFERENCES


Zhi Chen received his B.S. degree from University of Electronic Science and Technology of China in 2006. He is currently pursuing his Ph. D at Department of Electronic Engineering, Tsinghua University, Beijing, China.

His research interests include Network Information Theory, Cross Layer Design and Network Coding etc.

Wei Chen received his B.S. degree from Xi’Dian University of Science and Technology in 2005. He is currently pursuing his Ph. D at Department of Electronic Engineering, Tsinghua University, Beijing, China.

His research interests include Network Information Theory, Cross Layer Design and Multiple Access Control etc.

Pingyi Fan received the B.S and M.S. degrees from the Department of Mathematics of Hebei University in 1985 and Nankai University in 1990, respectively, received his Ph.D degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China in 1994. From Aug. 1997 to March. 1998, he visited Hong Kong University of Science and Technology as Research Associate. From May. 1998 to Oct. 1999, he visited
University of Delaware, U.S.A., as research fellow. In March 2005, he visited NICT of Japan as visiting Professor. From June 2005 to July 2005, Aug. 2006 to Sept. 2006, he visited Hong Kong University of Science and Technology. He was promoted as full Professor at Tsinghua University in 2002.

Dr. Fan is a member of IEEE and an oversea member of IEICE. He has attended to organize many international conferences including as TPC Chair of International Symposium of Multi-Dimensional Mobile Computing 2004 (MDMC’04) and TPC member of IEEE ICC2005 etc. He is currently an editor of Inderscience International Journal of Ad Hoc and Ubiquitous Computing and Interscience Journal of Wireless Communication and Mobile Computing. He is also a reviewer of more than 14 international Journals including 10 IEEE Journals and 3 EURASIP Journals. His main research interests include B3G technology in wireless communications such as MIMO, OFDM, Multicarrier CDMA, Space Time Coding, and LDPC design etc., Network Coding, Network Information Theory and Cross Layer Design etc.

Khaled Ben Letaief (S’85-M’86-SM’97-F’03) received the B.S. degree (with distinction) and M.S. and Ph.D. degrees from Purdue University, West Lafayette, IN, in 1984, 1986, and 1990, respectively. Since January 1985 and as a Graduate Instructor in the School of Electrical Engineering at Purdue University, he has taught courses in communications and electronics. From 1990 to 1993, he was a faculty member with the University of Melbourne, Melbourne, Australia. Since 1993, he has been with the Hong Kong University of Science and Technology, Kowloon, where he is currently a Professor and Head of the Electrical and Electronic Engineering Department. He is also the Director of the Hong Kong Telecom Institute of Information Technology as well as the Director of the Center for Wireless Information Technology. His current research interests include wireless and mobile networks, broadband wireless access, space-time processing for wireless systems, wideband OFDM, and CDMA systems.

Dr. Letaief served as consultant for different organizations and was the founding Editor-in-Chief of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He has served on the Editorial Board of other journals, including the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS Wireless Series (as Editor-in-Chief). He served as the Technical Program Chair of the 1998 IEEE Globecom Mini-Conference on Communications Theory, held in Sydney, Australia. He was also the Co-Chair of the 2001 IEEE ICC Communications Theory Symposium, held in Helsinki, Finland. He is the Co-Chair of the 2004 IEEE Wireless Communications, Networks, and Systems Symposium, held in Dallas, TX, and is the Vice-Chair of the International Conference on Wireless and Mobile Computing, Networking, and Communications, WiMob05, to be held in Montreal, QC, Canada. He is currently serving as the Chair of the IEEE Communications Society Technical Committee on Personal Communications. In addition to his active research activities, he has also been a dedicated teacher committed to excellence in teaching and scholarship. He was the recipient of the Mangoon Teaching Award from Purdue University in 1990, the Teaching Excellence Appreciation Award by the School of Engineering at HKUST (four times), and the Michael G. Gale Medal for Distinguished Teaching (the highest university-wide teaching award and only one recipient/year is honored for his/her contributions). He is a Distinguished Lecturer of the IEEE Communications Society.
An Incentive Mechanism for Tree-based Live Media Streaming Service

Shuang Yang*,†, Xin Wang*

*School of Computer Science, Fudan University, Shanghai 200433, China
†State Key Laboratory of Integrated Service Networks, Xidian University, Xi’an 710071, China
Email: {06300720227, xinw}@fudan.edu.cn

Abstract—Tree-based structure is widely used in peer-to-peer streaming service and is the fundamental of many other structures, but it suffers a lot from the existence of free-riders. Based on the static analysis of the tree-based structure, we further discuss the streaming service working in dynamic situation. We then present an incentive mechanism for tree-based live streaming service which requires the least cost to change the overlay by rotation, so it performs well in the dynamic situation. This incentive mechanism not only limits the damage of free-riders, but also provides better quality of service (QoS) for users with more contribution. In addition, we show this incentive mechanism can avoid cheating to some extent. We run a series of simulation experiments to show its validity.

Index Terms—incentive mechanism, live streaming, tree-based, dynamic

1. INTRODUCTION

Peer-to-peer applications such as file distribution systems and streaming services are widely used. Their performances largely depend on users’ cooperation. Therefore, the existence of free-riders [1], who only download from their neighbors but rarely upload, results in decrement of download speed. Unlike file distribution system with higher robustness, streaming services is much more vulnerable, because with low download rate, the user cannot play the media smoothly. The most serious problem is that only a small ratio of free-riders, about 5 percent, can block nearly half of users [2], while that scale of free-riders hardly damage the file distribution system. Therefore, it is very important to propose an incentive mechanism to reduce the number of free-riders or limit their damage.

The studies of file distribution systems have longer history and more breakthroughs than those in streaming services. The major methods to analyze the file distribution system include game theory [3] [4], micro payment [5] [6] and reputation [7] system. Their main goal is to differentiate users and provide different QoS based on their contribution. Some of these incentive methods can be transplanted directly to the streaming service, but few succeed due to the different characteristics between file distribution system and live streaming service.

There are mainly three reasons preventing directly transplanting incentive mechanism from file distributed system for live streaming service. First of all, in order to play the media, users have to download the first block of data, or at least give higher priority to the first block. As a result, it is very often that one user does not have anything to upload to its upstream node in return, making the most frequently used strategy tit-for-tat less useful. Secondly, live streaming service is more sensitive in time, so fluctuation may be fatal, which makes no difference in file distribution system. Therefore, the attempting to find better partner might be questionable. Finally, while the existence of free-riders does not make destructive impact on the file distribution system, it has large impact on streaming service. The three reasons explain why these two kinds of p2p application cannot share the same incentive mechanism.

Some good incentives work well with MDC [8] or in VoD [9]. However, in this paper we focus on the incentives without MDC and in live streaming service with tree-based structure. Tree-based structure still plays a crucial role in the streaming service now, and it also forms the basis of often-used mesh-based and multi-tree structure. PeerCast [10] is an example of tree-based streaming service. Because tree-based structure obtains some advantages so as to be easy to control compared to mesh-based. For instance, it is difficult to determine the initial delay time in mesh-based live media streaming, while in the tree-based, the initial delay increases with the depth. Also, the tree-based contributes to the tree-mesh hybrid structure [11], a trend of streaming system. Furthermore, researches on trees may promote the study on structures, providing useful inspirations for improving their robustness. According to the previous research, one of the fatal problems of tree-based structure lies in free-riders. If the damage of free-riders in tree-based overlay can be controlled, the tree-based structure would be much more stable. In addition, this would benefit later mesh-based and tree-mesh hybrid structures. As mentioned above, an incentive mechanism based on tree structure is in need.

Based on the previous research [2] on static situation, we discuss the dynamic situation. We propose an incentive mechanism to reduce the damage of free-riders as well as provide better QoS to users with significant contribution.

The remainder of the paper is organized as follow. In section II, we introduce related work of the static situation. In section III, we describe the dynamic model and analyze it. In section IV, we describe a new incentive
mechanism for the tree-based streaming service. In section V, we show the stimulations and discuss the results. Finally, the Section V concludes the work.

II. RELATED WORK

In this part we briefly review the previous work [2]. In the previous work, we analyze the impact of freeloaders in both tree-based and mesh-based static situations. However, in this paper, we mainly focus on the tree-based. Some results in static situation still apply in the dynamic analysis, such as the blocking probability for one particular node.

The probability of a node to be free-rider is defined as $P$. The depth of the node $i$ is $H_i$. Its blocking probability is $B_i$. We define a node is blocked if all paths from the node to the root or the server have at least one free-riders. In tree-based structure, a user is blocked simply because at least one of its ancestors is free-rider. Then, we can get:

$$B_i = 1 - (1 - P)^{H_i - 1} \quad \text{(1)}$$

The nodes with height 0 or 1 (the root and its directed children) have no probability to be blocked.

(1) shows the blocking probability of a certain user only depends on the probability of the free-riders and its depth. This conclusion is the basic to encourage cooperation in the incentive mechanism which will be discussed in the fourth section.

The algorithm to calculate the distribution functions between the free-riders proportion and the blocking users in static situation is provided in the previous work in the static situation. In this paper, we are interested in the performance for the tree with the distinctive degrees and the given probability of a node becoming a free-rider, and the swap of branches is non-isomorphic cases. The detailed formulas are omitted here. We will use these formulas to calculate the expectation of the blocked, and then compare it with the counterpart in the dynamic situation.

In [2], the rearrangement is used to improve the tree. The main idea is to put users with large degree or upload bandwidth closer to the source in order to diminish the damage of free-riders. This method has good performance for it leads the least potential damage caused by freeloaders in static situation. The potential damage of tree $T$ $PD_T$ is defined as follow:

$$PD_T = \sum_{i \in T} B_i \quad \text{(2)}$$

However, the cost is colossal if the situation is dynamic because it may change the whole overlay frequently. In addition, this method can hardly be implemented by the distributed algorithm. Moreover, it actually does not punish the free-rider, so it cannot be regarded as an incentive mechanism. However, the idea of this method plays an essential role in the analysis discussed in the next section.

In the next section of the paper, we will go further of this work, focusing on the dynamic model in tree-based streaming service.

III. MODEL AND ANALYSIS

In this part, we discuss the model of the streaming system and then provide detailed analysis of the performance of the whole system.

A. Dynamic Model Description

As mentioned before, we focus on the tree-based structure, which has the source located on the root of the tree. The source delivers data to its children, who then deliver the data to their children. The upload bandwidth of the nodes are distinctive, thus the number of maximum children it can have differs. Since it is a single tree, every user has exactly one parent to get data, this is very different from multi-tree or mesh. Since every deliver tree in the multi-tree deliver one session, it can be thought as every user has exactly one parent to get data from a particular session.

In the deliver tree, there are three kinds of nodes. The first kind is the source node. It does not have upstream nodes and always has plenty of data. There is exactly one source node in a tree as the root. The second kind is normal users. In order to play, they ask for data from their parents. Meanwhile, they send data to their children. Their capacities of children are different, depending on their upload bandwidth. We define the capacity of node i as $D_i$. Moreover, we assume the normal nodes can always satisfy their children’s need when the number of children does not exceed the capacity and they have the data. In other words, if a user’s parent is a normal node which can play the media smoothly, this user can also play smoothly. The last kind is free-riders. They also have a number of children and these may be also different. However, we assume they cannot satisfy any of their children’s need, which means the upload speed is insufficient for playing or they do not upload at all. In a word, if a user has one ancestor who is a free-rider, it cannot get enough data for playing. Here, we do not consider other kinds of nodes. For example, the nodes have enough upload rate to some of their children but not all. This kind of nodes do not appear in our model, because it can lessen its degree to become normal users.

We assume all nodes have the same initial delay time. This assumption is widely used in analysis of the performance of live streaming system. This assumption leads to another assumption, i.e., the data transmitted is either the block next to the previous received block when the deadline is not missed, or the current block for playing when the deadline is missing.

As for dynamic environment, users can enter or leave the system. When a new user enters the system, first it asks the server for an entry. The server returns this new user a node which has at least one position for the new child. Moreover, since this is a tree, the control message is sent through this tree from the root, so if a user has the ability to add a child, the server would not return its children instead of it as the entrance position. In addition, for each node with full children, which branch the new user enters is randomly chosen with the same probability.
We assume that all users leave peacefully, which means the nodes which connect with the left are aware of their leaving, so they have time to react to their leaving. The leaving nodes’ children rejoin the tree when their parents leave. Their rejoicing is the same as the new users except the buffer may not be empty. The uniform initial delay time and the buffer ensure this action does not affect these nodes playing much, so the fluctuation during this change is ignored. Moreover, every node is assigned a probability to leave at every time slot.

As mentioned above, every node has a buffer. When the buffer is empty, the node cannot play, so it is blocked. If a node’s parent is not a free-rider, we assume this user’s buffer can quickly catch up with the buffer of its parent.

We assume the upload rate of every node is constant except when a new user joins where its parent wants to fill its buffer quickly. For instance, the free-riders never become normal nodes. Also, the capacity of normal nodes is fixed.

For further analysis of the cost of the structure change, we define two kinds of connections, the data connection and the control connection. These two connections are physically the same. The data connections form the transmission tree where data transfers. The control connections are to get information from the grandchild nodes, so the control connections exist between every pair of grandparents and grandchildren. The media data does not transfer via the control connections. We pay more attention to the change of the data connections, because their change may affect playing. Control connections can change asynchronously, so we pay less attention to their change. However, if we add one data connection which is already connected to send control information, the only thing to do is to change the connection tag, which costs less than to form a new connection.

Comparing to the previous static model, this model has some changes including the buffer, leaving and joining. The buffer is to reduce the fluctuation in the dynamic environment. The leaving and joining are compulsory elements in dynamic situation. In general, this model is similar to the static one apart from these new factors.

\section*{B. Performance Measurement}

In this subsection we introduce how to gauge the performance of the whole system. Here we define two conceptions to measure the performance of the system, the contribution weight and the blocking ratio.

The contribution weight \( W_T \) is assigned to every subtree \( T \). It is the weighted sum of upload rate of all nodes in the tree. It is defined as:

\[ W_T = \sum_{i \in T} (H_i + 1) \ast V_i \]  

(3)

Here \( H_i \) is the depth of node \( i \) in the subtree \( T \), \( V_i \) is the upload rate of this node. In particular, according to the model defined above, \( V_i \) equals to the number of the current children if it is a normal node or the source, or equals to 0 if it is a free-rider. If we use \( W_r \) to indicate the contribution weight of the subtree with root \( r \), (3) can be rewrite as follow:

\[ W_r = G_r + \sum_{i \in r's \ children} W_i \]  

(4)

\[ G_r = \sum_{i \in T} V_i \]  

(5)

Here \( G_r \) means the total upload rate of the subtree with root \( r \).

With constant upload rates, the contribution weight can measure the tree’s performance. The trees with less contribution weight perform better than those with larger contribution weight, because the nodes with larger upload rate is closer to the source thus to diminish the height of the whole tree. It is proved in theorem 1 that in the static situation, the rearrangement which satisfies the condition of theorem 1 has the least contribution weight when the free-riders are unknown.

This definition may be a little confusing. Because it is obvious that we want contribution or the upload rate to be as large as possible, so less contribution weight means better performance seems ridiculous. However, due to the fixed playing need and limited buffer size, the need of data is not infinity. Therefore, the upload rates cannot be unlimited large. Instead, they have their boundaries. When the upload rate is fixed for every user, the contribution weight pays more attention to the depth of nodes, or the distance between nodes to the source. The less contribution weight does not mean less upload rate but means the tree is shorter and more balanced.

The contribution weight also has something to do with the potential damage by free-riders. The potential damage shows the impact of free-riders on the structure. The potential damage is the expectation blocking users number for a particular tree structure, where the free-riders’ location is unknown. It is proved in theorem 2 that the tree with the least weight also has the least potential damage when the free-riders are unknown. Therefore, the attempt to achieve minimal weight is reasonable.

In addition, when the free-riders are known, the tree with the least weight contribution has the least nodes blocked. It is proved in theorem 3. We use the term ideal case to indicate the structure with the least contribution weight.

Another important function to measure the performance in streaming system is the blocking ratio \( R \).

\[ R = \frac{N_b}{N_t} \]  

(6)

\( R \) is the ratio of the blocked users. \( N_b \) is the number of users blocked and \( N_t \) is the number of users in total. In the static situation, the expected blocking ratio can be calculated. It grows when the total user number increases. Due to the random leaving and joining, they are fluctuated and mathematical expectations are difficult to calculate. Hereby, in the fifth section, we run a sort of simulations. The results can indicate the performance of the streaming service.
Playback continuity is also widely used to measure the performance of the streaming service. Since it focuses more on individuals while blocking ratio focuses more on time, and some users only stay in the system for a very short time, we use average blocking ratio instead of average continuity to gauge the performance.

IV. INCENTIVE MECHANISM

In this section, we focus on a new incentive mechanism based on tree which would work well in the dynamic situation. In this incentive mechanism, we use rotation to adjust the overlay, which has extremely low cost comparing to other methods. First we introduce the rotation method and then put forward the algorithm of this incentive mechanism.

A. Rotation

From the previous analysis, to get the lower contribution weight, the overlay has to change at time. However, unlike the static situation where we only need to change the overlay once, the overlay may change frequently. It is impractical to change the whole overlay, i.e., we cannot rearrange all nodes when any one node enters or leaves. A good idea is to change the overlay locally.

A simple idea is to swap two nodes, just like what to do with a heap. It is a good idea because the ideal case is a heap somehow. However, the swap in the networks costs a lot. If the two nodes to swap have $D_1$ and $D_2$ children each, the total data connections to change would be $D_1 + D_2$. Additionally, if the parent node has $D_1$ children, the swap would form $D_1 - 1$ new data connections which are not covered by control connections. It is much smaller than the change the whole overlay, but it is still not tolerable.

A solution to this problem is using rotation instead of swapping. It is similar to what we do with the R-B tree or AVL tree. The difference is that the tree to rotate is not a binary tree, so the rule to rotate should change a little.

We define two kinds of rotation. The first kind of rotation happens when there is room for a new child at the node to rotate. Fig. 1 illustrates the rotate1(D, B). One data connections will change in this rotation, i.e., disconnect the link (B, A), connect the link (D, A). The second kind of rotation happens when there is no room for new children at the node to rotate. Fig. 2 illustrate the rotate2(D, B). Two data connections have to change to keep D’s degree from increasing, i.e., disconnect the link (B, A), (D, F), connect the link (D, A), (B, F). This change is to move one branch of D to B in order to keep the degree of D. The F can be any branch of B. According to later discussion, F should has the least $G$.

The cost of rotation, or the data connections to change in the overlay is less or equal than 2, which is much smaller than swapping. What is more, these 1 or 2 new data connections are already covered by control connections. Therefore, rotation costs very little.

B. Algorithm

The first algorithm is simple. When a user is blocked for a period of time, the user asks the server for a new entry, or reenters the system. This period is called reenter time $RT$.  

Algorithm 1 Reenter for Node X

Step 1: BlockedTime = 0
Step 2: If cannot play, BlockedTime++
else blockedTime = 0
Step 3: If BlockedTime ≥ RT
Reenter

This algorithm can reduce the damage or blocking ratio somewhat. The cost is small. It changes one data connection where already does not provide enough download rate. However, this method is not good enough. If the free-riders block all paths from the root to the leaves, no matter how often other users reenter, they cannot download enough data to play.

A good incentive mechanism should consider two parts. First of all, the free-riders are to be punished or the users with large contribution are to be encouraged. Secondly, the performance of the whole networks is to be better off. The main idea of our incentive mechanism is to put the free-riders far from the source while put the nodes with large contribution closer to the source. As a result, the users with larger contribution are less likely to be blocked while the free-riders are more likely to be blocked, according to (1). Additionally, this method can reduce the contribution weight thus to achieve a better performance.
To implement the idea, we use rotate method for every node X.

**Algorithm 2 Rotate for Node X**

for each child y of X:
Step 1. Calculate \( W_y, G_y, W_z, G_z \) for all \( z \in \{ i \mid i's \ parent \ is \ y \} \)
Step 2. Calculate the Potential Weight(\( PW_z \)) for all \( z \in \{ i \mid i's \ parent \ is \ y \} \)
Step 3. If exist \( z_i : PW_{z_i} < W_y \) and \( PW_{z_i} > PW_{z_j} \) for all \( j, i \neq j \)
   
   Then Rotate(\( z_i, y \))
Step 4. Update \( G \)

In step 1, we recursively calculate the contribution weight and the sum of upload rate. This calculation of \( W_y \) is described in (4). We can rewrite the \( G_y \) in the recursive mode:

\[
G_y = V_i + \sum_{i \in E_r's \ children} G_i
\]  
(7)

In step 2, the potential weight \( PW_z \) means the weight of \( z \) after rotate(\( z, y \)). In detail, if the children number of \( z \) is less than its capacity, rotate1 is used, otherwise rotate2 is used. When using rotate1, the potential weight is followed:

\[
PW_z = W_y - 2 * G_z + G_y
\]  
(8)

When using rotate2, suppose the branch rooted in \( e \) moves to \( y \), then

\[
PW_z = W_y + G_y + G_e - 2 * G_z
\]  
(9)

To make \( PW \) as small as possible, the \( e \) should be the branch with minimal \( G \).

In step 3, we want to find \( y \)'s child \( z \), which has the minimal potential weight. Instead of computing the potential weight, we can simply compute the difference of the potential weight and \( y \)'s contribution weight. Then we rotate(\( z, y \)) to reduce the contribution weight. After each rotation, the contribution weight of the whole tree strictly decreases. Therefore, this algorithm would terminate.

In the last step, if rotation takes place, the \( z \) and \( y \) should be assigned with the new \( G \) so that other nodes can use the method to get the correct \( G \). It is true that the \( W \) also changes, but to judge whether a rotation needs, we only use \( G \) to compare. Since the nodes in the subtree do not change, the rotation does not affect \( G \) in the ancestors. In addition, only \( G_z \) and \( G_y \) change and they should be:

\[
G_z' = G_y
\]  
(10)

\[
G_y' = G_y - G_z
\]  
(11)

\[
G_y = G_y - G_z + G_e
\]  
(12)

If the first rotation takes place, the equation (11) applies. Otherwise, the equation (12) applies.

The advantage of this algorithm is that it changes the overlay structure locally, thus it is a distributed algorithm. Also, the change of links is small, so it does not affect much performance of users involved. This algorithm runs every fixed number of time slots. It does nothing to the leaving nodes, although some changes seem to make the tree more balance. For example, when one of \( X \)'s child leaves the system, \( X \) does not attempt finding another user to fill the vacancy, because the cost is large. \( X \) does not know which user to choose. Also, the move of this chosen user may cost more move.

Additionally, we can combine the two algorithms, the reenter algorithm and the rotate algorithm together in order to achieve better performance.

**Algorithm 3 Rotate and Reenter for Node X**

for each child y of X:
Step 1. Calculate \( W_y, G_y, W_z, G_z \) for all \( z \in \{ i \mid i's \ parent \ is \ y \} \)
Step 2. Calculate the Potential Weight(\( PW_z \)) for all \( z \in \{ i \mid i's \ parent \ is \ y \} \)
Step 3. If exist \( z_i : PW_{z_i} < W_y \) and \( PW_{z_i} > PW_{z_j} \) for all \( j, i \neq j \)
   
   Then Rotate(\( z_i, y \))
Step 4. Update \( G \)
Step 5. BlockedTime = 0
Step 6. If cannot play, BlockedTime++
   
   else blockedTime = 0
Step 7. If BlockedTime \( \geq \) RT
   
   Reenter

The order of rotate and reenter does not matter. We can also run step 5-7 first and then run step 1-4.

**C. Prevent Cheating**

The conception of free-riders often has something to do with cheating. In the algorithm 2 and 3 mentioned above, if the \( W \) or \( G \) is misreported to their parents or grandparents, free-riders would take advantages of this algorithm. Therefore, the algorithm to calculate the \( W \), \( G \) and \( PW \) should make a little change.

We assume that only a few nodes cheating, therefore, the majority of nodes involved one calculation of node \( X \) is honesty. In addition, a cheating node both report their \( W \) or \( G \) to their parent and grandparent bigger than the fact. We compare the information of \( y \), say \( W_y, G_y \), which is reported by \( y \), with \( W'_y, G'_y \) which are computed by the information reported by \( y \)'s children. This information include their download rate and their \( W \) and \( G \). If \( W_y > W'_y \) or \( G_y > G'_y \), it indicates the \( y \) has large chance to cheat. If \( W_y < W'_y \) or \( G_y < G'_y \), it indicates that some children of \( y \) is cheating. In the first situation when \( y \) is cheating, we use \( W'_y \) instead of \( W_y \), \( G'_y \) instead of \( G_y \) to compute for the suitable rotation. In the second situation, it is hard for \( X \) to determine which one is cheating. However, when \( y \) runs the algorithm, it can decide which branch is cheating.

Free-riders sometimes modify the client to limit their upload speed or even to change their reputation functions. It is the users' parent or grandparent and their children who decide their QoS in our incentives. Therefore, it is...
of little use to modify the client software against the incentive mechanism.

V. SIMULATION AND DISCUSSION

In this part we run a series of simulation to show the effectiveness of incentive mechanism.

A. Environment Setup

We use C# to build up a simulation environment. This program runs time slot by time slot. In each time slot, first of all, a random number of users join the system. The number obeys Poisson distribution, and has an average number AveJoin. The new join user has different capacity of children, between $D_{\min}$ and $D_{\max}$ with uniform distribution. The user has probability of $P$ to become a free-rider. All users have the same buffer size of $T$ time slots. Then at each time slot, source and normal users deliver their content to their children. If there’s enough content to play, the user plays, otherwise it is blocked. The incentive method runs for every node $X$ once in each time slot. To avoid synchronous problem, the two users when one is the ancestor of the other cannot run the method together. One of them is locked until the other finishes. At the end of each time slot, every user with probability of $P_{\text{Leav}}$ leaves this system. Each experiment is run for 100 time slots. In algorithm 1 and algorithm 3, every user reenter the system after being blocked for $RT$ time slots. $RT$ should be larger than 1, because they are blocked at the time slot they join.

B. Results and Discussion

Fig. 3 shows the blocking ratio $R$ increases with free-rider probability $P$. Because of leaving, the results are very different from the static situation. When $P \leq 50\%$, algorithm 1 does a little improvement, but it is very limited. This time, Algorithm 2 and algorithm 3 provide much better improvements comparing with algorithm 1, and algorithm 3 is better than algorithm 2. When $P > 50\%$, algorithm 1 does not affect the system, and the algorithm 3 becomes worse. In general, algorithm 3 is the best in all situations. Because the new users cannot play the media, so when the free-rider probability is 0, still some users are blocked.
Fig. 4 shows the choice of reenter time $RT$. From this figure, we can see the blocking ratio $R$ increases when $RT$ goes up. The smaller reenter time means more frequent join messages. In practical, the reenter time may be around 4.

Fig. 5 indicates the blocking ratio $R$ changing along with the max children capacity $D_{\text{Max}}$. With rotation, the diversity of users upload capacity makes little impact, but it affects the system without rotation. With the same amount of user, larger capacity means large upload rate, so it is clear that the $R$ would drop. However, it is very interesting why the algorithm 2 shows the reversed results. It is because the capacity of free-riders also increases. As a result, more users might have a free-riker as their parent. Therefore, it is difficult for the normal users to rotate. Although the results of algorithm 2 are not very satisfying, they are still better than algorithm 1 or the results without incentives.

Fig. 6 shows that the blocking ratio $R$ decreases with the buffer size $T$. For the system without incentives and the system using algorithm 1, the improvements are obvious. With rotation, the improvements are much small, so with rotation the users do not need large buffer size.

Fig. 7 shows the relationship between blocking ratio $R$ and the average join rate $AveJoin$. Large average join rate means more users in the model. The systems without rotation grow worse dramatically, but the large join rate makes little influence to the system with rotation.

Fig. 8 shows the relationship between blocking ratio $R$ and the leaving probability $P_{\text{Leav}}$. When the leaving probability is high, none of these algorithms work. This figure also shows with a small leaving probability, the systems without rotation can be better while the systems with rotation goes worse. When leaving probability is 0, the results of system with no incentives are similar to the counterpart of the evaluation of the static situation.

In all, the system with rotation provides much better performance to the tree-based system. The combination of rotation and reenter has the best performance.

VI. CONCLUSION

Research of this paper is based on the tree-based live media streaming service. We propose an incentive mechanism using rotation. According to the results, the rotation algorithm can reduce the damage of free-riders a lot as well as make users with larger contribution less probability being blocked.

APPENDIX

Theorem 1: A tree with the least contribution weight if and only if (1) Do not exist node a and node b where $D_b > D_a$ but $H_b > H_a$; (2) Except the nodes with the largest height or height which is 1 smaller than the largest height of the tree, all nodes’ children number reach its maximum, that is $V_i = D_i$.

Proof: We give the proof by contradiction. For the ‘only if’ part, suppose there’s another tree with the least contribution weight but not satisfy condition (1). By changing this two nodes, and keep $D_b - D_a$ branches whose sum of $G$ denotes as $G'$ with node b together.

$$\Delta W = (D_b - D_a + G') \times (H_a - H_b) \leq 0$$

This is contradicted to this tree is of the least contribution weight.

If it does not satisfy condition (2), which means exist a node a and a node b, $H_a - H_b \geq 2$ and $V_b < D_b$. We simply make a become b’s child, since $V_b < D_b$, so there is still room for node a.

$$\Delta W = G_a \times (H_b + 1 - H_a) < 0$$

This is also contradicted to this tree is of the least contribution weight. Therefore, a tree with the least contribution weight must satisfy these two conditions.

For the ‘if’ part, first we prove when this conditions are held, the contribution weight is fixed. The contribution weight equals to the tree we order all nodes with degrees descendent and put it layer by layer. To not contradict the condition, the only changes we can make to the tree is to swap the node with the same height, or swap the nodes with the same degree, which make no difference to the contribution weight.

If there exist other methods to make the tree with less contribution weight, it should not hold these conditions. Proved in the ‘only if’ part that it is impossible.

Therefore, a tree has the least contribution weight if and only if it satisfies these two conditions.

Theorem 2: The structure with least contribution weight has the least potential damage when the free-riders are unknown.

Proof: From theorem 1, the tree with least contribution satisfies the two conditions: (1) Do not exist node a and node b where $D_b > D_a$ but $H_b > H_a$; (2) Except the nodes with the largest height or height which is 1 smaller than the largest height of the tree, all nodes’ children number reach its maximum, that is $V_i = D_i$. Also, we know the trees satisfying these two conditions have fixed contribution weight. If it does not has the least potential damage, in other words, if another tree which does not
hold the two conditions has less potential damage, we make similar change as we do in the theorem 1.

If condition (1) does not hold, we change this two nodes, and keep $D_a - D_a$ branches with node b together. For convenience, the set $S$ is used to refer the nodes in those branches and node b, $h$ denotes $H_b - H_a$. The difference between the previous potential damage and the counterpart after this change is followed:

$$\Delta = \sum_{i \in S} B'_i - B_i$$

$$= -\sum_{i \in S}((1 - P)^{H_i - h - 1} - (1 - P)^{H_i - 1})$$

$$< 0$$

This is contradicted to this tree has less potential damage.

If it does not satisfy condition (2), which means exist a node a and a node b, $H_a - H_b \geq 2$ and $V_b \leq D_b$. We simply make a become b’s child, since $V_b < D_a$, so there is still room for a. The difference between the previous potential damage and the counterpart after this change is followed:

$$\Delta = B'_a - B_a$$

$$= -(1 - P)^{H_a - 1} + (1 - P)^{H_b + 1 - 1}$$

$$< 0$$

This is also contradicted to this tree has less potential damage. Therefore, a tree with the least potential damage must satisfy these two conditions.

Also, the potential damage is fixed if the two conditions are held. The potential damage equals to the tree we order all nodes with degrees descendent and put it layer by layer. To not contradict the condition, the only changes we can make to the tree is to swap the nodes which can have the same number of children, which make no difference to the potential damage.

If there exist other methods to make the tree with less potential, it should not hold these conditions. According to the previous part of proof, it is impossible.

Therefore, the structure with least contribution weight has the least potential damage when the free-riders are unknown.

**Theorem 3:** The structure with least contribution weight has the least nodes blocked when free-riders are known.

**Proof:** The free-riders are considered as the nodes with $D = 0$. We define

$$D_{sum} = \sum_{i \notin \text{free-riders}} D_i$$

(13)

When the number of nodes $N$ is not greater than $1 + D_{sum}$, no users blocked if the tree has the least contribution weight, because all free-riders are placed at the leaves. Thus it is minimal. Otherwise, the blocked nodes are $N - D_{sum} - 1$. Every node i except free-riding can make at most $D_i$ nodes unblocked. Therefore, at most $1 + D_{sum}$ nodes are not blocked (the 1 is the source). In other words, at least $N - D_{sum} - 1$ nodes are blocked. Therefore, the structure with the least contribution weight has the least nodes blocked when free-riders are known.

**Acknowledgement**

We thank the anonymous reviewers for their helpful advices. This work was supported in part by NSFC under Grant No. 60702054, 863 program of China under Grant No. 2006AA01Z203, Shanghai Municipal R&D Foundation under Grant No. 07dz15004-1, the Shanghai Rising-Star Program under Grant No. 08QA14009, and State Key Laboratory of Integrated Service Networks under Grant No. ISN-9-06. Xin Wang is the corresponding author. We also thank the Chun-Tsung fund.

**References**


Shuang Yang was born in Chengdu, China, on October 20, 1987. She is currently an undergraduate student at Fudan University of Shanghai, China. She will receive her B.S. degree in computer science in 2010 expected.

Xin Wang is currently an Associate Professor in the School of Computer Science at Fudan University. He received his B.S. degree in information theory and M.S. degree in communication and electronic systems from Xidian University, China, in 1994 and 1997, respectively. He received his Ph.D. degree in computer science from Shizuoka University, Japan, in 2002. His research interests include wireless networks, peer-to-peer networks, and network coding.
A programmable architecture for layered multimedia streams in IPv6 networks

Brendan McAllister, Alan J. Marshall and Roger F. Woods
School of Electronics, Electrical Engineering and Computer Science, Queen’s University Belfast
Email: bpmcallister@btinternet.com, {a.marshall, r.woods}@qub.ac.uk

Abstract—A new configurable architecture is presented that offers multiple levels of video playback by accommodating variable levels of network utilization and bandwidth. By utilizing scalable MPEG-4 encoding at the network edge and using specific video delivery protocols, media streaming components are merged to fully optimize video playback for IPv6 networks, thus improving QoS. This is achieved by introducing “programmable network functionality” (PNF) which splits layered video transmission and distributes it evenly over available bandwidth, reducing packet loss and delay caused by out-of-profile DiffServ classes. An FPGA design is given which gives improved performance, e.g. link utilization, end-to-end delay, and that during congestion, improves on-time delivery of video frames by up to 80% when compared to “static” DiffServ.

Index Terms— Communication system performance, communication system routing, teleconferencing.

I. INTRODUCTION

There has been an increased interest in using Internet Protocol (IP) for multimedia streaming applications such as Skype™ videotelephony, web-based CCTV video systems and video conferencing. However the basic IP transport service cannot offer any connection guarantees resulting in delayed or even lost packets and causing screen freezes and frame errors at the destination when delays exceed 400ms [1]. Scalable video coding [2] can be used to address this, either by layering video in a multicast environment based on the receiver subscribing to a number of layers depending on available bandwidth [3], or by integrating layered video into a retransmission strategy based on the importance of the layer and its expiry time [4]. This offers a stepped approach to video quality by allowing the decoder to decode a subset of the total bitstream to give a reduced resolution [5], or display the information at a reduced frame rate. Fine-granular scalability (FGS) provides base and single enhancement layers and uses bit plane encoding to offer a range of bandwidths thereby allowing a streaming server to send a desired amount of enhancement layer information based on network bandwidth and receiver processing capability without re-encoding [6].

Differentiated Services (DiffServ) has emerged as the favored Quality of Service (QoS) method as it offers a scalable hands-off approach to congestion management by allocating higher bandwidth levels to voice/video with high sensitivity to delay and jitter, than that allocated to HTTP/FTP traffic [7]. As DiffServ offers throughput guarantees based on a per-class basis [8] (Table I), within a router each class may receive more traffic than it can send within its bandwidth allocation, resulting in congestion, and high priority video traffic being delayed/dropped.

A programmable network functionality (PNF) is presented which works together with both current DiffServ QoS mechanisms and layered MPEG-4 transmission, to allow uninterrupted video playback. It extends work in interactive video streaming and content delivery [9], increasing performance by offering individual flow processing within aggregated DiffServ classes. QoS is improved for interactive video traffic by utilizing scalable MPEG-4 streaming at the network edge and specific video delivery protocols. Using a scalable MPEG-4 stream, the PNF approach prioritizes this traffic within a DiffServ environment when congestion occurs using a novel ‘splitting and recombining’ flow technique which splits a layered stream at a point of congestion and then reroutes the enhancement layers to an alternative link or remarks it to a lower priority class thereby reducing congestion for the high priority base layer traffic over the heavily utilized link. Advantages of this approach include:

- **No additional network protocols:** The PNF extends DiffServ using the current TCP/UDP over IP protocols using the hop-by-hop extension header in IPv6.
- **Compatibility with current technology:** A router not enabled with PNF functionality will ignore the processing that the packet has requested.

PNF needs only localised programmability and is invoked using a queue trigger mechanism indicating when queue sizes exceed a pre-configured level. Unlike [6] & [10], no server/streaming technology is required within the network in order to utilize the layered encoding; and it offers varied levels of video playback at the receiver under increasing congestion levels.
The paper is organized as follows; the first section describes existing DiffServ and scalable video; in section III, the PNF is described, and performance indicators when compared to conventional DiffServ are given in section IV; some details for a Xilinx Virtex 4 FPGA implementation are given in section V.

II BACKGROUND

A. Differentiated Services

DiffServ allocates higher bandwidth to video (and voice) thereby providing higher sensitivity to delay and jitter than http/ftp traffic. The two additional basic service classes that accompany this best-effort (BE) service are expedited forwarding (EF) [11] and assured forwarding (AF) [12]. EF is designed for low bandwidth-high delay and packet loss sensitive traffic e.g. VoIP and is increasingly used to obtain the end-to-end delay requirement of 200ms [1]. The basic AF class hierarchy is split into 4 sub-classes with each allocated a bandwidth percentage according to priority and level of the traffic. Each AF class packet can be also marked with low, medium or high drop precedence (DP) which matches well to priority levels in layered video streaming, namely base, spatial and temporal.

It is generally accepted that the ISP allocates EF, AF and BE services (as illustrated in [12] for the Cisco-based DiffServ node) and the recommended different functionality that offers end-to-end QoS capabilities [13]. The EF queue is serviced by: a priority queue (PQ) scheduler, offering admission control to prevent it from exceeding this allocated bandwidth; a weighted fair queue (WFQ) scheduler servicing four AF class queues and; a Best Effort (BE) queue. These are policed using the weighted random early detection (WRED) offering admission control for all other managed queues.

The disadvantage of DiffServ is that no specific end-to-end guarantees can be given to a traffic flow and because each class has guaranteed QoS, a class can get congested and incoming excess traffic can get out-of-profile. Typical DiffServ implementations currently shape the traffic (delaying it) or drop out-of-profile traffic. It is this uncertainty that is overcome through the use of the treatments available within the PNF.

B. Layered MPEG-4

The group of video (GoV) for non-layered and layered video streams is shown in Figure 1. For layered video, a high priority base layer which is always needed, provides lower quality video playback at 10fps, a spatially enhanced layer gives high quality video at 10 fps and a temporally enhanced video layer provides high quality video at 30 fps. The non-layered GoV consists of a single stream with one full frame (I frame) every 12 frames; for the layered GoV, base and spatial layers have an I frame every 12 frames with the temporal layer being totally made up of predictive frames (P frames), resulting in a GoV of 32 frames between successive I frames.

Using layered video, this higher priority base layer can be prioritised to ensure uninterrupted video playback. With DiffServ, interactive video traffic is classified into the AF4 aggregated flow [13] as it has low delay and packet loss requirements. By assigning different levels of drop precedence to each video layer, all layers are delivered via the same high-priority aggregated flow but with differing probabilities of being dropped. If required, enhancement layers’ packets can be dropped before vital base layer packets depending on the WRED configuration [9]. All non-layered video stream traffic is assigned to the AF41 class as video playback cannot commence unless all this traffic is received [2].

III PROGRAMMABLE NETWORK FUNCTIONALITY

The PNF is built on top of the DiffServ setup discussed in Section II; this allows the standards in place for DiffServ QoS to be available and to permit retrofitting of current hardware. From Fig. 2(a), the PNF monitors the queue levels in the primary link and secondary link to make an intelligent decision on whether additional bandwidth is available on an alternative queue/link. It only does this if available resources are present thus ensuring that other flows and classes are not affected. The flows that are tagged for processing use the IPv6 hop-by-hop option extension header, to tag IPv6 video flow packets with the required information. Each data field plays a pivotal role in providing the PNF functionality:

- **Split Location**: Indicates which layers of current interactive video flow are eligible for processing within the current flow.
- **Layer ID**: Identifies the video layer to which the current packet belongs.
- **Org DSCP**: Contains original DSCP packet marking thereby allowing marking back to original value once the congestion has been eliminated over a link.
- **Treatment**: Highlights which of the 4 treatments are currently available for PNF, indicating the preferred treatment requested by the packet.

Using the hop-by-hop option, the packet can be processed by the PNF-enabled nodes. A novel packet processing scheme is described for interactive video which gives guaranteed video playback for heavy network congestion.
The PNF checks if the primary class is congested and forwards traffic according to the split location field (base → 1, spatial → 2, temporal → 3) and current packet layer id fields (1, 2, 3) in the hop-by-hop extension header. Four specific processing treatments are identified for interactive video traffic:

1. **Remark** treatment remarks enhancement layer traffic to a lower class within the DiffServ node (Fig. 2(b)) in the face of congestion, freeing high priority (AF4) queue for the more timely delivery of base layer video. This is done on a 'hop-by-hop' basis and allows remarking once.

2. **Reroute** treatment reroutes enhancement layer traffic away from a primary link under congestion to an alternative link thereby freeing bandwidth in the congested link (Fig. 2(a)), assuming a pre-defined secondary link is available as the backup link (otherwise it is not activated). This uses available network bandwidth to deliver enhancement layer traffic in a more timely fashion than 'static DiffServ' and also gives a reduction in packet loss due to WRED (Fig. 6) or queue tail dropping.

3. **Remark/Reroute** attempts to remark enhancement layer traffic if bandwidth is available in a lower DiffServ class over the primary route before attempting to reroute it.

4. **Reroute/Remark** attempts to reroute before remarking the enhancement layer traffic to a lower class at times of congestion.

The IPv6 hop-by-hop option extension header is used to encapsulate the information required by the PNF to process the interactive video flow packets. When congestion occurs, the PNF detects when the throughput approaches or exceeds the class bandwidth allocation within a link, and then remarks the higher video layers to a lower class or reroutes them to an alternative link. The steps taken are outlined in the flow chart (Fig. 3). If there is no congestion in the AF4 DiffServ class, then all interactive video traffic traverses through each class using a token bucket and invokes when the tokens for a specific class are lower than a pre-configured level. As the PNF is based on a hop-by-hop basis, recombination will occur in the case of the remarking treatment at the next hop on the path. For rerouting, this will occur once the traffic has passed the congested link. The Real Time Protocol (RTP) can ensure that packets from the same stream reaching the destination are played in the correct order.
IV SIMULATION PERFORMANCE RESULTS

A typical network (Fig. 4) was implemented in OPNET™ modeller to analyze the interactive video performance using a head and shoulders video stream, symbolic of typical video conference. This was generated from the ‘foreman’ video test sequence and consisted of a 300 frame, 30s QCIF (176x144) cycle [14]. Using the Microsoft Visual Reference MPEG-4 encoder, the stream was compressed into non-layered and layered streams detailed in [15] as detailed in Table II.

WRED was configured to drop all incoming traffic when the resultant delay increases beyond 400ms, as this is double the network delay requirements of interactive video. This occurs when the queue increases beyond a level at which the router can service it within the required time. The maximum queue size \( Q_{\text{MAX}} \) is calculated as,
where LDR is the link data rate in bps, CBA is the class bandwidth allocation for the class in question given as a percentage, MTU is the maximum transmission size of a packet in bytes and D is the maximum delay in (s) allocated within the node. The probability that a packet is dropped $P_d$ will depend on the drop precedence of the packet and the prevailing size of queue. The way that WRED is configured by the user, has a strong influence on throughput response. The configuration in Fig. 5 has been used here as it ensures all lower precedence values are dropped before higher values, thus ensuring the high priority base layer video traffic will be dropped last.

\[
Q_{max} = \frac{LDR \times CBA}{100 \times MTU} \times D
\]

MULTIPLE LAYERED INTERACTIVE VIDEO STREAMS WERE CHOSEN AS THEY REPRESENT WORST-CASE SCENARIOS WHICH ARE UNLIKELY TO OCCUR IN A RANDOMLY GENERATED NETWORK SIMULATION, OFFERING A RANGE OF VIDEO SESSIONS GIVING 4 CONCURRENT FLOWS OVer AN INDIVIDUAL LINK AS ILLUSTRATED BELOW.

Each of the routers A-F are PNF-enabled but only those which see increased network congestion will switch on and perform additional processing. Three worst-case scenarios have been simulated and include:

1. Independent links congestion (D→E & B→G), analyzing the effect of video flows which may or may not traverse the effected links,
2. Parallel link congestion (D→E & G→F), essentially blocking both links to a localized area of the network,
3. Centrally located links (C→D & G→D) that are common to multiple flows and used increasingly for rerouting at time of congestion.

The QoS improvement is measured in terms of end-to-end delay and the timely delivery (the % of traffic reaching the destination within the end-to-end delay target (0.2s in this case)) of the base and enhancement layer video streams to all destinations. All treatments available were simulated and compared to static DiffServ. Background traffic is generated at up to 80% over the selected links and created concurrently over each of the links under consideration, so any correlation between the links and video performance can be examined. In all cases, the first link is set to a specified congestion level (e.g. 3Mbps (60%)) i.e. normal link under no congestion while the congestion over the second link is ramped from 1.5Mbps (30%) to 4Mbps (80%). This is necessary to show the video performance in terms of the relevant congested link as well as any effects that the second link congestion may have on the flow which without any PNF traffic management, would travel over the primary link.

<table>
<thead>
<tr>
<th>Video session</th>
<th>Source</th>
<th>Destination</th>
<th>Route</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>G</td>
<td>A→B→G</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>C</td>
<td>B→A→C</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>E</td>
<td>C→D→E</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>F</td>
<td>D→E→F</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
<td>F</td>
<td>A→B→G→F</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>D</td>
<td>B→G→D</td>
</tr>
<tr>
<td>7</td>
<td>G</td>
<td>E</td>
<td>G→D→E</td>
</tr>
<tr>
<td>8</td>
<td>A</td>
<td>E</td>
<td>A→C→D→E</td>
</tr>
<tr>
<td>9</td>
<td>C</td>
<td>D</td>
<td>C→D</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
<td>F</td>
<td>B→G→F</td>
</tr>
<tr>
<td>11</td>
<td>C</td>
<td>F</td>
<td>C→G→F</td>
</tr>
</tbody>
</table>

**1) Independent links (D→E & B→G)**

Various delays for links in the topology shown in Fig. 4 are presented in Figs. 6 and 7. Increasing congestion over the D→E link which has four concurrent video sources traversing it, causes the end-to-end delay of the vital base layer to exceed the 0.2s delay limit bound at 60% background traffic congestion (Fig. 7). Both remark treatments guarantee 100% base layer timely delivery within the delay bounds up to 70% background traffic. It is however, the reroute treatments which can increase the timely delivery to almost 100% at up to 80% background traffic congestion. Beyond this point, all treatments undergo significant reduced timely delivery, as there is no longer enough bandwidth to support 4 base layer video flows. The enhancement layer end-to-end delay and timely delivery show the increased performance as a result of the PNF, allowing the end-to-end delay of 0.2s to be achieved by dropping any traffic which cannot meet this delay bound resulting in an increased timely delivery using all treatments.

![Figure 5. PNF flow chart progression](image-url)
The enhancement layer end-to-end delay and timely delivery show the increased performance as a result of the PNF, allowing the end-to-end delay of 0.2s to be achieved by dropping any traffic which cannot meet this delay bound, resulting in an increased timely delivery using all treatments.

The most improved approach is reroute where almost 90% is achieved illustrating the PNF’s ability to utilize all available bandwidth within the network at times of congestion. With greater bandwidth available on the B→G link, its ability to support four concurrent flows is improved. The end-to-end delay encountered with no functionality treatment does not exceed 0.2s until 67% of background traffic is present (Fig. 6(a)). Once again, the remark treatments increase the ability to meet this delay bound up until 73%, guaranteeing almost 100% timely delivery up to 80% background traffic congestion (Fig. 6(c)). The reroute treatments can still deliver over 90% of the base layer ensuring much smoother playback than that with no treatment which can only deliver below 20% in a timely fashion. The improvement in enhancement layer delivery is also clear, particularly with the reroute treatments which can deliver upwards of 90% of it at 80% background traffic congestion illustrated in Fig. 6(d). This further illustrates that the use of available resources elsewhere significantly helps improve the video quality due to improved enhancement layer delivery.

As congestion increases, the PNF significantly alters the link utilization over affected links where the reroute treatments move 20% of the video traffic from the D→E link to the D→G link, ensuring it is not excessively delayed causing untimely delivery. The end-to-end delay encountered by all traffic not common to either link D→E or B→G, shows no noticeable effect on the rerouting of traffic over alternative links; this is because all traffic, base and enhancement layer, is delivered in a timely fashion. Overall, the functionality offered substantial improvement and could guarantee 80% of the vital base layer traffic right up to 80% background link congestion.

1) Parallel link congestion (D→E & G→F)

For flows traversing D→E in a parallel link scenario, the base layer end-to-end and timely delivery results are comparable to that expected from the independent link congestion scenario, where the PNF can guarantee 100% up to 70% link congestion. The enhancement layer end-to-end and timely delivery however, illustrates significant effects due to the presence of increasing congestion over the second link. With increasing G→F congestion even at low D→E congestion (Fig. 7(a)), there is an increased end-to-end delay for enhancement layer delivery. Initially, this does not affect enhancement layer delivery (Fig. 7(b)) but as D→E congestion increases, the delivery drops to as low as 50% for the reroute treatments.
In comparison to no treatment, this is an improvement and illustrates how the approach can continue to protect the base layer of flows native to links which are used as an alternative link for rerouting enhancement layer traffic.

The same effect is seen for flows native to the link G→F in Figs. 7(c) and (d). As congestion increases over G→F, more enhancement traffic is routed over G→D→E, causing it to suffer increased delay as D→E congestion increases (Fig. 7(c)) and leading to reduced timely delivery (Fig. 7(d)). All treatments available within the PNF, deliver over 80% of the vital base layer traffic at over 80% link congestion. With static DiffServ alone, less than 10% can be delivered in a timely fashion. The effect of blocking a localized area of the network through parallel link congestion is shown in Fig. 7(e) and (f), where rerouted traffic from D→E is sent over G→F (via D→G) causing the increased throughput spikes. PNF, however, only accepts the level of enhancement layer traffic that it can deliver within the delay bound while always ensuring the native base layer traffic of the link is given priority over this rerouted traffic.

In this scenario, clear correlation is seen with two points of congestion as they ‘block’ a localized area of the network causing rerouted enhancement traffic to be affected by congestion in the second link. A significantly
higher level of vital base layer delivery is still guaranteed which remains unaffected by rerouted enhancement layer traffic as this traffic is dropped to protect the base layer traffic native to a link as shown in Figs. 7(b) and 7(d).

2) Central link congestion (C→D & G→D)
This scenario shows the effect of central mesh network congestion on the edge destination locations of the interactive video. Fig 8 shows the results for link C→D, and similar results were obtained for link G→D. With more flexibility in the central network, the improvement due to PNF can be seen as the base layer timely delivery is maintained at 100% for the reroute treatments for C→D congestion and for G→D congestion, even with 80% congestion on either link. The approach provides better distribution of enhancement traffic thereby giving 100% timely delivery even at 80% background link utilization. An increased end-to-end delay bound was observed for enhancement layer traffic for increasing G→D congestion, but still remained well within the required limits. In this scenario, the PNF reroute treatment provides full utilization of the network resources delivering 100% of the base and enhancement layers in a timely fashion, compared to less than 10%, with no treatment, i.e. “Static DiffServ”. This serves to show that in a central network mesh, the PNF is even more robust to network congestion.

**II. OPTIMIZED FPGA IMPLEMENTATION**

Network processors have an inherent limitation in their ability to only process 1 thread per processing core but Field Programmable Gate Arrays (FPGAs) can provide true synchronized parallel processing through optimized design. The PNF operates above the network level in the switch requiring minimal communication with the output blades with congestion and queue triggers indicating a high queue size as shown in Fig. 9. Fig 10(a) shows the PNF top level architecture illustrating the 3 main blocks; the Queue Status Update block analyzes the queue data written to the Distributed RAM memory and relevant packet processing attributes (treatment type, packet layer info) and performs the programmable processing to minimize congestion and prevent excessive video delay.

**A) PNF architecture**
The first key stage is to detect congestion. In order to prevent excessive flapping caused by multiple bursts of network traffic, a hysteretic loop is employed to ensure the functionality will remain enabled until the current delay within the DiffServ node has dropped significantly. Simulation has shown that such flapping is reduced by 50%. The specific queue size triggers which will correlate to the delay as illustrated can be calculated as below:
\[ Q = \frac{D \cdot CBA}{MTU} \]  

(2)

where \( Q \) is the queue size in packets, \( D \) is the maximum allowable video end-to-end delay in seconds, \( CBA \) is the class bandwidth allocation for a specific class in bit/s and \( MTU \) is the maximum transmission unit (packet size) in bits. The functionality disable threshold has been configured to 60% of this queue size i.e. a maximum delay of 0.12s. The approach reduces signaling between output ports and the switch backplane when compared to the instantaneous queue update technique but offers greater accuracy as the queue size is refreshed at a lower rate.

1) Queue Status Update block
   The Queue Status Update block in Fig 10(b) receives ‘queue status signals’ from the queue monitoring architecture (Fig. 9 and Fig. 10(d)), indicating that the queue status of one of the output ports has crossed the pre-configured trigger level. Fig. 10(b) shows the architecture for the Queue Status Update block which includes an address selector to generate a 6-bit address. The top 2 bits are used to select the correct distributed RAM, and set the relevant write-enables (WE), whilst the bottom 4 bits (WRADDR) are used as an address. This address and write enable data is then used to address the queue data (DOUT) when written to the Distributed RAM memory in (Fig. 10(a)).

1) Distributed RAM sub-block and addressing
   The only data storage required is the 1-bit congested/non-congested indicator for each queue within each link. For an 8-link, 5-class router, a 40-bit memory is used for the queue trigger technique, implemented as 3 of 16x1D distributed RAM blocks. Each link requires five locations, one for each DiffServ class indicating if it is congested or not. Writing/reading of the RAM requires knowledge of the location of each congestion indicator, computed within the relevant sub-blocks.

2) Packet processing sub-block
   The packet processing sub-block determines the link and class over which the layered video traffic is transmitted by reading the queue status signals from memory and writing them into registers for immediate access by the link selector functionality which uses them, along with the packet relevant inputs, to process functionality-enabled packets.

3) Queue Monitoring Functionality block
   As shown in Fig. 9, the PNF needs a minimal level of communication between output queues and the functionality itself. A block is used to track the queue triggers and if the status of a queue changes (indicated through one of the queue trigger inputs), it sends the change in status to the PNF along with the link and class identification. If the status of a queue changes, this block registers this by comparing the input queue status to a previously registered value and sending a status update to the PNF along with link and class identification.

Figure 9. Communication channels for PNF and queue monitoring

Figure 10. PNF architecture and schematics
B) PNF Hardware Performance

Packet processing comprises reading of the distributed RAMs for the relevant link and class status for the current packet being processed. The ‘Addr Sel’ in Fig. 10(c) indicates each location to be read. Ten locations have to be read for each packet as the current status for all classes (5 in total) for both the primary and pre-defined alternative link is needed. Due to memory read latency, the ability to pipeline the functionality is limited, resulting in each packet taking 15 clock cycles to be serviced. Assuming a 100MHz clock, the packets processed per second of the PNF can reach 66.7Mpkts/s.

When implemented on a Xilinx XC4VLX15 Virtex4 FPGA device, the PNF functionality only required 6.2% DSP slices, 2% registers and 1.5% LUTs, allowing over 90% of the overall FPGA area to remain for further functionality expansion; the latency remains well within the requirements for a hi-speed network router.

To assess feasibility of the queue trigger technique within a packet switch, the bit rate which can be serviced was calculated. For a minimum packet size of 512 bytes, a throughput rate of 33.8Gbit/s is found to be achievable. Assuming Ethernet traffic for a maximum packet size of 1518 bytes, the throughput rate can reach 801.5Gbit/s. A throughput rate of over 33Gbit/s is achievable for a minimum, average and maximum packet size of 512, 1000 bytes and 1518 bytes respectively.

III. CONCLUSIONS

A novel approach is presented for a programmable stepped approach to video quality using a layered MPEG-4 bit-stream.. Using remarking or rerouting, enhancement layer traffic is shifted away from the high priority AF4 class and through lower priority classes, thus giving a stepped improvement in video quality over DiffServ and a guaranteed delivery of the base layer traffic. This was achieved by ensuring that at minimum, the base layer traffic containing the vital I-frames of the MPEG-4 stream were received before enhancement layer traffic through the use of the PNF. The functionality requires minimal communication with the output links and is enabled only when it reaches pre-configured congestion levels. Queue monitoring techniques were introduced at the queues along with a communication channel for the main PNF located within the backplane and a queue trigger technique developed.

ACKNOWLEDGMENT

The authors acknowledge funding provided by Xilinx Inc. Ca., USA and the Department of Education and Learning and guidance from Gordon Brebner, Philip-James Roxby and Wilson Chung from Xilinx Inc.

REFERENCES


Brendan McAllister received the M.Eng. and Ph.D. degrees in Electrical and Electronic Engineering from Queen’s University Belfast, UK. He is currently working with First Derivatives plc.

Alan J. Marshall is currently professor of Telecommunications Engineering at Queens University Belfast. He received the BSc from University of Ulster and PhD from Aberdeen University. His research interests include network architectures and programmable networking concepts.

Roger F. Woods received the B. Sc. and Ph.D. degrees in Electrical and Electronic Engineering from Queen’s University Belfast, UK. He is currently a professor of Digital Systems at the same university with interests in programmable hardware and high level design tools.
Novel Stream Cipher Using SCAN and Variable Ordered Recursive CA Substitutions and Its DSP+FPGA Implementation

Rong-Jian Chen\textsuperscript{a,*}, Jui-Lin Lai\textsuperscript{a}, and Shi-Jinn Horng\textsuperscript{a,b}
\textsuperscript{a} Department of Electronic Engineering, National United University, Taiwan
rjchen@nuu.edu.tw, jllai@nuu.edu.tw
\textsuperscript{b} Department of Computer Science and Information Engineering, National Taiwan Univ. of Sci. and Tech., Taiwan
horngsj@yahoo.com.tw

Abstract—This paper presents a new stream cipher for data security, which is based on permutation of the data and replacement of the data values. Permutation is done by scan patterns generated by the SCAN approach. The replacement of data values using variable ordered recursive cellular automata (CA) substitutions. To achieve this aim, an encryption-specific SCAN technique was firstly developed, 2-D hybrid CA was next built, and then 1\textsuperscript{st}-ordered and 2\textsuperscript{nd}-ordered generalized CA transforms were introduced to build variable ordered recursive CA substitutions. The proposed stream cipher satisfies the properties of confusion and diffusion because of characteristics of the SCAN and the CA substitutions are flexible. Moreover, the characteristics of the proposed stream cipher are loss-less, symmetric private key, very large number of security keys (number of possible security keys is more than $10^{1768} \sim 10^{4783}$ - according to the size of the 2-D von Neumann CA), and key-dependent pixel value replacement. Experimental results obtained using some color images clearly demonstrate the strong performance of the proposed stream cipher. This paper also shows the DSP+FPGA implementation of the proposed stream cipher for the real-time image security.

Index Terms—Stream cipher, SCAN, Variable ordered recursive CA substitutions, DSP+FPGA implementation

I. INTRODUCTION

With the ever-increasing growth of multimedia applications, data security is an important issue in communication and storage of data, and encryption is one the ways to ensure security. Data security has applications in inter-net communication, multimedia systems, medical imaging, telemedicine, and military communication. There already exist several methods of data security. They include SCAN-based methods [1-5], chaos-based methods [6-8], tree structure-based methods [9-11], and other miscellaneous methods [12-15]. However, each of them has its strength and weakness in terms of security level, speed, and resulting stream size metrics. We therefore proposed a new method of data security to overcome these problems. The proposed data security belongs to stream cipher which encryption method is based on permutation of the data and replacement of the data values. Permutation is done by scan patterns generated by the SCAN approach, the SCAN approach described in [5] is used because it produces a high volume of scan pattern. The data values are replaced using the variable ordered recursive CA substitution with a sequence of CA data that is generated from 2-D hybrid CA with special evolution rules. The advantages of CA in the proposed data security are described as follows.

(1) CA has been applied successfully to several physical systems, processes, and scientific problems that involve local interactions as in image processing [17], data encryption [18, 19], byte error correcting code [20]; it has also been used in pseudorandom number generators for VLSI built-in self-test [21]. (2) Number of CA evolution rules is very large. Hence, many techniques are available for producing a sequence of CA data that is generated from 2-D hybrid CA with special evolution rules. The advantages of CA in the proposed data security are described as follows. (1) CA has been applied successfully to several physical systems, processes, and scientific problems that involve local interactions as in image processing [17], data encryption [18, 19], byte error correcting code [20]; it has also been used in pseudorandom number generators for VLSI built-in self-test [21]. (2) Number of CA evolution rules is very large. Hence, many techniques are available for producing a sequence of CA data for encrypting and decrypting data. (3) Recursive CA substitution only requires integer arithmetic and/or logic operations simplifying the computation.

The proposed data security also belongs to the case of the general framework called iterated product cipher [4, 22, 23], which is based on repeated and intertwined application of permutation and substitution. This
general framework has been extensively studied and developed in terms of cryptographic strengths and attacks [4] and forms the basis of many modern encryption methods, including Data Encryption Standard [23], Advanced Encryption Standard [24], and chaos-based methods [6-8]. The proposed image security method differs markedly from that used elsewhere [18]. In this work, hybrid 2-D von Neumann CA was used to generate a high-quality random sequence as key-stream, with recursive CA substitution in the encryption and decryption schemes, such that the proposed image security system was secure. The cipher systems in the cited study [18] are affine and based on 1-D CA, and the encryption and the decryption schemes in [18] are non-recursive. Another study [25] showed that affine cipher systems are insecure. The proposed data security is an extended and improved version of that presented in cited study [26, 36] because the proposed one used SCAN techniques and 20 groups of CA recursive substitution to make the exhaustive searching attack much harder and to enhance the performances of system. This new data security proposed herein has additional features, such as key-dependent permutation, key-dependent pixel value replacement, very large key space, keys of variable length, and encryption of larger blocks. Moreover, it is a symmetric private key security system, meaning that the same keys are needed for encryption and decryption. Therefore, both sender and receiver must know the keys.

After well development of the proposed stream cipher, we then developed SCAN-CA-based image security system to illustrate the performance of the proposed stream cipher. Moreover, we built hardware/software implementation of the SCAN-CA-based image security system to satisfy the real-time requirement due to real-time image processing systems are widely used in many fields, such as industry, military, medical image processing and so on. For real-time image processing system, it needs high speed because of mass image data, so we can use DSP to solve this problem. On the other hand, FPGA has capable of flexible logic control, large memory and fast executing speed. So the real-time image processing system can be constituted by the combination of DSP and FPGA. In order to satisfy the real-time demand, we established hardware/software implementation of the SCAN-CA-based image security system within an embedded platform that includes a DSP and FPGA where DSP is SCAN processing unit and FPGA is the processing unit of variable ordered recursive CA substitutions.

This paper is organized as follows. Section II describes SCAN approach and 2-D hybrid CA, Section III illustrates SCAN-CA-based image security system using the proposed stream cipher. Section IV presents the possible keys and the cryptanalysis. Software simulation results of the SCAN-CA-based image security system are drawn in Section V; moreover, we also show the DSP+FPGA implementation of the proposed SCAN-CA-based image security system in Section V. Finally, section VI gives the discussions and conclusions.

II. SCAN AND 2-D HYBRID CA

A. SCAN Approach

Scanning of a 2-D array is an order where each element of the array is accessed exactly once. In other words, scanning of a 2-D array is a permutation of the array elements. Thus, scanning of a 2-D array $F_{MN} = \{f(i,j) : 0 \leq i \leq M - 1, 0 \leq j \leq N - 1\}$ is a mapping function from $F_{MN}$ to the set $\{g(l) : 0 \leq l \leq (M \times N - 1)\}$. Note that an $M \times N$ array has $(M \times N)!$ scanning paths.

![figure1.png](attachment:figure1.png)

Figure 1. Basic scan patterns

![figure2.png](attachment:figure2.png)

Figure 2. Transformations with partition

The SCAN represents a family of formal language-based 2-D spatial accessing approaches, which can represent and generate a large number of scanning paths systematically. There are several versions of SCAN language, such as Simple SCAN, Extended SCAN, and Generalized SCAN; each of them can represent and generate a specific set of scanning paths. Each SCAN language is defined by a grammar. Each language has a set of basic scan patterns, a set of transformations of scan patterns, and a set of rules to
is the start symbol, and production rules follow. An encryption-specific SCAN language is described as partition patterns were divided into 3 groups: rotation. 7: Vertical reflection. 6: Horizontal reflection. operations (identity, horizontal reflection, vertical reflection, rotation by 90°, 180°, and 270°) and their combinations. The rules for building complex scan patterns from simple scan patterns are specified by the production rules of the grammar of each specific language. Readers are referred to [1-5] for a detailed description of syntax and semantics of SCAN languages and their applications.

In the proposed encryption method, the scanning patterns are used as the encryption keys to rearrange pixels of the image. The scanning patterns are generated by an encryption-specific SCAN language [16] which is formally defined by the grammar

$$H = \{ \Gamma, A, \Pi \}$$

where \( \Gamma = \{ A, S, P, U, V, T \} \) are non-terminal symbols, \( \Sigma = \{ r, c, d, l, a, i, t, w, B, Z, X, (, ) \} \) space, 0, 1, 2, 3, 4, 5, 6, 7\} are terminal symbols, A is the start symbol, and production rules \( \Pi \) are given by

$$A \rightarrow S | P \quad S \rightarrow UT \quad P \rightarrow VT(A A A A) \quad U \rightarrow r | c | d | l | a | i | t | w \quad V \rightarrow B | Z | X \quad T \rightarrow 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7$$

The semantics of the encryption-specific SCAN language is described as follows. \( A \rightarrow S | P \): Process the region by scan S or partition P. \( S \rightarrow UT \): Scan the region with scan pattern U and transformation T. Each of scan patterns has eight transformations (Figure 2) which are defined as: 0: Identity. 1: Horizontal reflection. 2: 90° clockwise rotation. 3: 90° clockwise rotation followed by vertical reflection. 4: 180° clockwise rotation. 5: Vertical reflection. 6: 270° clockwise rotation. 7: 90° clockwise rotation followed by horizontal reflection. \( P \rightarrow VT(A A A A) \): Partition the region with partition pattern V and transformation T, and process each of the four sub-regions in partition order using A as from left to right. Figure 2 shows that partition patterns were divided into 3 groups: B, Z, and X, each group has eight transformations as that of scan patterns.

$$U \rightarrow r | c | d | l | a | i | t | w \quad \text{Scan with a specific scan pattern as in Figure 1, the letters } r, c, d, l, a, i, t, \text{ and } w \text{ in Fig. 1 indicate the type of scan patterns}.$$

Figure 2 shows that partition patterns were divided into 3 groups: B, Z, and X, each group has eight transformations as that of scan patterns. The image is first partitioned into four sub-regions using \( X3 \) partition order. These four sub-regions are scanned using \( c2, Z2(d0 \ w2 \ a4 \ l1) \), \( r0 \), and \( i4 \). The second sub-region is further partitioned into four sub-regions using \( Z2 \) order and these four sub-regions are scanned using \( d0, w2, a4, \) and \( l1 \).

Figure 3 shows the scanning path of the scan pattern \( X3(c2 \ Z2(d0 \ w2 \ a4 \ l1) \), \( r0 \) \( i4 \) for a \( 16 \times 16 \) image. The

\[ \text{Figure 3. Example of the scan pattern } X3(c2 \ Z2(d0 \ w2 \ a4 \ l1) \), \( r0 \) \( i4 \) \]

B. 2-D Hybrid CA

Cellular automata are dynamic systems in which space and time are discrete. The cells, as arranged in a regular lattice structure, have a finite number of states. These states are updated synchronously according to a specified local rule of neighborhood interaction. The neighborhood of a cell refers to the cell and some or all of its immediately adjacent cells. In 2-D CA space, the specified node \( P \), with its four nearest neighbors form the von Neumann neighborhood. Figure 4a shows the 2-D von Neumann CA space. The state of the given node at time step \( (t+1) \) will be determined from the states of nodes within its neighborhood at time step \( t \). Using a specified rule, the states are updated synchronously in time steps for all cells. Let \( a(i, j) \), represent the state of \( (i,j) \text{th cell at time } t \), whose von Neumann neighborhoods are in the states: \( a(i-1, j) \), \( a(i, j-1) \), \( a(i+1, j) \), and \( a(i, j+1) \). Then the rule of 2-D von Neumann CA evolution way can be expressed as

\[ a(i, j)_{t+1} = F(a(i+1, j), a(i, j-1), a(i, j+1), a(i-1, j)) \quad (1) \]

where \( F \) is a Boolean function that defines the rule. Readers are referred to [17-21] for a detailed description of 2-D von Neumann CA.

The hardware implementation of Eq. (1) for 1-bit 2-D von Neumann CA is shown in Figure 4b. Such a structure is referred as a programmable additive CA (PACA) due to it is implemented using EXOR gates. Using the 1-bit 2-D von Neumann PACA structure, one can build the desired hybrid \( N \times N \) -bit cellular automata [27-29]. It costs \( 6N^2 + N^2 + 4N \) bits to assign boundary condition, rule control, and initial data.
to indicate the 2-D hybrid \( N \times N \) von Neumann CA generator to produce CA data sequence. Given a 2-D \( N \times N \)-cell dual-state von Neumann CA runs over \( T \) time steps, it has \( 2^6 \) rules, \( 2^{N^2} \) initial configurations, \( 2^{4N} \) boundary conditions, and results in \( 2^{6N^2} \times N^2 \times 4N \) CA evolution ways for generating \( T \times N \) N-bit generalized CA data. Consequently, cyclic boundary conditions were imposed on a 2-state/3-site/ \( N \times N \)-cells CA to generate the states of the automata.

Figure 4. 2-D von Neumann CA, (a) 2-D von Neumann CA space, (b) the structure of 1-bit PACA

III. SCAN-CA-BASED IMAGE SECURITY SYSTEM USING THE PROPOSED STREAM CIPHER

Let \( F(i), 0 \leq i \leq L_i - 1 \) be a sequence of \( N \)-bit input data, \( E(i), 0 \leq i \leq L_i - 1 \) be a sequence of \( N \)-bit output encrypted data, \( D(i), 0 \leq i \leq L_i - 1 \) be a sequence of \( N \)-bit output encrypted data, and \( CA_p(i), 0 \leq i \leq L_i - 1 \) be a sequence of \( N \)-bit CA data. We firstly defined variable ordered generalized CA transforms (GCATs) as TABLE I. There are 4 groups of 1st-ordered GCATs and 16 groups of 2nd-ordered GCATs which were labeled as GCAT1 and GCAT2 in TABLE 1, respectively. Among GCAT1 and GCAT2, each group of GCAT1 consists of \( 2^N \) 1st-ordered GCATs and each group of GCAT2 consists of \( 2^{2N} \) 2nd-ordered GCATs, therefore \((5+2N)\)-bit Type Selection data were used to specify the type of GCAT.

Based on the definition of variable ordered GCATs, we firstly develop the architecture of the variable ordered recursive CA encrypted/decrypted substitution which is shown in Figure 5. In Figure 5, when encryption/decryption control bit is set to one and zero, it executes the recursive CA encryption and decryption substitution respectively. For example, the encryption/decryption control bit is set to one, it will cause that the switches SW1-1 and SW1-2 are in left position and system will perform variable ordered encryption according to the status of type selection. Type selection controller in Figure 7 used control bits \( T_4T_3T_2T_1T_0 \) to achieve type selection, where \( T_4 \) is used for 1st-ordered GCAT and 2nd-ordered GCAT selection, \( T_3 \) is \( CA_p(i) \) and \( CA_p(i-1) \) selecting bit, \( T_2 \) is EXOR and NEXOR selecting bit, \( T_1 \) is the control bit for selecting \( LS_1 \) and \( LS_2 \), and \( T_0 \) is used for \( E(i-1) \) and \( E(i-2) \) selection. Type selection control bits and its corresponding GCAT is listed in TABLE 1, for example, \( T_4T_3T_2T_1T_0 = 10000 \) will cause system to perform group 2 of 2nd-ordered recursive CA-encrypted/decrypted substitutions because \( T_4T_3T_2T_1T_0 = 10000 \) makes switches SW2-1 and SW2-2 to change position from left to right, switch SW2-3 is in up position, and all other switches are keeping in their current positions.

Figure 5. Architecture of the variable ordered recursive CA encrypted/decrypted substitutions

We then developed the 1st-ordered and the 2nd-ordered recursive CA-encrypted substitutions to achieve stream cipher as Eqs. (2) and (3), respectively. \( GCAT1(E(i-1), CA_p(i) \lor CA_p(i-1)) \) in Eq. (2) means that \( E(i-1) \) and \( CA_p(i) \) or \( CA_p(i-1) \) execute the 1st-ordered GCAT. Meanwhile, \( GCAT2(E(i-1), \ldots) \)
In Eq. (3) means that $E(i-1)$, $E(i-2)$, $C_A(i)$, and $C_A(i-1)$ execute the 2nd-ordered GCAT. Reversing the operations of the recursive CA encryption is to perform the recursive CA decryption. The 1st-ordered and the 2nd-ordered recursive CA decrypted substitutions can be expressed as Eqs. (4) and (5) respectively. Notably, the generalized CA transforms $GCAT(E(i-1), C_A(i)\lor C_A(i-1))$ and $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$ for encryption and for decryption are identical. Since the CA encryption/decryption scheme is lossless, the sequence of $N$-bit decrypted data $D(i), 0 \leq i \leq L_1 - 1$ is identical to the original sequence $F(i), 0 \leq i \leq L_1 - 1$. To achieve the goal of data security, we further develop the scheme of CA encryption/decryption (stream cipher system) as Figure 6, where the CA generating scheme shown in dash-line block is controlled by CA key to generate N-bit CA data sequence $C_A(i), 0 \leq i \leq L_1 - 1$ for CA substitution. For CA encryption, the encryption/decryption control bit is set as 1; simultaneously, the input is a sequence of $N$-bit data and the output of recursive CA-encrypted substitution is a sequence of $N$-bit encrypted data.

<table>
<thead>
<tr>
<th>Groups</th>
<th>Operations</th>
<th>Type selection control bits $T_1T_2T_3T_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: $GCAT(E(i-1), C_A(i))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i)) \mod 2^N$</td>
<td>000xx</td>
</tr>
<tr>
<td>2: $GCAT(E(i-1), C_A(i))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i)) \mod 2^N$</td>
<td>010xx</td>
</tr>
<tr>
<td>3: $GCAT(E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>011xx</td>
</tr>
<tr>
<td>4: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>10000</td>
</tr>
<tr>
<td>5: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>10010</td>
</tr>
<tr>
<td>6: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>10100</td>
</tr>
<tr>
<td>7: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>10110</td>
</tr>
<tr>
<td>8: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>11000</td>
</tr>
<tr>
<td>9: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>11010</td>
</tr>
<tr>
<td>10: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>11100</td>
</tr>
<tr>
<td>11: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>11110</td>
</tr>
<tr>
<td>12: $GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))$</td>
<td>$((E(i-1)+L_S)\oplus C_A(i-1)) \mod 2^N$</td>
<td>11111</td>
</tr>
</tbody>
</table>

**TABLE 1. GENERALIZED CA TRANSFORM (GCAT)**

1st-ordered CA encryption: $E(i) = [F(i) + GCAT(E(i-1), C_A(i)\lor C_A(i-1))] \mod 2^N, 0 \leq i \leq L_1 - 1$;  
2nd-ordered CA encryption: $E(i) = [F(i) + GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))] \mod 2^N, 0 \leq i \leq L_1 - 1$;  
1st-ordered CA decryption: $D(i) = [E(i) - GCAT(E(i-1), C_A(i)\lor C_A(i-1))] \mod 2^N, 0 \leq i \leq L_1 - 1$;  
2nd-ordered CA decryption: $D(i) = [E(i) - GCAT(E(i-1), E(i-2), C_A(i), C_A(i-1))] \mod 2^N, 0 \leq i \leq L_1 - 1$.  

© 2010 ACADEMY PUBLISHER
The proposed recursive CA encrypted substitution satisfies both confusion and diffusion properties. The confusion and diffusion properties are achieved by transforming the sequence $F(i), 0 \leq i \leq L_1 - 1$ into the sequence $E(i), 0 \leq i \leq L_1 - 1$ according to Eqs. (2) and (3). The sequence $E(i), 0 \leq i \leq L_1 - 1$ yields uniformly distributed pixels because the high-quality random key-stream $CA_i(i), 0 \leq i \leq L_1 - 1$ and/or $CA_j(i-1), 0 \leq i \leq L_1 - 1$ are used in the transformation. The sequence $E(i), 0 \leq i \leq L_1 - 1$ has the diffusion property because a single change in value $F(i)$ changes $E(i)$ which changes $E(i+1)$ which changes $E(i+2)$ and changes propagate up to the end of the sequence.

SCAN-CA-based Image security system using the proposed stream cipher system is shown in Figure 7. The security keys for encryption and decryption consist of four components, namely, SCAN key, data reformation key, type selection key and CA key. These keys are identical and are known to both the sender and the receiver before the communication of encrypted image. The scan key is represented by encryption-specific SCAN language and is used for reformatting the data type of the input image. The scan key is represented by encryption-specific SCAN language. For a sequence of $N$-bit data to produce a sequence of $N$-bit encrypted data $E(i), 0 \leq i \leq L_1 - 1$.

The receiver site performs inverse operation of the sender site. Suppose receiver has received data SCAN key, reformation key, type selection key, CA key, and a sequence of $N$-bit encrypted data. The decryption will be done as follows. Firstly, the recursive CA decrypted substitution performs CA decryption to generate the sequence of $N$-bit decrypted data $D(i), 0 \leq i \leq L_1 - 1$. Inverse data reformation was next done to produce the decrypted data with the original data type, and then the SCAN key rearranges these data into 2-D array as original input image.

The proposed stream cipher system is an extended and improved version of that presented in [26] because the proposed one used 20 groups of variable ordered CA-based recursive substitution to make the exhaustive searching attack much harder and to enhance the performances of system.

IV. POSSIBLE SECURITY KEYS AND CRYPTANALYSIS

Let $S(n)$ be the number of scan patterns of an 2-D $2^n \times 2^n$ array generated by the SCAN key defined by the encryption-specific SCAN language. For a $2^n \times 2^n$ image, there are eight basic scan patterns shown in Figure 1 each with eight transformations resulting in 64 basic scan-transformation patterns. When $n \geq 3$, there are additionally 24 ways shown in Figure 2 to partition the image into four sub-regions of size $2^{n-1} \times 2^{n-1}$ each having $S(n-1)$ recursive scan patterns. This results in $S(2) = 64$ and $S(n) = 64 + 24S(n-1)^2$, $n \geq 3$. However, only a portion of scan patterns with a finite number of scan iterations shall achieve a good dispersion, the length of scan key was thus carefully determined as 46 bits, meaning that only $2^{46}$ scan patterns shall be used in our simulation.

A 2-D hybrid $N \times N$-cell dual-state von Neumann hybrid CA that runs over $T$ time steps produces $2^{(2N) + 2N^2 + 4N} \times (T \times N)!$ possible groups of $T \times N$ N-bit CA data. However, for efficient computer simulation, a special CA rule generator with $(6+n)$-bit rule control data is used to specify specific CA rule numbers, where $n = \log_2 N$. Furthermore, since $T \times N$ N-bit generalized CA data have $(T \times N)!$ possible permutations, it could be a very large number. Therefore, the linear permutation with $n_r = \log_2 (T \times N) = \log_2 T + \log_2 N = n_r + n_N$ bits
is compact in representing and generating a specific set of permutations. In summary, in the computer simulation, $2^{(46+4)+((5+2N)+((6+n)+N^2+4N+n_p))}$ possible groups of $T \times N$-bit generalized CA data were used. Notably, the proposed system specifies four data types and $2^{5+2N}$ GCAT types. The basic idea of the proposed stream cipher system is that it uses a specified key to generate a key-stream and use it to encrypt a plaintext string according to Eqs. 2 and 3. We hence choose the length of key-stream at least equals to the length of plaintext to match the goal of security. Additionally, we know that the length of a CA state cycle is very important in determining the suitability of the CA as a generator of random numbers. According to [30], the average cycle length for 2-D $N \times N$-cell dual-state von Neumann CA increases exponentially and is on the order of $2^{N^2-3}$ for $N < 8$ or $2^{N^2-4}$ for $N \geq 8$. Therefore, we have to choose a suitable 2-D $N \times N$-cell CA to produce high quality key-stream for encryption according to the size of image. The relationship between the image size and the minimum size of a suitable 2-D CA is described as follows. If an image is with size of $2^n \times 2^n$ want to be encrypted, then the minimum size of a suitable 2-D CA will be longer than $60^2$, which will produces 8-bit key-stream with the cycle length more than $2^{65}$ and is larger than the size of tested images, i.e. $2^{65} = 256 \times 256$. Note that the size of 2-D CA is not more than $8 \times 8$ according to [30] even the image size is more than $4096 \times 4096$. However, we used CA with different size of $4 \times 4$ until $32 \times 32$ to show its possible CA keys are high volume increasing.

In summary, in the computer simulation, $2^{46}$ scan patterns and $2^{(46+4)+((5+2N)+((6+n)+N^2+4N+n_p))}$ possible groups of $T \times N$-bit generalized CA data were used. Notably, the proposed system specifies four data types and $2^{5+2N}$ GCAT types. Thus, TABLE II presents a high volume of keys.

### TABLE II

<table>
<thead>
<tr>
<th>Possible SCAN keys for $2^n \times 2^n = 256 \times 256$ images</th>
<th>Possible data reformation keys</th>
<th>Possible GCAT type selection</th>
<th>2-D $N \times N$ von Neumann CA</th>
<th>Time steps $T$</th>
<th>Possible CA keys $2^{(46+4)+((5+2N)+((6+n)+N^2+4N+n_p))}$</th>
<th>Possible security keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&gt; 10^{556}$</td>
<td>4</td>
<td>$2^{5+2N}$</td>
<td>$4 \times 4$</td>
<td>32768</td>
<td>$2^{217}$</td>
<td>$&gt; 10^{9568}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$8 \times 8$</td>
<td>8192</td>
<td>$2^{891}$</td>
<td>$&gt; 10^{9770}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$16 \times 16$</td>
<td>2048</td>
<td>$2^{1039}$</td>
<td>$&gt; 10^{10688}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$32 \times 32$</td>
<td>512</td>
<td>$2^{7580}$</td>
<td>$&gt; 10^{14785}$</td>
</tr>
</tbody>
</table>

Cryptosystems must withstand the most types of attack such as ciphertext only attack, known plaintext attack, chosen plaintext attack, and chosen ciphertext attack. We had shown that our image encryption method satisfies the perfect secrecy condition $P[F = F(i)|E = E(j)] = P[F = F(i)] \forall i$ in [26], that is, the cryptanalyst can yield no information about the plaintext by observing the ciphertext because of the system’s perfect secrecy. This result proves that the system can withstand ciphertext only, and chosen ciphertext attacks. The cryptanalyst can use known plaintext and chosen plaintext attacks to this scheme guess the security key because the cryptanalyst cannot obtain information about the plaintext by observing the ciphertext. For known plaintext and chosen plaintext attacks, cryptanalyst can do exhaustive key search attacks to guess the security key; however, it is a difficult task because the proposed system has many security keys with variable length $(2^{46}+4) + (5+2N)+((6+n)+N^2+4N+n_p)$ bits, for example, the proposed system has $2^{190}$ 190-bit and $2^{430}$ 430-bit possible security keys to encrypt $256 \times 256$ images for the 2-D von Neumann CA with size of $8 \times 8$ and $16 \times 16$ respectively. Moreover, cryptanalyst has to know the exact length of the security keys before he mounts his exhaustive key search attacks. Cryptanalyst knows that the security key consists of four sub-keys: SCAN key, data...
reformation key, type selection key, and CA key under the assumption that the length of the variable length security key is known, he hence wants to do divide and conquer attacks to separately and sequentially guess the individual sub-keys. The complexity of the worst case in the divide and conquer attack could be $2^{6N} + 2^2 + 2^5 + 2^N + 2^{4N} + 4N + 8$ due to all sub-keys are independently decided by authorized users. However, cryptanalyst really difficulty gets any information from the proposed system to guess all sub-keys because we have developed DSP+FPGA implementation to prevent the divide and conquer attacks. Finally, we will discuss a kind of security threat in which we assume that cryptanalyst gets the recovered part of key-stream revealed by the divide and conquer attack. In this case, minimum length of the recovered part of key-stream is $2^{-3N}(3N-1)$ bytes. Cryptanalyst firstly mounts brute force search attacks with the complexity of $2^{N^2}$ to guess the initial bits of CA key. Next, cryptanalyst uses the revealed rule control bits, boundary condition bits, and permutation control bits to do brute force search with the complexity of $2^{N^2}$ to guess the initial bits of CA key, once the states of 2-D $N\times N$ -cell von Neumann CA are matched with a special part of $2^{-3N}(3N-1)$ bytes recovered key-stream, cryptanalyst shall get the exact initial bits of CA key and then the exact CA key can be revealed. However, when the states of 2-D $N\times N$ -cell von Neumann CA always can’t match all of $2^{-3N}(3N-1)$ bytes recovered key-stream, how long the process of each brute force search attack shall be terminated? As mentioned above, we know that the average cycle length of the 2-D $N\times N$ -cell von Neumann CA is on the order of $2^{N^2}$ ($N < 8$) or $2^{N^2-4}$ ($N \geq 8$), which causes each mismatched brute force search attack shall be terminated after $2^{N^2}$ ($N < 8$) or $2^{N^2-4}$ ($N \geq 8$) time steps. For example, cryptanalyst terminates each mismatched brute force search attack after more than $2^{60}$ time steps for 2-D 8×8 -cell von Neumann CA. Cryptanalyst shall spends more than 34 years to terminate one mismatched brute force search attack under one time step is one nanosecond assumption. The fact of very long cycle length of the 2-D $N\times N$ -cell von Neumann CA consequently prevents cryptanalyst to reveal CA key from the recovered part of CA key-stream.

V. SOFTWARE SIMULATION RESULTS AND DSP+FPGA IMPLEMENTATION

A. Software Simulation Results

The proposed stream cipher system performed well encryption not only the general text data but also compressed images and uncompressed images. Several simulations of SCAN-CA-based image security system were conducted to test various properties of the proposed data security system that include confusion and diffusion properties. All of the software implementations were performed using personal computer with Intel® P4 CPU (3.2 GHz), Microsoft® Windows® XP, and Borland® C++ builder® 6.0. In our simulations, $Z_0(Z_0(01i01i01)Z_4(1d01d1d0)X_3(i0i1i0i1)X_2(i1i0i1i0))$ scan key with number of scan iterations 8 is used to rearrange the pixels of these tested images. The data reformation key is 01, which makes the data type for encryption and decryption to be 8-bit. The CA key is selected that chooses 6C uniform initial states, zero boundaries with cyclic boundary at lower right corner, and a special rule control to produce 2-D hybrid 8×8 von Neumann CA as Figure 8. Once the 2-D hybrid 8×8 von Neumann CA has been built that run over 8192 time steps to generate CA data sequence of size 65536. Then 8-bit permutation control bits 0001 in CA key guides the system to do linear permutation from the first 8-bit of the CA data sequence to generate the CA sequence to generate the CA_s (i−1), 0 ≤ i ≤ L−1. We used 1st-ordered GCAT with LS_1 = 128, and 2nd-ordered GCAT with LS_2 = 128 and LS_3 = 128 to perform the recursive CA encryption and decryption substitutions.

Figure 8. 2-D hybrid 8×8 von Neumann CA

![Figure 8. 2-D hybrid 8×8 von Neumann CA](image)

Figure 9. Test images, (a) Lena, (b) Lincoln Tower
Note that in all the following experiments, images are used for simulation and all images are of size 256 × 256. Figures 9a and 9b show YUV formatted color Lena and Lincoln Tower images that were used for testing the performance of SCAN-CA-based image security system. Encrypted images of Lena and Lincoln Tower are shown in Figure 10. Histograms of the encrypted Lena and the encrypted Lincoln Tower in Figure 11 show that the encrypted images get uniformly distributed pixels. This fact illustrates that the proposed data security system satisfies the confusion property. Encrypted images perform the process of decryption will produce the decrypted images. The decrypted images are exactly identical to the original images. This fact shows that the proposed system works well as our expectation.

In order to determine the diffusion property of the proposed system with respect to images, Lena image was modified by incrementing the value of one randomly chosen pixel by 1. The Y-value of pixel (0, 0) was incremented from 161 to 162. Both the original Lena and the modified Lena were encrypted using the same keys. The pixel-wise absolute difference of two encrypted images is displayed in Figure 12, which shows that the two encrypted images have no similarities even though their original images differ by only one pixel. Thus, it proves the diffusion property of the proposed system with respect to images.

Figure 10. Encrypted images, (a) 1st-ordered Lena, (b) 2nd-ordered Lena, (c) 1st-ordered Lincoln Tower, (d) 2nd-ordered Lincoln Tower

Figure 11. Histograms of images and its corresponding encrypted images, (a) Lena, (b) Lincoln Tower

Figure 12. Diffusion property of the proposed system with respect to images

Figure 13. Survival property of the proposed system,
(a) Corrupted ciphertext (encrypted image), (b) Decryption of the corrupted ciphertext, (c) Inverse SCAN of the decrypted image.

We previously mentioned that the SCAN-CA-based image security system is a synchronous cipher because the key stream of the proposed system is generated independently of the plaintext and the ciphertext, making it susceptible to synchronization problems. In a synchronous stream cipher, the sender and receiver must be in step for decryption to be successful. If digits are added or removed from the image during transmission, then synchronization is lost. One approach to solve the synchronization problem is to tag the ciphertext with markers at regular points in the output. However, if a small fraction of the ciphertext is corrupted in transmission, rather than added or lost, then only the corresponding fraction in the plaintext is affected and the error does not propagate to other parts of the plaintext; a large area of the image therefore survives. This characteristic makes the proposed system reliable in transmissions with high error rate. The survival properties of the proposed system are shown in Figure 13. Figure 13c shows that the pixels of the corrupted block are dispersed all over the image after inverse scan, which act noise-like pixels. These noise-like pixels can be filtered using the techniques of low-pass filtering.

<table>
<thead>
<tr>
<th>Item</th>
<th>Proposed system (Enhanced version of [26])</th>
<th>System of [26] with one iteration ((N \times N))</th>
<th>DCPcrypt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Stream cipher</td>
<td>Stream cipher</td>
<td>Stream cipher</td>
</tr>
<tr>
<td>Key length (bits)</td>
<td>(\text{length}^a)</td>
<td>(\text{length}^b)</td>
<td>256</td>
</tr>
<tr>
<td>Complexity of cryptanalysis</td>
<td>(2^{39+6N+3N^2+n_p})</td>
<td>(2^{11+4(N^2+N)+n_p})</td>
<td>(2^{306})</td>
</tr>
<tr>
<td>Possible GCA type selection</td>
<td>2 (^{5+2N})</td>
<td>6</td>
<td>—</td>
</tr>
<tr>
<td>CPU encryption time (c) (msec/3 Mbytes)</td>
<td>46 ± 16</td>
<td>31 ± 16</td>
<td>70 ± 8</td>
</tr>
<tr>
<td>CPU decryption time (c) (msec/3 Mbytes)</td>
<td>46 ± 16</td>
<td>31 ± 16</td>
<td>70 ± 8</td>
</tr>
<tr>
<td>Entropy of ciphertext (bits)</td>
<td>7.9999</td>
<td>7.9999</td>
<td>7.9999</td>
</tr>
</tbody>
</table>

a. \(\text{length}^a = 46 \text{ bits (SCAN key)} + 2 \text{ bits (Data reformation key)} + (5+2N) \text{ bits (Type selection key)} + (6 + n + N^2 + 4N + n_p) \text{ bits (CA key)}\).

b. \(\text{length}^b = 2 \text{ bits (Data reformation key)} + 3 \text{ bits (Type selection key)} + (6 + n + N^2 + 4N + n_p) \text{ bits (CA key)}\).

c. CPU encryption/decryption time is the CPU processing time for encrypting/decrypting 3 Mb of plaintext/ciphertext excluding the time required for hard disk storage.

B. DSP+FPGA Implementation

At present, real-time data security has become one of main key problems of real-time multimedia applications. We hence have the motivation to develop a real-time image security system based on DSP and FPGA to evaluate the performances of our proposed system. In the proposed SCAN-CA based image security system, the SCAN approach rearrange 2D image pixels into 1D array which costs a large amount of computations and consumes a lot of memory, we therefore use DSP to implement. Moreover, the variable ordered recursive CA substitutions replace the rearranged pixel values which cost a lot of arithmetic and logic operations, and therefore FPGA is one good solution to implement. Figure 14 shows the architecture of SCAN-CA based image security system where TI DSP development board provides TMS320C6416T DSP processor to serve as SCAN processing unit, Altera GFEC Stratix II FPGA board gives EP2S60F1020 FPGA chip to serve as the processing unit of variable ordered recursive CA substitutions. However, due to TI DSP development board and Altera GFEC Stratix II FPGA board have different the clock rate and the voltage supply, it is hence need a additional connecting board to perform the
interfacing buffer. The detailed architectures of SCAN processing unit, processing unit of variable ordered recursive CA substitutions, and connecting board for interfacing between DSP board and FPGA board are shown in Figs. 15, 16, and 17, respectively.

We used TI Code Composer Studio 3.1 to program SCAN approach and then the compiled code should be downloaded into TI DSP development board using USB560 JTAG emulator. Meanwhile, Altera provided Quartus II 5.0 can be used for the implementation of the variable ordered recursive CA substitutions, and the output POF file should be written into the EPROM of EP2S60F1020 FPGA chip. Figure 18 shows the integration of DSP+FPGA platform of the proposed SCAN-CA-based image security system which can be used to develop the real-time image security system. Readers are referred to [36] for a detailed description of DSP+FPGA platform of the proposed SCAN-CA-based image security system.

**VI. DISCUSSIONS AND CONCLUSIONS**

This paper presented a new stream cipher system based on SCAN and variable ordered recursive CA substitutions. Its security method is based on permutation of the data and replacement of the data values. Permutation is done by scan patterns generated by the SCAN approach. The replacement of data values using variable ordered recursive cellular automata (CA) substitutions. The salient features of the proposed stream cipher system can be summarized as follows. (1) Keys consist of SCAN key, data reformation key, type selection key, and CA key, which are of variable lengths producing a large number of possible keys, more than $10^{66} \sim 10^{78}$ - according to the size of the 2-D von Neumann CA. (2) Choosing a suitable size for the 2-D CA, according to the size of the image, enables the system to withstand the cropping-and-replacement attack. (3) The system is economic in consuming computational resources because the encryption/decryption scheme uses integer arithmetic and logic operations. Comparative results
show that the performance of the proposed system is superior to those of [26], RC4, Triple-DES, and AES because of its more complicated cryptanalysis and shorter CPU decryption and decryption time for particular ciphertext entropy. Moreover, this system withstands the survival against attack as well as RC4, Triple-DES, and AES. Finally, we established DSP+FPGA platform for developing real-time image security system. The experiments on such a real system demonstrate that the proposed image security system is suitable for the real-time image security.

ACKNOWLEDGEMENTS

The authors would like to thank anonymous reviewers for their valuable suggestions to improve this paper. The National Science Council of Republic of China under the contract NSC-97-2221-E-239-036-MY3 supported this work.

REFERENCES


Rong-Jian Chen was born in 1958, Taiwan. He received the B.S., M.S., and Ph.D. degrees in electronic engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1987, 1991, and 1995, respectively.

He joined the faculty of the National Lien-Ho Institute of Technology in August 1995, where he was the Chair of the Department of Electronic Engineering during 1999–2001 academic years. At present, he is Associate Professor of the Department of Electronic Engineering, National United University.


Jui-Lin Lai was born in 1955, Taiwan. He received the B.S degree from the Electronic Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, 1984. He received the M.S and Ph.D degree in the Institute of Control Engineering and the Institute of Electronic Engineering from the National Chiao-Tung University, Taiwan, R.O.C., in 1990 and 2004, respectively.

He joined the faculty of the National Lien-Ho Institute of Technology in August 1984, where he was the Chair of the Department of Electronic Engineering during 1993–1996 academic years. At present, he is Associate Professor of the Department of Electronic Engineering, National United University.

Dr. Lai had been elected as IEEE Senior Member in 2005. His research interests include analog and digital VLSI design, neural networks, and computing architecture, and nanotechnology.

Shi-Jinn Horng was born in 1957, Taiwan. He received the BS degree in Electronics Engineering from National Taiwan Institute of Technology, Taipei, the MS degree in Information Engineering from National Central University, Taiwan, and the PhD degree in Computer Science from National Tsing Hua University, Taiwan, in 1980, 1984, and 1989, respectively.

He was a Professor and Dean of the College of Electrical Engineering and Computer Science, National United University, Miaoli, Taiwan. Currently, he is a Professor at the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology.

Dr. Horng has published more than 170 research papers and received many awards; especially, the Distinguished Research Award between 2004 and 2006 from the National Science Council in Taiwan; Outstanding I. T. Elite Award, in 2005; Outstanding EE. Prof. Award, the Chinese Institute of Electrical Engineering; Outstanding Research and Invention Award between 2006 and 2008 from National Taiwan University of Science and Technology.
A Novel Data Mining-Based Method for Alert Reduction and Analysis

Fu Xiao  
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P. R. China  
Email: fuxiao1225@hotmail.com

Shi Jin  
School of State Secrecy and Department of Information Management, Nanjing University, Nanjing, P. R. China  
Email: zgnjack@hotmail.com

Xie Li  
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P. R. China  
Email: xieli@nju.edu.cn

Abstract—Current system managers often have to process huge amounts of alerts per day, which may be produced by all kinds of security products, network management tools or system logs. This has made it extremely difficult for managers to analyze and react to threats and attacks. So an effective technique which can automatically filter and analyze alerts has become urgent need. This paper presents a novel method for handling IDS alerts more efficiently. It introduces a new data mining technique, outlier detection, into this field, and designs a special outlier detection algorithm for identifying true alerts and reducing false positives (i.e. alerts that are triggered incorrectly by benign events). This algorithm uses frequent attribute values mined from historical alerts as the features of false positives, and then filters false alerts by the score calculated based on these features. We also proposed a three-phrase framework, which not only can filter newcome alerts in real time, but also can learn from these alerts and automatically adjust the filtering mechanism to new situations. Moreover our method can help managers analyze the root causes of false positives. And our method needs no domain knowledge and little human assistance, so it is more practical than current ways. We have built a prototype implementation of our method. Through the experiments on DARPA 2000, we have proved that our model can effectively reduce false positives. And on real-world dataset, our model has even higher reduction rate. By comparing with other alert reduction methods, we believe that our model has better performance.

Index Terms—Intrusion detection, false positives, outlier detection, alert reduction

I. INTRODUCTION

Current system managers often have to process huge amounts of alerts per day, which may be produced by all kinds of security products, network management tools or system logs. Even awfully, single product can also produce a lot of alerts. For example, people have observed that IDS can easily trigger thousands of alerts per day, up to 99% of which are false positives (i.e. alerts that are triggered incorrectly by benign events)[1]. This flood of mostly false alerts has made it very difficult for managers to analyze security state. Moreover the reactions to dangerous attacks are often delayed, because the alerts for them are hidden in huge amounts of trivial ones and are often neglected. So how to reduce false positives in intrusion detection is an important problem needing researchers pay more attentions to.

Now one popular solution to this problem is using certain algorithms (e.g. machine learning or statistical algorithms) to identify true alerts and filter false ones from raw data. Some related methods have been proposed but most of them have 3 limitations. Firstly, they usually need a lot of labeled training data or domain knowledge to build their alert reduction model. However these data are often difficult to obtain. Secondly, most of them are off-line model which will delay the reaction to attacks. Thirdly, most of them can not adapt to new situations. In this paper we proposed a novel method, which hasn’t above limitations. It is based on a new data mining technique, outlier detection. This technique has been successfully applied in many fields (e.g. fraud detection, weather prediction), but has not been used to reduce false positives. Moreover, our method can help managers find the root causes of false positives. As we know, most of current alert reduction systems can’t do this work.

In order to filter IDS alerts better, we have designed a special outlier detection algorithm for this field, i.e. an improved frequent pattern-based outlier detection algorithm. It assigns each alert an FP score, which indicates the possibility that the alert is a false positive. The score is calculated based on how many frequent attribute values the alert contains. Usually the more frequent patterns an alert has, the higher its score is, and the more likely it is a false positive. In order to filter alerts in real time, we also design a three-phrase framework. In the learning phrase, we build the feature set of false positives and calculate the threshold of true alerts based...
on this set. Then in the online filtering phrase, we compare the FP score of each new event with this threshold so as to determine whether it is false positive or not. The feature set is automatically updated so as to keep its accuracy. Finally in the phrase for discovering root causes, the feature set is used to help manager analyze the root causes of false positives.

We have validated our method by experiments on both DARPA 2000 dataset and real-world data. The results on DARPA 2000 show that when 86% of alerts have been filtered by our model, 100% of true alerts still remain. On real-world dataset our model has even higher reduction rate. And after comparing with two representations of current reduction methods, we believe that our system has better performance.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 introduces our outlier detection-based alert reduction model in detail. Section 4 presents our experiments and compares our method with others. Section 5 concludes the paper and introduces the future work.

II. RELATED WORK

There is still not so much work on identifying and reducing IDS alerts. Current methods can be divided into three categories, which are described as follows:

Method based on Classification. The main idea of this method is building an alert classifier that tells true from false positives. Tadeusz Pietraszek has presented one such system in [2]. It firstly generates training examples based on the analyst’s feedback. Then these data are used by machine learning techniques to initially build and subsequently update the classifier. Finally the classifier is used to process new alerts. This method can classify alerts automatically. But its main shortcoming is producing labeled training data is labor intensive and error prone.

Method based on Root Cause Analysis. This method firstly discovers and understands the root causes of IDS alerts. Then according to these causes, the alerts triggered by attacks can be distinguished from those triggered by benign events. In [1, 3 and 4], Klaus Jilisch has proposed such a model based on conceptual clustering. It generalizes alerts according to the Generalization Hierarchies built on each alert attribute. And the final generalized alerts will be presented to users to help root cause analysis. The main disadvantage of this method is Generalization Hierarchies are not easy to build. It depends on the experience of domain experts and collecting enough background knowledge. Moreover this kind of methods only supports offline process.

Methods based on the Assumption that “Frequent Alert Sequences Are Likely Resulted from Normal Behaviors”. Now many alert reduction methods belong to this category. For example, in [5] frequent episode mining are applied on IDS alerts. When the frequent sequences of alerts that have the same destination are found, they will be presented to users in order to determine whether they are false positives or not. [6] and [7] is similar to [5], but [6] uses continuous and discontinuous patterns to model false positives, and [7] using frequent itemsets and association rules to model them. Paper [8] filters IDS alerts by monitoring alert flow. It models regularities in alert flows with classical time series methods. After removing the periodic components, slowly changed trend and random noise from time series, the remaining is regarded as true alerts. All methods mentioned above are based on modeling false positives by frequent or periodic alert sequences. It is effective in the environment where normal behaviors seldom change. However it often mistakes the new normal behaviors or infrequent ones for true alerts.

Our work differs from the above in that we use an unsupervised data mining technique that needs no domain knowledge and labeled training data. In addition, it needs little human assistance. So our model can overcome the shortages that classification or root cause analysis based methods have. In addition, our method can help managers find the root causes of false positives. Now only [1] can do this. Although our method is also based on frequent patterns, it is quite different from the third category of methods, because they are build on different assumptions. Our method assumes that alerts containing many frequent attribute values are likely resulted from normal behaviors. However the third category uses frequent alert sequences to identify these alerts. In other words, in our model, the frequent pattern is made up of alert attributes. But in the third category of methods, frequent pattern is constituted by alerts. Moreover current alert reduction methods such as [8] and [1, 3 and 4] can obtain true alerts only after removing all false positives or clustering all alerts. But our method can directly identify the outliers (true alerts) in larger amounts of data, so the cost wasted for processing false positives will be reduced and the reaction to attacks will be more rapid. In addition, our model can constantly update the feature set so that it is adaptive to new normal behaviors.

Alert correlation [9], which is a little similar to above work, is another kind of technique for analyzing intrusion alerts. However it mainly focuses on the reconstruction of attack scenarios, while our focus is identifying and reducing false positives.

III. OUTLIER DETECTION-BASED ALERT PROCESS

Outlier detection is a new data mining technique which has absorbed many attentions recently. It is able to identify abnormal data (called outlier) in large dataset. Now in the field of network security, people have successfully used outlier detection to implement IDS [10, 11]. However it has not been applied in alert reduction. In fact, compared with false positives which are the majority of IDS alerts, true alerts (i.e. alerts related to attacks) are just so-called “outliers”. So this technique can also be used to filter IDS alerts. Moreover, for its high efficiency, low cost and the unsupervised character, methods based on this technique can achieve good performance.

In this paper, we use a frequent pattern-based outlier detection technique (FP-Outlier) to filter false positives. We choose this algorithm because it can process the uneven IDS alerts (i.e. alerts are often the mixture of
several alert clusters, and the densities of these clusters are not equal). Moreover this algorithm can meet all requirements that Abdulrahman et al. proposed for alert reduction technique in [6] (i.e. high scalability, low cost, independent reduction process, and the capability of dealing with noisy data and any type of attributes contained in the alerts). In order to make this algorithm more fit for processing IDS alerts, we have improved it in two ways. In this section, we firstly introduce our improved algorithm, and then present an alert reduction and analysis framework based on it.

A. Improved FP-Outlier Algorithm

Frequent pattern-based outlier detection (i.e. FP-Outlier) is presented by Zengyou He et al. in [12]. This method is based on the following truth: given a set of supermarket transactions, where each transaction is a set of literals (called items). Frequent itemsets (also called frequent patterns) are those combinations of items that have transaction support above predefined minimum support (support means percentage of transactions containing these itemsets). If a transaction contains many frequent itemsets, it means that this transaction is unlikely to be an outlier because it possesses the “common features”. There are mainly 3 steps in this algorithm: Firstly, all frequent itemsets are found in the dataset by a certain mining algorithm. Secondly the outlier score of each transaction is calculated based on these frequent patterns. Finally, transactions are sorted ascendingly according to their outlier scores, and the top p% of which are selected as candidate outliers. In the second step, the outlier score is measured by Frequent Pattern Outlier Factor (FPOF). Its definition is given as follows:

Definition 1 (Frequent Pattern Outlier Factor) Let \( D = \{ t_1, …, t_n \} \) be a set of n transactions. Support(\( X \)) denotes the ratio between the number of transactions that contain itemset \( X \) and the number of transactions in \( D \). minisupport is a user defined threshold. It defines the minimal support of frequent itemset. And \( FPS(D, \text{minisupport}) \) is a set of frequent patterns mined from \( D \) with minisupport. Given a transaction \( t_i \), its Frequent Pattern Outlier Factor (FPOF) is calculated as follows:

\[
FPOF(t_i) = \frac{\sum \text{support}(X)}{\text{FPS}(D, \text{minisupport})} \tag{1}
\]

The key idea of FP-Outlier is using frequent itemsets as features to identify “normal” data. So the more frequent and repeated the normal behaviors are, the better this algorithm can work. After analyzing IDS alerts, we found that they just had above characters. For example, a Snort deployed in the network of our laboratory can produce more than 30,000 alerts a week, and 95% of them are 4 types, viz., “SNMP public access udp”, “SNMP request udp”, “(http_inspect) BARE BYTE UNICODE ENCODING” and “NETBIOS SMB IPC$ unicode share access”. Through carefully analyzing, we believe that they are all triggered by configuration problems. It means all of them are false positives. Thus it is obvious that if we mine frequent itemsets from IDS alerts (each alert is regarded as a transaction, and each alert attribute is regarded as an item), the frequent patterns we get will mostly come from false positives. So these patterns can be regarded as features of false positives and used to filter them.

In order to make the FP-Outlier algorithm more fit for IDS alerts, we have improved it in two aspects.

Our first improvement is assigning each alert attribute a weight. That is because we have observed that when determining whether or not an alert is false positive, the attributes of this alert usually have different impact to the final decision. For example, there are two alerts. The first one contains frequent pattern “AlertType = SNMP public access udp”. And the second one contains frequent pattern “Destination Port = 80”. In general the first one is more likely to be a false positive than the second, because port 80 is common to both attacks and normal events. In order to show this difference, we should assign a higher weight to the “AlertType” attribute than to the “Destination Port” attribute.

In our model, the frequent pattern mined from IDS alerts is composed of alert attributes, so after assigning alert attributes different weights, frequent pattern will have a weight too. Its weight is defined as follows:

**Definition 2 (Weight of a Frequent Pattern)** Let \( A = \{ A_1, \ldots, A_k \} \) be a set of alert attributes name, each with an associated domain of value. The weight of \( A_i \) has been given as \( w_i \) (\( 1 \leq i \leq n \)). A frequent pattern \( X \) which contains \( A \) is a tuple over \( A \). Then the weight of \( X \) is defined as:

\[
\text{Weight}(X) = \max\{w_i\}, \quad \text{where } w_i \text{ is the weight of } A_i, \quad A_i \in A \text{ and } A \text{ is contained by } X, 1 \leq i \leq n.
\]

In other words, the weight of a frequent pattern is equal to the largest weight of all alert attributes contained by it.

We have proved by experiments that assigning alert attributes different weights can really improve performance (re. section IV.A). And currently in our prototype, the weight of alert attributes is set by a semi-automated method, which includes following steps: Firstly, we sampled n alerts randomly from all IDS alerts (in our experiment, \( n = 30 \)), and manually classified them into true alerts and false positives. Secondly, for each alert attributes \( i \), all frequent values of it were found, and the total number (recorded as \( S_i \)) of frequent values that appeared in both false positives and true ones was calculated. After that, the weight of attribute \( i \) (i.e. weight\( i \)) was calculated according to formula 2:

\[
\text{Weight}(i) = 1 - \frac{S_i}{\text{the number of frequent values of attribute}}. \tag{2}
\]

Finally, above steps were repeated for several times, and the weight of each attribute was set to the mean of the results we got in each round. The weight value can also be designed by an automated method, such as the technique frequently used in information retrieval. For example the formula proposed by A. Leuski in [13] can be used to calculate the weight value automatically. But the computational cost of this formula is quite high. So we are still looking for better automated method now.
Our second improvement to FP-Outlier is using dynamic feature set to describe normal behaviors (i.e. false positives). With the arrival of new alerts, new type of normal behavior also emerges. So in order to keep the feature set up to date, the frequent pattern describing new behaviors should be constantly added to features set. This improvement will cause the value of ||FPS(D, minisupport)|| in formula 1 also change constantly. Then the outlier score of alerts in different period will become incomparable. For example, with the growth of alert set, the number of frequent patterns will also increase. Then even for the same alert, the outlier score calculated in the early time will be bigger than the one calculated later. It is obviously inappropriate. In order to solve this problem, we removed ||FPS(D, minisupport)|| from our calculating of outlier score. This change will not violate FP-Outlier’s basic idea (i.e. determine the outlier degree of an object by how many frequent pattern it contains). So it is the needless cost for us in deed.

According to above improvements, we propose our algorithm for identifying false positives. It can be described as following.

 Firstly, we assign each alert a False Positive Score (FP score) which represents the possibility that this alert is a false positive. This score is measured by Common Feature Factor (CFF). Its definition is:

**Definition 3** (Common Feature Factor) Let \( D = \{a_1, \ldots, a_n\} \) be a set of alerts, and each attribute of alert in \( D \) has a predefined weight. Given \( \text{minisupport} \), the minimal support of frequent itemset, \( FPS(D, \text{minisupport}) \) is a set of frequent itemsets mined from \( D \) with \( \text{minisupport} \). Support\((X)\) denotes the ratio between the number of alerts that contain itemset \( X \) and the number of alerts in \( D \). And weight\((X)\) is the weight of \( X \). Then for each alert \( a \), its Common Feature Factor (CFF) can be calculated as follows:

\[
\text{CFF}(a) = \sum_{X \subseteq a, X \in FPS(D, \text{minisupport})} \text{support}(X) \times \text{weight}(X) \cdot (3)
\]

Then base on this formula, we present an algorithm CalculateFPScore to calculate each alert’s FP score. Following is the pseudo code of this algorithm.

```
Algorithm CalculateFPScore
Input: FS // the feature set of false positives
       alert // the alert to be processed
Output: FPScore // indicates the possibility that
         an alert is a false positive
01 begin
02 FPScore = 0
03 for each frequent itemset X in FS do begin
04    if alert contains X then
05        FPScore = FPScore + weight(X)*support(X)
06    end if
07 end
08 return FPScore
09 end
```

Figure 1. Algorithm for Calculate Each Alert’s FP Score

---

**B. Framework for Alert Reduction and Analysis**

This section gives a detailed introduction on how to use the improved algorithm to filter false positives. The original FP-Outlier algorithm is only designed for off-line process. The main shortcoming of off-line process is that attack-related alerts can’t be presented to users immediately and the reactions to them are often delayed. So we design a real-time framework to solve this problem. This framework consists of 3 phrases: the learning phrase and the online filtering phrase (shown in figure 2) are used to filter alerts in real time. And the third phrase, phrase for discovering root causes, is used to help managers analyze causes of false positives.

**Learning Phrase.** The main work in this phrase is building the feature set of false positives and calculating the threshold of true alerts. Feature set is composed of frequent itemsets mined from IDS alerts. These itemsets, which are the combinations of attribute values, are regarded as the features of false positives and used to identify them. The threshold of true alert is used to tell true from false positives. Alerts whose FP scores bigger than this threshold are used as false positives, otherwise true. In this paper, the threshold is equal to the largest FP score of candidate true alerts.

There are two difficulties in this phrase: Firstly, new type of false positives continually emerges, so it is difficult to determine how many data are enough for building the feature set. Secondly, not all frequent patterns mined from IDS alerts can be used as features. That’s because some attacks (called **frequent attack** in the rest of paper) can also trigger many similar IDS alerts (e.g. some probing or DoS attacks). Frequent patterns mined from these alerts are obviously not features of false positives.

In order to address the first problem, we design a block-by-block mining method. At first, the alerts sequence is divided into several blocks. Each block contains \( n \) alerts which have neighboring timestamps. Then we mine frequent patterns in every block, and the patterns found are put into feature set. If the next two or three blocks all can not contribute new features, the feature set will be regarded as stable and we can stop collecting new alerts. Because some itemsets (called **global frequent itemset** in the rest of paper) are only frequent in a long run, we have to combine all blocks and mine them to find this kind of pattern in the end. It should be noticed that the supporting instances of frequent patterns are all more than one predefined minimal value.

As to the second problem, we designed an algorithm to automatically filter this kind of pattern. We have observed that alerts triggered by frequent attack usually occur many times in a short period, and they often have the same destination or source address. But false positives usually don’t have these characters. So this is helpful to identify them. In our filtering algorithm, we designed two statistics: the first one is the number of certain type of alerts from the same source IP in the last \( T \) seconds, and the second one is the number of certain type of alerts to the same destination IP in the last \( T \) seconds. We calculate the two statistics for each alert type in the
frequent itemsets. If any statistic is larger than predefined threshold (e.g. in DARPA 2000, we define this threshold is 15 alerts per minute), this type of alerts will be regarded as triggered by frequent attack.

Detailed process in learning phrase is presented as follows (shown in figure 2):

Step1, Constructing Feature Set. Collect n alerts in the order of arrival as one block and mine them for frequent patterns. Add new patterns into the feature set, and then collect next n alerts. Repeat above process until the next k blocks (we choose k = 3) all contribute no new patterns. Combine all blocks into one set and mine the whole dataset for global frequent itemsets. Put these patterns into feature set. In our system, we use Apriori algorithm, which is a famous data mining algorithm, to mine frequent itemset. The detail of this algorithm can be found in paper [14].

Step2, Filtering Frequent Attacks. Check all alert types that appear in feature set, and determine whether or not they are frequent attacks. For those frequent attacks, the FP score of related alerts are set to 0, and related frequent patterns are removed from the feature set.

Step3, Calculating the Threshold of True Alerts. Based on the feature set obtained in above steps, use CalculateFPScore algorithm to calculate the FP score of each remaining alert. Alerts are sorted ascendingly according to their score and top p% of them is put into the set of candidate true alerts. Set the threshold of true alerts to the largest score in the set of candidate true alerts. And after all alerts in this set are recommended to users, the set of candidate true alerts is reset to empty.

Online Filtering Phrase. The main work in this phrase is calculating the FP score for each newcome alert, and then comparing this score with the threshold of true alerts, so as to determine it is true or false. Besides this, the feature set is updated continually in order to keep its accuracy. Detailed process in this phrase is presented as follows (shown in figure 2):

Step1, Filtering Each Newcome Alert. When a new alert come, it is firstly put into the alert cache, then its FP score is calculated based on the feature set by CalculateFPScore algorithm. If the score is not bigger than the threshold, this alert is regarded as true and is recommended to users, otherwise it is discarded.

Step2, Updating the Feature Set. As soon as the account of alerts in cache is equal to n, a frequent itemset mining is executed on these alerts. If the result contain some frequent patterns that the feature set hasn’t, filter frequent attacks from these new patterns (similar to step2 in the learning phase) and the remaining are added into the feature set. As to the patterns that the feature set already has, we only need to update the support, i.e., their supports in feature set are set to the mean of new support and the old one. All frequent attack-related alerts are put into the set of candidate true alerts and then recommended to users. Finally, store all alerts in cache into alert database, and then clear up the cache.

We should mention that with the coming of new alerts, the threshold calculated based on the early feature set...
will become inaccurate gradually. Moreover, although the feature set is updated continually, some global patterns are still possibly missed. So it is necessary to mine the whole alert databases so as to adjust the threshold and the support of frequent patterns. This process can be done at comparatively long intervals. And in order to avoid affecting the real-time filtering, it had better be executed in background offline.

Phrase for Discovering Root Causes. The main work in this phrase is using the feature set obtained before to help managers find the root causes of false positives. After careful observation, we found that the frequent patterns in the feature set not only can be used to identify false positives, but also can disclose the general rules of normal behaviors that trigger false positives. And these rules are very useful for understanding the root causes of these false positives. For example, table I shows part of feature set obtained after mining 4000 true IDS alerts (come from the Snort deployed on a proxy of our laboratory). From it, we can know that there are totally 887 alerts whose type is “SNMP public access udp” in the 4000 alerts. Moreover the 887 alerts all aims at the 161 port of site 202.119.36.253. As we known, site 202.119.36.253 is a printer in fact. It is a HP Laser printer shared by the whole laboratory through intranet. Considering this, we can conclude that these alerts are triggered by normal print tasks and are likely to be false positives. Then after careful analysis, we found the root causes of these alerts are that the client of HP Printer needs to use SNMP to identify the extended state of printer. So our guess is right.

### TABLE I.

**PART OF FEATURE SET FROM TRUE ALERTS**

<table>
<thead>
<tr>
<th>Frequent Itemset</th>
<th>Support Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlertType=SNMP public access udp</td>
<td>887</td>
</tr>
<tr>
<td>AlertType=SNMP public access udp</td>
<td>887</td>
</tr>
<tr>
<td>Destination IP=202.119.36.253</td>
<td>887</td>
</tr>
<tr>
<td>AlertType=SNMP public access udp</td>
<td>887</td>
</tr>
<tr>
<td>Destination Port=161</td>
<td>887</td>
</tr>
<tr>
<td>AlertType=SNMP public access udp</td>
<td>887</td>
</tr>
<tr>
<td>Destination Port=161</td>
<td>887</td>
</tr>
</tbody>
</table>

Now our model directly presents the feature set to users, and some mechanism such as querying and sorting interfaces are also provided so as to help analysis. The root causes of false positives still can not be automatically found currently. However based on the feature set and before-mentioned mechanisms, the analyzing task will become easier. And in the future, we will try to develop an automatic way to find root causes.

IV. EXPERIMENTS

We have built a prototype implementation of our alert reduction method using the Weka framework [15]. The prototype has been validated on DARPA 2000 dataset. And in order to assess the performance of our method in a real setting, we also applied it to real network data from our laboratory. We summarize the results obtained in this section. Before presenting the results, we will introduce the measures used in our experiments firstly. They are reduction rate, completeness and soundness. Their definitions are given as follows:

**Definition 4 (Reduction Rate)** Let \( N \) be the total number of alerts and \( N_f \) the number of alerts filtered by the systems. The reduction rate \( R_r \), which assesses how many alerts can be filtered by the system, is defined as the ratio between \( N_f \) and \( N \), i.e.,

\[
R_r = \frac{N_f}{N}.
\]

**Definition 5 (Completeness)** Let \( td \) be the number of true alerts correctly detected by the system, and \( tm \) the number of true alerts missed by the system. The completeness \( R_c \), which evaluates how well the system can detect true alerts, is defined as the ratio between \( td \) and the total number of true alerts, i.e.,

\[
R_c = \frac{td}{td + tm}.
\]

**Definition 6 (Soundness)** Let \( td \) be the number of true alerts correctly detected by the system, \( N \) the total number of alerts and \( N_f \) the number of alerts filtered by the systems. Then \( N-N_f \) means the number of candidate true alerts. The soundness \( R_s \), which measures how correctly the alerts are recommended, is defined as the ratio between \( td \) and the total number of candidate true alerts recommended by the system, i.e.,

\[
R_s = \frac{td}{N - N_f}.
\]

### A. Results Obtained from DARPA 2000 Data Set

DARPA 2000 is a famous synthetic data set for intrusion detection-related tests. It includes two scenarios: LLDOS 1.0 and LLDOS 2.0.2. Because the purpose of our experiment on DARPA 2000 is just verifying the feasibility and studying how to set the parameters, we did not use too many data so as to simplify our implementation. The dataset we used is phrase 2 to phrase 4 from DMZ in LLDOS 1.0. On this dataset, the IDS (we use RealSecure) can produce 886 alerts, which include 51 true alerts and 835 false positives. There are 45 attributes in each alert. And in our alert reduction method, we mainly use 5 of them for analysis. They are “alert type”, “source IP”, “source port”, “destination IP” and “destination port”.

Our experiment on DARPA 2000 is composed by two parts: Firstly we studied the effect of main parameters (including p, minisupport and weights of alert attributes) used in our model so as to choose appropriate values for them. And then we test how well our real-time framework work. **Experiments on Different Parameters.**
Effect of \( p \). Parameter \( p \) denotes the proportion of candidate true alerts in the whole dataset (re. Section III.B). It is mainly used in the learning phrase and determines the threshold of true alerts. In order to study how \( p \) influences the performance of our method, we change the value of \( p \) and obtain two groups of data. In this experiment, the weights of alert attributes are set to \( \{1, 0.6, 0.4, 0.8, 0.2\} \) (they are respectively the weights of “alert type”, “source IP”, “source port”, “destination IP”, and “destination port”), and the minisupport is set to 5%. Figure 3 shows the result:

From figure 3, we can see that it is difficult to achieve both high completeness and good soundness. Increasing of \( p \) leads to higher completeness, but the soundness often decreases too. In alert reduction systems, missing true alerts is usually more serious than incorrectly recommending false positives to users, so completeness is more important. In other words, the value of \( p \) should firstly ensure high completeness. Then on the base of it, the soundness and reduction rate should also be comparatively high. So according to figure 3, we think that the value between 15% and 20% is appropriate for \( p \).

Effect of minisupport. Parameter minisupport defines the minimal value of support for frequent patterns mining. Small minisupport will bring more frequent patterns, but the possibility that the patterns are mined from frequent attacks also grows. In order to study what value of minisupport is appropriate, we set it respectively to 5% and 3%. When minisupport was equal to 5%, the frequent patterns mined from the dataset did not contain any frequent attacks, while equal to 3%, they contained a frequent attack ‘Sadmind Amsverf Overflow’.

The final result of this experiment is shown in figure 4 (In this experiment, the weight is still \( \{1, 0.6, 0.4, 0.8, 0.2\} \)). From it, we can see that when minisupport is 3%, our method has better completeness and soundness. We believe that is because the feature set of false positives in this situation is more precise than that when minisupport is 5% (It should be noticed that this precision is based on removing all unsuitable features, i.e., patterns from frequent attacks).

Effect of Alert Attribute’s Weight. In order to verify our conclusion on the weight of alert attribute (re. Section III.A), we assigned three groups of weights for alerts and observed the corresponding results. In the first group, all alert attributes had the same weight values (i.e.1). In the second group, the weights of “alert type”, “destination IP”, “source IP”, “source port”, and “destination port” were respectively 1, 0.8, 0.6, 0.4, and 0.2. And in the third group, the weight of “alert type” was 1, the weight of “destination IP” and “source IP” were both 0.5, and the weight of “source port” and “destination port” were both 0.1. Then we set minisupport to 3%.

The result of this experiment (shown in figure 5) exactly proved our assumption. From it, we can see that after setting different weights for alert attributes according to their importance, both soundness and

![Figure 3](image_url)

![Figure 4](image_url)

![Figure 5](image_url)
reduction rate of our model are improved to some extent. And we are also surprised to find when the weights are respectively set to \(\{1, 0.6, 0.4, 0.8, 0.2\}\) and \(\{1, 0.5, 0.1, 0.5, 0.1\}\), the results obtained are very similar. We think that is caused by the characters of data in DARPA 2000. It also means our model is not very sensitive to parameters. And slight inaccuracy of the parameter’s value does not influence the result too much. This will make our model more practical.

**Efficiency of Our Real-time Framework.**

Based on above result, we also test the efficiency of our real-time framework on DARPA 2000. In the learning phase, we firstly regarded every 100 alerts as one block in the order of arrival. Then we mined each block for frequent patterns. After mining 4 blocks, the feature set of false positives was stable, so we stop collecting training data. 12 frequent patterns can be found after mining these 400 alerts. Because there is no frequent attack in them, all patterns were regarded as features. Then we set weights to \(\{1, 0.6, 0.4, 0.8, 0.2\}\), and calculated the FP score for these alerts. After sorting alerts ascendingly by their FP scores, we found that the maximal score in the top 15% alerts is 0.054. So the threshold of true alerts is set to 0.054.

In the online filtering phase, the remaining 485 alerts were input to our prototype one by one. We mined every 100 alerts so that the new feature of false positives can be found and added into the feature set in time. In our experiment, new patterns can be found only when mining the first 100 alerts. Among these patterns (i.e., “Alert Type = Admind”, “Alert Type = Sadmind_Amslverify_Overflow” and “Source IP = 202.077.162.213”), only the alerts that type of “Sadmind_Amslverify_Overflow” denoted a frequent attack, so the corresponding patterns (i.e., “Alert Type = Sadmind_Amslverify_Overflow”) can not be added into the feature set.

After 485 alerts had been processed, 70 alerts were recommended to users as true alerts. The reduction rate is 86%. Because 50 alerts among them were really true, and there are totally 51 true alerts in the dataset, the completeness of our prototype is 98%, and the soundness is 71%. However, after analyzing all recommended alerts, we found that the only true alert missed by our prototype was a duplicated one. In fact, when RealSecure detects an intruder breaks into a host by telnet, it will generate 3 types of alerts for this single attack step, i.e., “TelnetEnvAll”, “TelnetXdisplay” and “TelnetTerminaltype”. And we only missed the alert that type of “TelnetTerminaltype”. Users can still identify this attack step by the other two alerts. So the completeness of our prototype can be regarded as 100% considering this.

**B. Results Obtained from Real Data**

Due to various limitations of synthetic dataset [16], we have repeated our experiments on real-world IDS alerts. This dataset is collected over the period of 2 weeks in the network of our laboratory. The network includes 10 hosts, which connect to the Internet through one proxy server. A network-based IDS (snort) recorded information exchanged between the Internet and the intranet. There are totally 65528 alerts in this dataset, 95% of which are 4 types of alerts, i.e., “SNMP public access udp”, “SNMP request udp”, “(http_inspect) BARE BYTE UNICODE ENCODING” and “NETBIOS SMB IPCS unicode share access”.

In this experiment, we regarded 1000 alerts as one block, and used the same set of parameters as for DARPA 2000. By executing similar steps as in the previous paragraph, we obtained 5258 candidate true alerts finally, and the reduction rate is 92%. After thoroughly analyzing, we found that all alerts filtered by our prototype were false positives, so the completeness is 100%. Among the true alerts recommended by our prototype, the account of alerts triggered by real attacks was 2620, so the soundness is 50%. By comparing with the result obtained from DARPA 2000, we found that our model has higher reduction rate but lower soundness on real dataset. The likely explanation of this fact is that real dataset contains fewer intrusions and more redundancy than DARPA 2000. We believe that if we fit the parameters to the real data, the performance of our prototype can be even higher.

**C. Comparison with current methods**

In order to show the performance of our method, we have also compared it with other two typical alert reduction systems. The first one is the system presented by Klaus Julisch in [1, 3 and 4] (recorded as “Klaus’s method” in table II). It can identify false positives based on root cause analysis. And the second one is a system presented by Abdulrahman Alharby et al [6] (recorded as “Alharby’s method” in table II). It processes alerts based on continuous and discontinuous patterns and can filter false positives in real time. The result of comparison is shown in table II.

<table>
<thead>
<tr>
<th>Method</th>
<th>Reduction Rate</th>
<th>Completeness</th>
<th>Time Consumption</th>
<th>Real Time?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our method</td>
<td>86%-92%</td>
<td>100%</td>
<td>Low</td>
<td>Yes</td>
</tr>
<tr>
<td>Klaus’s method</td>
<td>87%</td>
<td>100%</td>
<td>High</td>
<td>No</td>
</tr>
<tr>
<td>Alharby’s method</td>
<td>80%</td>
<td>100%</td>
<td>Middle</td>
<td>Yes</td>
</tr>
</tbody>
</table>

From above table, we can find that the reduction rate of our method is a little better than Klaus’s method. However Klaus’s method is not able to filter alerts in real time. It can distinguish true alerts and false ones only after clustering all alerts, so its time consumption is high. Alharby’s method has the capability of real-time filtering, but its reduction rate is far lower than ours. Moreover, this method needs a lot of labeled data to build its model and can not filter alerts in training phase, while our method does not have these limits. So using our method, security managers can response to attacks more quickly.

In another word, the time consumption of our method is lowest. From above comparison, we believe that our system has better performance than current methods.
D. Discussion on our method

About the measures we used. It should be noticed that almost all of current alert reduction systems did not consider the soundness of their filtering results. However, soundness is an important measure, because low soundness will make users feel that the alert reduction system is unworthy of belief. And it is useful especially in the environment where the amount of alerts is large. In this situation, the alerts recommended to users are usually numerous. If the soundness is low, the recommendation that contains a lot of false positives will be useless to users. Considering this, we evaluated our prototype by this measure. On DARPA 2000 dataset, the soundness of our prototype is 71%. It is well but not perfect enough. And on real dataset, its soundness is even low, so we will do more research on improving it. In fact, the soundness of our model is mainly influenced by two factors. Firstly, the more accurate the feature set is, the higher soundness we can obtain (See in Fig 4, lower minisupport leads more features and higher soundness). In addition, it can also affected by how exact the FP score is (See in Fig 5, different weights cause different score and different soundness). So our improvement in the future will mainly focus on these two aspects.

About the method for setting weights. In our prototype, we calculated the value of weight by random sampling. This method has several advantages, for example, it is easy to implement and the weights it produced are very fit for current dataset. However, this method needs manual analysis of sample alerts. Fortunately, the amount of samples is little, so it is still better than the alert reduction methods that need a lot of labeled training data (e.g. method based on classification). Moreover, the calculation of weight value needs not be repeated for every dataset. In fact, the weights calculated by this method can be regarded as reference values of other dataset. And we have proved in section IV.B that even on different dataset, we can also get well result by the same parameter values.

About how can our framework achieve “real-time”. Our framework can process alerts in real-time, i.e. when a new alert coming, the framework can determine whether it is false positive or not almost immediately. This is implemented by following steps (re. section III.B): Firstly, based on a feature set of false positives, our framework calculates the FP score for each newcome alert. Secondly, the FP score is compared with the threshold of true alerts, so as to determine this alert is true or false. In order to finish above steps, a feature set, which can describe false positives accurately, and a threshold, which can distinguish true alerts with false one, have to be built in advance. These are just what the learning phrase of our framework needs to do. As far as the algorithm for calculating the FP score, before-mentioned experiments have proved that our algorithm is both accurate and with low time consumption. It is also an important pre-condition for real-time process.

About the adaptability of our model. In practice, the behaviors of attackers and normal users are both frequently changed. So the alert reduction system must be adaptive to this change, i.e. it should be able to identify the false positives triggered by new normal behaviors, and avoid to mistakenly filter out true alerts caused by new attacks. Our model has done some efforts in this aspect.

In order to identify new false positives, we designed a dynamic feature set (re. section III.A), i.e. with the arriving of new alerts, the new frequent patterns which represent new false positives are continually mined and put into the feature set, so that the accuracy of feature set can be kept. The experiment shown before has proved the effectiveness of this method. However it still has some limitations. For example, only when the amount of false positives triggered by new normal behaviors is larger than a predefined threshold, can their feature patterns be discovered. In order to identify these false positives more rapidly, we can choose low threshold. But this will possibly bring unrelated frequent patterns into feature set. So a trade-off has to be found between them.

As far as the change of attackers’ behaviors, we believe that it has little effect on our model. Because our model is based on profiling false positives by frequent attribute values. And any alert (no matter it has new type or not) which does not possess the features of false positives will be regarded as true alert. This is similar to the idea of anomaly detection in the field of IDS. In other words, new or unknown attacks can be identified by our model unless they have the same features with false positives. However even in the later situation, our model can also identify most of them by the algorithms for filtering frequent attacks (re. section III.B).

V. Conclusions

In this paper, we present a novel alert reduction method based on a new data mining technique, i.e., outlier detection. And we also build a real-time framework to filter false IDS alerts using this method. Now we have finished the prototype implementation of our model. And through the experiments on DARPA 2000, we have proved that when 86% of alerts have been filtered by our model, 100% of true alerts still remain. And on real-world dataset, our model has even higher reduction rate. By comparing with current other alert reduction methods, we believe that our model has better performance. However, our method also has some shortages, for example, now its soundness is still not perfect. In the future, we will study how to improve the soundness of our model. In addition, we will look for low-cost automatic method for setting weights and do more experiments on real-world IDS alerts so as to know how our model works with alerts generated by multiple IDSs.

ACKNOWLEDGMENT

This work is supported in part by the High Technology Research Project of Jiangsu Province of China under Grant No.BG2005029, and in part by the
Natural Science Foundation of Jiangsu Province under Grant No.BK2007136.

REFERENCES


Fu Xiao, born in Jiangsu Province, China and on 25th December, 1979. She is a Ph. D. candidate of Department of Computer Science and Technology, Nanjing University, Nanjing, China. Her main research interests include network security and machine learning. And she has published several papers in the areas of network security and machine learning.

Shi Jin, born in Anhui, China and on 4th July, 1976, and received the Ph.D. degree in computer science and technology from the Nanjing University, Nanjing, China, in 2008. He is an Assistant Professor in the School of State Secrecy and Department of Information Management, Nanjing University, where his research focuses on network security, privacy protection. He published over 20 papers in the areas of network security and privacy protection.

Xie Li, born in 1942, professor and Ph. D. Supervisor of Computer Science and Technology, Nanjing University, Nanjing, China. His current research interests include distributed computing and advanced operation system.
Measuring the botnet using the second character of bots

Zhitang Li
School of computer science and technology, Huazhong University of Science and Technology, Wuhan, China
Email: junehu1210@gmail.com

Jun Hu, Zhengbing Hu, Bingbing Wang, Liang Tang and Xin Yi
School of computer science and technology, Huazhong University of Science and Technology, Wuhan, China
Email: {leeying, hzb}@mail.hust.edu.cn

Abstract—Botnets have become one of the most serious threats to the Internet. They are now the new key platform for many Internet attacks, such as spam, distributed denial-of-service (DDoS), and we call these attacks “the second character of bots”. In this paper, we focus on characterizing spamming botnets by leveraging both spam payload and spam nodes traffic properties. Measurement of botnets is an important and challenging work. However, most existing approaches work only on specific botnet command and control (C&C) protocols (e.g., IRC) and structures (e.g., centralized). In this paper, we present two measurement frameworks (MFNL and MFAL) that based on the second character of bots to measure the size of the botnet. We have easily implemented our prototype system and evaluated it using many real network traces, and we also compare these two approaches from several points.

Index Terms—botnet, SMTP, spam, size, MFNL, MFAL

I. INTRODUCTION

It is widely accepted that botnet poses one of the most significant threats to the Internet [13]. But the size of the botnet continues to be an issue of debate among the research community [2, 3, 4, 5, 6, 7]. The recent measurement approaches can be divided into two categories: active measuring and passive measuring.

In the active measuring area, most researchers use honeynets and crawlers to infiltrate the botnet getting some information of the peers and the state of the network, by running an instance of the botnet and developing a crawler using specific protocol queries, they can collect some information, such as node IDs, IPs, and port numbers. The advantage of this method is that it can directly get the topology of the network and has a more correct result, but the crawler is based on the protocol, so it need a priori knowledge of botnet(such as captured bot binaries and the protocol the botnet using), so it will be difficult to operate on a new botnet, especially on the encrypted botnet.

In the passive measurement area, the monitors are fixed on the edge of the backbone or the core routers, or the bound of the ISP, the monitors can get the node numbers and the flow character etc. Using these overview flow information, adopt suitable methods, they can measure the botnet. The advantage of this method is that it needs no a priori knowledge, but with a high false positive rate. We pick out some typical papers in the both areas and analyze them respectively below.

M.A. Rajab [1] pointed out that botnet size can be the size of footprint or the size of live population. According to different definitions, the size varies with different measurement methods. For example, earlier studies have proposed a number of techniques to measure the size of botnets which led to different results. Dagon [2] established that botnet size could reach 350,000, but Rajab [5] indicated that the effective size of botnets rarely exceeded a few thousand.

In this paper, we attempt to estimate the live population of the botnet, including IRC botnet and P2P botnet. Two new methods are proposed to measure the size of the botnet.

Early study primarily focused on IRC botnet. At present, there are a number of mechanisms for estimating botnet sizes, but there are some limitations in each of them. The most direct method is to infiltrate the botnet by joining the command and control channel [5]. Because of its simplicity, this technique suffers from bot cloning and temporary migration of bots. So it is difficult to provide an accurate bot count in these cases, because there is no significant difference between actual bots and temporary clones or migrants.

Dagon [2] developed a technique for counting infected bots by manipulating the DNS entry associated with a botnet’s IRC server and redirecting connections to a local sinkhole. This technique can only measure the botnet’s footprint, but knows nothing about whether the bots belong to the same botnet. Moheeb [5] explored the use of DNS cache snooping to uncover a botnet’s footprint. But this technique requires DNS servers that...
Botnets have been widely used for sending spam emails at a large scale [6, 11]. Ramachandran [11] performed a large scale study on the network behavior of spammers, providing strong evidence that botnets are commonly used as a platform for sending spam. Almost every bot in a botnet sends spam. Spam senders belong to different zombie networks, serve different controllers for different purposes and send spam with different content. But if they belong to the same botnet and serve the same controller, they may send spam with the same or similar content in the favor of the controller. Since most bots are family computers, they only run a few hours every day. However, they have to complete sending a certain amount of junk emails in a short period of time.

### III. Case Study: Mybot

<table>
<thead>
<tr>
<th>TABLE I. RESULT OF THE SPECIMEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mybot A</td>
</tr>
<tr>
<td>IP</td>
</tr>
<tr>
<td>Time of the packets</td>
</tr>
<tr>
<td>Total packets num</td>
</tr>
<tr>
<td>SMTP packets num</td>
</tr>
<tr>
<td>Ratio of SMTP packets</td>
</tr>
<tr>
<td>Frequency of SMTP packets</td>
</tr>
<tr>
<td>(num/minute)</td>
</tr>
<tr>
<td>Spam num</td>
</tr>
<tr>
<td>Spam sending frequency</td>
</tr>
<tr>
<td>(num/minute)</td>
</tr>
<tr>
<td>URLs</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Mybot is the most known IRC bot and is the one using a completely centralized IRC protocol. We monitored the spamming behavior of the bot in our test environment. Our test environment is setup in a secure environment to avoid infecting other machines or participating in
malicious attacks. We used a botnet specimen for our case study. Table I shows some observed results of the specimen. In our study, we take one SMTP session as a successful spam. Fig. 2 shows the whole SMTP session.

From Table I, we find that the bot has a high unusual SMTP frequency and sent a large number of spam than normal users. In about 90 minutes it has sent more than 1000 spam, but the URLs in all these spam is only five. We run the same specimen on Mybot A and Mybot B, so they belong to the same botnet, from the Table I, we find the spam has the same URLs, this proved our previous inference.

IV. MEASURING FROM THE NETWORK LEVEL (MFNL)

---BY MONITORING GROUP ACTIVITIES IN SMTP TRAFFIC

A. SMTP Decode Module

In the last section, through the analysis of the botnet specimen, we find that in a short time the bot sent a lot of SMTP packets, and sent a large number of spam than normal users, and the spam sent by a bot specimen has the same or similar content. According to these characteristics, this paper presents a botnet measurement technology based on the analysis of the SMTP packets. The whole process can be described as Fig. 3.

The whole system contains two modules: SMTP decode module and clustering module. In the SMTP decode module we extract all the SMTP packets from the network traffic, statistic the SMTP frequency, the number of spam each IP sent and extract the URLs of the spam. If the values exceed the threshold we will mark the IP as a suspicious bot.

Approximately 70 billion emails are sent each day, half of which are spam[17]. Supposing 10 billion of the spam is sent by 150 million bots, each bot must send at least 60 spam a day on average. Ordinary users only keep their computers on for a few hours each day, so such a bot need to send 50 or more spam per hour. By observing the activity of a sample of mybot has proved the inference, which sent 1144 spam in 90 minutes.

In the cluster module, we cluster them into different spam campaign based on the same or similar URLs, each campaign consider belonging to the same botnet.

```
1  for each packet
2  |
3    if (SMTP)
4      get the source IP;
5    if (is a new email session)
6    |
7      (the num of the spam sending by this source ip)++;
8    |
9  |
10    if (time out)
11    |
12    for each IP
13    |
14      statistic the num of the spam it has sent;
15    if (count>THRESHOLD)
16    |
17      the ip is a suspicious bot;
18    |
19  |
20  }
```

B. Cluster the spam campaigns

From the last section and the case study of mybot, we know that if the bots belong to the same botnet. They will send spam with the same or similar URLs, because of the bot controller’s profit. So when we extract the URLs from the SMTP packets, we cluster the same and similar URLs to a spam campaign, all the IP in this campaign are considered in the same botnet, then we can easily get the size of this botnet.
If the URLs we extracted are short, we can just use string comparison algorithm to cluster them into a spam campaign. But for long and complicated URLs we defined entropy reduction, leverages information theory to quantify the probability of a coming URL matching a stored URL (signature). Given a coming URL \( e \), let 
\[ R_e(x) \]

denote the expected number of bits used to encode a random string \( x \) with the signature. 
\[ R(x) \]

Denote the expected number of bits used to encode a random string \( x \) without the signature. 

The entropy reduction \( d(e) \) is defined as the difference between \( R_e(x) \) and \( R(x) \), for example, 
\[ d_e = R_e(x) - R(x) \]. The similarity is defined as 
\[ P(e) = \frac{2^{R_e(x)}}{2^{R(x)}} = \frac{1}{2^{d(e)}} \]  \hfill (1)

Given a regular expression \( e \), if \( d(e) \) is small, \( P(e) \) tends to be large, which means \( e \) is similar with the stored URL.

If the URLs in the spam sent by one IP is the same or similar with the URLs in the spam sent by another IP, we will cluster them into a spam campaign. All the IPs in the same campaign belongs to a botnet. The number of the IP is the size of the botnet.

C. Experimental evaluation

Fig. 4 shows our test environment which is setup in a secure environment to avoid infecting other machine or participating in malicious attacks. Our test environment setup consists of one software firewall installed with IPCop. Five computers, installed with Windows XP SP2 and the subnet E which used by the normal users are all connected through a switch to the WAN. One host will be logging all the traffic with Wireshark. The other four will serve as bots in our botnet. The infected PC A and PC B have run the IRC specimen (mybot.exe), infected PC C and PC D have run the SDbot.exe.

We captured half an hour data of A, B, C, D, E separately in the morning, noon, and evening for our experiments. Fig. 5 shows the spam num/half-hour of each. From Fig. 5 we find that the bots sent a sharp large num of spam than the normal users.

The Fig. 6 shows part of the measurement results in the test environment. The 218.199.92.69 and 218.199.92.68 were in the same botnet. It fits well with our test environment. Using our system, the 218.199.92.71 and 218.199.92.72 were clustered to another same botnet too.

<table>
<thead>
<tr>
<th>URL</th>
<th>IP</th>
<th>SCALE</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://fakeslick.com">http://fakeslick.com</a></td>
<td>218.199.92.68</td>
<td></td>
</tr>
<tr>
<td><a href="http://loulybrool.com">http://loulybrool.com</a></td>
<td>218.199.92.68</td>
<td></td>
</tr>
<tr>
<td><a href="http://rosqaise.com">http://rosqaise.com</a></td>
<td>218.199.92.68</td>
<td></td>
</tr>
<tr>
<td><a href="http://slicksick.com">http://slicksick.com</a></td>
<td>218.199.92.68</td>
<td></td>
</tr>
<tr>
<td><a href="http://treatlove.com">http://treatlove.com</a></td>
<td>218.199.92.68</td>
<td></td>
</tr>
</tbody>
</table>

For further study, we have some interesting findings; the most interesting finding is the fraction of the target IP of the bots. From Fig. 7 we find Mybot and SDbot has different target IPs. This is because different botnet was rented by different merchant for different purpose. But Mybot A and Mybot B almost have the same distribution; SDbot C and SDbot D have the same situation. This is because bots in the same botnet have the same instructions from their botmaster. They may get the spam and the sending list from the same place.

MFNL can be fixed on the edge of the backbone or the core routers, or the bound of the ISP. It will be very effective and sensitive to the nodes which sent a lot of spam in a short time. For most SMTP packets are not encrypted now, so it will work well to get the size of a botnet in a local area network.
In the last section we have proposed our approach from the network level to measure the botnet. This section we will present another approach to measure the botnet size by using URL and collaborative Mailservers.

A. Overview of the Systems

Botnets have been widely used for sending spam emails at a large scale [6, 11]. Ramachandran [11] performed a large scale study on the network behavior of spammers, providing strong evidence that botnets are commonly used as platform for sending spam. Almost every bot in a botnet sends spam. Spam senders belong to different zombie networks, serve different controllers for different purposes and send spam with different content. But if they belong to the same botnet and serve the same controller, they may send spam with the same or similar content in the favor of the controller. Since most bots are family computers, they only boot a few hours every day. However, they have to complete sending a certain amount of junk emails task, usually a large number of messages in a short period of time. From section III by observing the activity of a sample of Mybot proved the inference, which sent more than 1000 spam in about 90 minutes. So if we find an IP sending a large number of emails in a certain time, we think it is a suspected bot, and we will store the content and query the Mail Servers to find out all the IPs sending emails with the same content. Those IPs are considered to be the members of the same botnet. We can get the size of this botnet based on the number of IPs. In the same way we can roughly determine the size of different botnets. In our method, we first get the IP sending large num of spam. Then we compare the content of the spam and highlight the URLs embedded in email content, because URLs play an important role in directing users to phishing web pages or targeted product web sites. In this way, we can almost get all the members of a botnet [2].

B. Architecture of MFAL

MFAL, is a distributed system that identifies the source of each e-mail and then stores this information in a distributed database that is used and updated by all peers. MFAL depends on the collaboration of peers. Fig. 9 shows the architecture of the MFAL.

MFAL consists of several distinct parts: identifying the source of e-mails, keeping track of how many e-mails were recently sent by a source, and disseminating this information for the purposes of statisticizing the size of the botnet. Additionally, these tasks must be coordinated for each e-mail as it is received by an MS. Thus, MFAL comprises four parts that perform the respective functions.

First when an e-mail arrives at a MS, the message is passed to the source identification module to determine the source of the e-mail.

Second, MFAL queries the distributed database about the number of e-mails that were recently sent by a source, and disseminating this information for the purposes of statisticizing the size of the botnet. Additionally, these tasks must be coordinated for each e-mail as it is received by an MS. Thus, MFAL comprises four parts that perform the respective functions.

First when an e-mail arrives at a MS, the message is passed to the source identification module to determine the source of the e-mail.

Second, MFAL queries the distributed database about the number of e-mails that were recently sent by a source, and disseminating this information for the purposes of statisticizing the size of the botnet. Additionally, these tasks must be coordinated for each e-mail as it is received by an MS. Thus, MFAL comprises four parts that perform the respective functions.

Third, if the num of spam sending by this source cross the threshold, MFAL extract and store the URLs. Then get the size of the botnet.
When receiving the emails, MS will extract the following information: URL string, source IP address and mail-sending time. The most challenging thing is source identification since the email-senders can be faked.

We assume that once the email is received by a trusted relay, the received lines added by these relays are correct. That means the received line added by the first trusted relay, identifies the entrusted host that injected the email. The host is considered to be the source of the email. The key challenge then is to identify this first trusted relay, and its corresponding received line, which contains the defacto IP address of the sender. But the bots typically do not have static IP address. Thus, we consider that a relay is trusted if it has a static IP address. As pointed out in [8], we can find the first trusted relay from the MX entry and several approaches. When we find the ingress relay, we check from the received line corresponding to the ingress relay, using several heuristics to get the source of the email.

Many spam messages are not sent from botnets, so we can not recognize all the IP sending large number of emails as suspicious bots. We need to remove the static IP which is notorious in blacklist, such as popular proxies, open relays, etc. If the sender IP address of a message is one of these, we exclude that IP from further analysis.

When we get the real IP, we need to track the number of the emails the IP sent. Since a bot does not necessarily target multiple recipients within a single domain, it will need a collaborative effort to determine whether an IP is a suspicious bot.

Assuming that $MS_1$ received an email, first we extract the IP of the email, then query the number of the emails sent by this IP on the $MS_1$ in an hour, and record the num as $n_1$;

Then, we query the number of the emails sent in the same hour by this IP on the $MS_2$ and record it as $n_2$;

We query the rest of the MSs, and record the number of emails as $n_3, n_4, ..., n_n$.

So we can get the total number of the emails sent by this IP in an hour, recorded as $N$.

$$N = n_1 + n_2 + ... + n_n;$$

If $N$ is large and crosses a certain threshold, which is 50 in the present study, we will mark this IP as a suspicious bot.

When we find the suspected bots, we store the following information on the MS for later query.

<table>
<thead>
<tr>
<th>IP</th>
<th>TIME</th>
<th>CONTENT</th>
</tr>
</thead>
</table>

From section III we know that in a botnet each bot sends the same or similar content emails for controllers’ benefit. If we record the content sent by this IP, we know exactly what is sent by this botnet. Then we can easily get the size of the botnet. We just need to query the MSs to get all the other ips which send the above content. We can surely say that the IP sending the same content is a member of the same botnet. When we get all the members of the botnet, we can know the exact size of the botnet.

**D. IP SOURCE IDENTIFICATION AND TRACKING**

Since it is believed that a fraction of bots send very few spam each day[11], it’s not accurate to consider the number of IP which sent a large amount of email in a short period of time (in this paper we defined it as an hour) as the size of the botnet. However, from these statistics we can know exactly what the botnet is sending. The sending content of an IP may be more than one, so we need to query each MS to get all the IP sending the same content in a period. This period can be decided by the life-cycle of the botnet (several days).

Botnets are now used for a commercial purpose to attract users to buy the goods and services in the spam. So the same or similar content email will be sent to thousands or maybe millions of users by bots. Although the recent botnets are very smart, and they can be customized for different users with different mails, it is impossible for spammers to produce thousands of the same meaningful spam. Therefore, different versions of the spam should be very similar to each other with minor changes. In particular, different versions of the spam will contain many of the same sentences, especially the URLs.

Fig. 10 shows an example of three emails sharing the highlighted URL, but mixed with a number of other information. The example came from a sample of Peacomm (a P2P botnet), and we collected the SMTP (Simple Mail Transfer Protocol) data every 15 minutes using Wireshark.
The URLs cannot be easily changed: users must be able to correctly contact the spammers if the spam is to be of any use to the spammers. So the URL can be used as a valuable resource to compare the content of the spam. URL is very important for a botnet to achieve the purpose, and all bots will send the spam containing the same URL.

In the section IV We defined entropy reduction, leverages information theory to quantify the probability of a coming URL matching a stored URL (signature) to cluster the spam with the same or similar URLs into a spam campaign. Here we can use the same method as section IV, then statistic the number of IPs in the campaign to get the size of a botnet.

E. IP DYNAMIC AND ESTIMATING BOTNET SIZE

When We get all the IP in a life cycle of the botnet, we still can not fully determine the size of the botnet because many home computers are connected to the internet through ADSL, cable or other devices which make the user’s IP changes daily or even hourly [12]. Zhuang [10] has proposed an effective method to estimate IP dynamics. After IP dynamic treatment, we can count the number of distinct machines, and then obtain the size of the botnet.

Our study is based on the spam sent by a sample of Peacomm (one of the P2P botnet) and the emails collected from HUST mail server. In our dataset, a total of 103 emails containing the highlighted URL were sent on the same day from 70 different IP address. We suspect the corresponding hosts were from the same botnet.

In order to measure the whole size of the botnet, we defined \( \alpha_i \) as the ratio of the number of the users of MS\(^i\) and HUST Mail Server. We defined \( k_i \) as the number of the users of MS\(^i\), and \( k_h \) as the number of the users of HUST Mail Server. So \( \alpha_i \) can be denoted as:

\[
\alpha_i = \frac{k_i}{k_h}. \tag{3}
\]

Where \( k_i \) and \( k_h \) always be a constant, we can get them easily. So we can roughly get the size of the botnet when we get the distinct IP \( \sum_{i=1}^{n} \alpha_i \cdot B_h \) on HUST Mail Server. We defined \( B \) as the total member of the botnet, then

\[
B = \sum_{i=1}^{n} \alpha_i \cdot B_h
\]

By using this method in our experiment, most botnets have hundreds to thousands of IP addresses; this result is consistent with other’s research. Since our experiment is simple and only uses sampled emails, the reported botnet sizes are expected to be much smaller than the actual sizes.

MFAL can be fixed as a plug in on the Mail servers to achieve all of its functions.

VI. COMPAREMENT OF MFNL AND MFAL

We have proposed two approaches to measure the size of the botnet. Each of them has their own advantages and disadvantages.

MFNL can be fixed on the edge of the backbone or the core routers, or the bound of the ISP. It will be very effective and sensitive to the nodes which sent a lot of spam in a short time. For most SMTP packets are not encrypted now, so it will work well to get the size of a botnet in a local area network. It is the MFNL’s advantage that MFNL is easy to fix and have high accuracy. But it can only get parts of the results, because it is applicable to the LAN. However, the target of the botnet permeates many countries. Another disadvantage is that MFNL will be ineffective when the threshold is low, however today’s botnet gets more ‘cleverer’ than before. For escaping the detection, botmaster use some tactics to design the botnet making them hard to track. One of these tactics is that make parts of the bots’s spamming behavior not so obviously, for example, sending less spam in a short time to hide themselves. MFNL will not be so sensitive to these bots.

MFAL is a collaborative system, which has a global view, while MFNL just has a local view. And MFAL can get all the bots whatever they sent a lot of spam or less spam. But MFAL is hard to carry out in the real world and IP source identification is the most challenge thing.

We find that the advantage of MFNL is the disadvantage of the MFAL, while the disadvantage of MFNL is the advantage of MFAL. So maybe we use them together, we can get a better result.

VII CONLUSION AND FUTURE WORK

We have proposed two measurement frameworks (MFNL and MFAL) that based on the second character of bots to measure the size of the botnet. We have easily implemented our prototype system and evaluated it using many real network traces. And we also compare these two approaches from several points. At last we conclude that if we want to get a much better result we need to use the two approaches together.

In the future we want to make the MFNL and MFAL work together to get a better result.

ACKNOWLEDGMENT

© 2010 ACADEMY PUBLISHER
This work is supported by the National Natural Science Foundation of China under Grant No. 60573120, and by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2007AA01Z420, and by the Key project sponsored by Natural Science Foundation of Hubei Province under Grant No. 2008CDA021.

REFERENCES


Control Strategy of Scheduling Invalidation of Data Broadcasting

Hu Wen-bin\(^1\), Hu Zhengbing\(^2\), Yang Fu-qiang\(^1\), Fan Cun-lian\(^1\), Wang Lei\(^1\), Tan Xin-long\(^1\)

\(^1\) (Computer School, Wuhan University\(^1\), Wuhan, 430079)
\(^2\) (State Key Lab. For Novel Software Technology, Nanjing University\(^2\), P.R.China)
\(^3\) (Department of Information Technology, Huazhong Normal University\(^3\))
(hwb77129@126.com)

Abstract - With the number increment of mobile computing equipment, the invalidation of equipment and broadcasting information because of wireless bandwidth, communication quality and too long scheduling queue that badly influence the system computing performance. Mobile cache is a good technology to solute this difficult problem. Considering the priority problem of broadcasting queue generated by data invalidation cache, this paper designs a multi-buffer queue-scheduling model, which classifies the priority of broadcasting data. An invalidation control strategy of data broadcasting scheduling based on mobile Agent and IBM Application Response Measurement (ARM) is presented to guarantee data executing on the terminal in the right time. The results of experiments show that the invalidation rate and average work response time of the presented model and strategy in this paper are less (20\%) than the single buffer queue mode and random invalidation controlling strategy.

Index Terms - invalidation, rate of invalidation, strategy, data broadcasting, schedule

I. INTRODUCTION

Mobile computing is the hot issues of the research and application. The mobile computing technology makes it possible that users have access to information at any time and place, it is becoming a popular method for calculating, compared with traditional computer networks, the wireless communications which support for mobile computing, have the shortcomings of small bandwidth and poor quality of communication. In order To support a large number of mobile computing devices to visit sever data concurrently in real time, people make this mode of transmission, data broadcasting. Data broadcasting possesses a very good application prospect in information publishment and military commander. At present, most of the researchers pay attention to study how a sever can get the broadcasting sequence by Access Probabilities, the most appropriate for mobile computing, and they have proposed a series of dispatch algorithm\(^1\) and achieved a great deal of research results, but with the rapid development of mobile computing Technology which makes the mobile users becoming more and more, the Asymmetry of communication uplink and downlink and the frequent disconnection and connection of Communication link bring the Data organization and distribution in mobile computing environment a great challenge.

The system’s scalability makes mobile computing environment achieving the required computing power by expanding the size of the system, but with the increase of mobile computing devices, equipment failure and the distribution of scheduling information failure on mobile computing performance have become an unavoidable problem of large-scale mobile computing systems’ normal and reasonable running\(^2\). Invalidation control for scheduling of data broadcasting in mobile computing environments including two parts: ①Invalidation control for data loss in data broadcasting; ②Invalidation control strategy of mobile computing devices. In a mobile computing environment, because a computing operation’s several broadcasting tasks were running in different groups of mobile computing equipment independently, they continue to communicate in the running process, when one broadcasting task of a computing operation stop, the others are forced to stop, so invalidation control for scheduling of data broadcasting will affect the all mobile computing tasks.

In short, if in the process of data broadcasting scheduling, we can obtain the time and position of mobile equipment invalidation, and cache and control the failure scheduling information by handing over these two phenomena above, combining with a strategy of failure scheduling control. It will improve mobile computing system’s reliability, robustness and efficiency largely. To this problem, this paper designs a multi-buffer queue-scheduling model, and an invalidation control strategy of data broadcasting scheduling based on mobile Agent and IBM Application Response Measurement (ARM) is presented to guarantee data executing on the terminal in the right time.

II. QUESTION DERIVATION

In mobile computing environment, an important characteristic is that mobile computing device frequently separate with computing sever voluntarily and not voluntarily. In the disconnected period, mobile computing devices will continue to work. Mobile computing device sends out read-write operation to sever data, the mobile computing device can discover its data requested(broadcasting data) has invalidated only after connecting again(effectively). At this time the broadcasting data in the system is the next round data
which will be send by sever. If then this mobile computing device makes use of this data as primary data, it will cause the mobile computing task’s wrong running [3]. In order to solve this problem, paper [3] [4] propose the multisession-validation information (MV-VI) data broadcast mechanism. During broadcast cycle, the strategy designs a multi-buffer queue-scheduling model (VI) of transactions. The fast invalidation reports (FIRs) are inserted after data broadcasting to broadcast frequently updated data. But because this control strategy request to back up every broadcasting data, and let the sever save every data’s all backup version, it brings very tremendous pressure to the sever, and to the mobile computing system of arduous data broadcasting duty, it will cause the sever overload and so on. In addition, sever save every data’s several versions, so it may cause the problem of version’s choice and coverage, and if once versions are chosen or covered wrongly, it will also result in mobile computing tasks’ running errors. Because various editions’ data actually exist priority order, judging from the current domestic and foreign-related research, all of them adopt the general cache method to realize broadcasting data, but don’t take the broadcast queue priority into consideration after invalidation data being cached. To this problem, this paper designs a multi-buffer queue-scheduling model, through multi-buffer queue this model re-classify broadcasting data based on priority.

III. A MULTI-BUFFER QUEUE-SCHEDULING MODEL

A. multi-buffer queue-scheduling model designs

Traditional scheduling model takes a single-queue scheduling buffer (Figure 1). After sever receive the data and compute the scheduling sequence, all data are ranked in the same queue according to priority. Customers wait for a long time, have the high failure rate, and don’t deal with the failure request data, so it is difficult to meet customers’ demand.

In order to solve the priority problems which the time cost of early invalidation data brings about and mobile computing devices themselves brings out, This paper designs a multi-buffer queue-scheduling model (Figure 2), each of the buffer queue have their own priority, the data to be dealt with will enter a different queen in accordance with their priority when it is analyzed by the algorithm. Set data in high-priority queue having a high probability to be broadcasted than data in low-priority queue, so that we can allow an urgent data to be broadcasted earlier. Sever determine the invalidation data periodically based on the response time and cut-off time of customer’s request data. The invalidation data will be re-released back to the high-priority queue and wait for re-broadcasting after it is marked and given new deadline. When the data which has been ineffective invalidate in the next cycle, it will be treated as second-failure data. Sever will delete the second-failure data and send a failure request message to the customer.

B. Scheduling model validity proof

This paper considers that there are \( \lambda \) shares of data per second, sever handle a share of data per second in average but it can only deal with a share of data once. Establish a random service model of limited resources, the stability of the system is described as:

\[
\lambda = \rho \sum_{n=0}^{N} \rho^n P_0
\]

When the system is in \( P_0 \) state, it changes to \( P_1 \) after it receives \( \lambda \) shares of data. As the same, the system changes to \( P_0 \) state after \( \mu \) shares of data quit it. For any state, switching-in rate is equal to switching-out rate; we get the system stable probability equation:

- To the 0 state: \( \lambda P_0 = \mu P_1 \)
- To the 1 state: \( \lambda P_0 + \mu P_2 = (\lambda + \mu)P_1 \)
- To the N-1 state: \( \lambda P_{N-1} + \mu P_N = (\lambda + \mu)P_{N-1} \)
- To the N state: \( \lambda P_{N-1} + \mu P_N = (\lambda + \mu)P_N \)

Stable equation:

\[
\begin{align*}
P_1 &= \rho P_0 \\
P_2 &= \rho^2 P_2 \\
P_N &= \rho^N P_0 \\
\sum_{n=0}^{N} \rho^n P_0 &= 1
\end{align*}
\]
\[
\rho = \frac{\lambda}{\mu}
\]

is broadcasting data’s volume fraction and the

\[
P_0 = \frac{1}{\sum_{n=0}^{N} \rho^n}
\]

first state probability is: \(B\). So in the system, the expectation value of broadcasting data quantity

\[
L = \sum_{n=0}^{N} \rho^n P_0
\]

is:

\[
L = \frac{\rho}{1 - \rho} \frac{(N + 1)\rho^{N+1}}{1 - \rho^{N+1}}, \quad (\rho \neq 1)
\]

When the buffer is full and equal to N, effective data rate of arrival is:

\[
\lambda_e = \sum_{n=0}^{N} \lambda_n P_n
\]

\(\lambda_e = \lambda(1 - P_N)\). In the buffer the average amount of broadcasting data which are waiting processing is:

\[
L_q = \frac{\lambda}{\mu}, \quad q = \frac{\lambda(1 - P_N)}{\mu}
\]

Every data’s average residence time in the server is:

\[
W = \frac{L}{\lambda_e}, \quad W = \frac{L}{\lambda(1 - P_N)}.
\]

Every data’s average latency in the buffer queue is:

\[
W_q = W - \frac{1}{\mu}, \quad W_q = \frac{L}{\lambda(1 - P_N)} - \frac{1}{\mu}.
\]

Under the limited invalidation time of buffer queue, the smaller \(W_q\) is, the lower failure rate is, so we need to reduce failure rate by reducing \(W_q\). Therefore this paper establishes priorities in the buffer queue model: supposing that the broadcasting frequency is divided according to 70% broadcasting data. The data of high frequency has high priority, and the data of low frequency has low priority. In each of the priority, service order is based on first-come-first-server basis. In addition, considering that the request to user data may be overtime, this paper assumes that once the data broadcasting is overtime, the data will be added into high-priority queue immediately and be ready to be repeated. If the data fails to be send at the second time, it will not be dealt with and the sever feedback the cause of failure to user.

The input of scheduling of data broadcasting is \(\lambda\) ‘s Poisson flow. Each time there is a variety of data to reach and arrange in accordance with the queue’s priority. The distillation function for the processing time of each queue is:

\[
G(t), G_2(t), \cdots, G_n(t)
\]

So

\[
0 < \frac{1}{\mu} \int_{0}^{\infty} dG_i(t) < \infty \quad (i = 1, 2, \cdots, n)
\]

\[
\beta_i = \int_{0}^{\infty} t dG_i(t) < \infty \quad (i = 1, 2, \cdots, n)
\]

\[
g_i(s) = \int_{0}^{\infty} e^{-st} dG_i(t) \quad (i = 1, 2, \cdots, n)
\]

Comparing with the queue system without priority, the system has only one type of data, the input parameters is \(\lambda\) ‘s Poisson flow. Services in descending order, and the time distribution for service is: \(G(t) = G_1(t) + G_2(t) + \cdots + G_n(t)\). The average residence time and average latency is:

\[
W = \frac{L}{\lambda(1 - P_N)}, \quad W_q = \frac{L}{\lambda(1 - P_N)} - \frac{1}{\mu}.
\]

In multi-buffer queue model which considers the priority, each data’s average residence time and average latency in the sever which this paper calculate before is (two cases for the buffer queue):

\[
\rho_1 = \frac{\lambda}{\mu_1}, \quad \rho_2 = \frac{\lambda}{\mu_2}
\]

High-priority queue server’s average residence time:

\[
W_1 = \frac{1}{\mu} \frac{\lambda \beta_1}{2(1 - \rho_1)}
\]

low-priority queue server’s average latency:

\[
W_q = \frac{\lambda \beta_1}{2(1 - \rho_1)}\]

According to the time of calculation, there are two cases for the buffer queue:

\[
W > 0.7W_1 + 0.3W_2, \quad W_q > 0.7W_q + 0.3W_q.\]

From the above proof, we can see that Adding priority queue (multi-buffer queue) to the buffer queue will effectively reduce the failure rate of data broadcasting, and we can likewise prove the same conclusion for more than two buffer queues.

IV. INVALIDATION CONTROL STRATEGY

IBM Application Response Measurement provides the request measure value information and save it to the journal file to provide the procedure gain and analysis, or transmits to the Application Response Measurement proxy procedure, the request measure value provides the response time to each Application request through ARM API. This paper with the aid of the characteristic of IBM ARM provides the response time and the request measure value to each task, an invalidation control strategy of data broadcasting scheduling based on mobile Agent and IBM Application Response Measurement (ARM) is presented, each ARM from the realization of a mobile Agent. When a new broadcast task produces, the server ARM registration center produces a registered ARM Agent, and produces the directory index of this ARM, this ARM will be the "attachment" in the broadcast data, it records the execution situation of the broadcast data, such as: the execution time, the standby period, waiting for the result of the cost of time, whether to expire and so on. Concrete invalidation control strategy is as follows (Figure 3):

(1) Mobile computing system produces a new broadcast task: \(Data_A\), the mobile computation server system perceives the new duty’s production, registers and produces a registration number in the ARM registration center, this registration number record this broadcast
Fig. 3 Invalidation controlling strategy based on ARM

2) Producing an ARM Agent Relied on this registration number, this Agent adheres to DataA forms a new broadcast data DataAARM, this Agent in front of DataA data, guiding the data’s scheduling and execution.

3) Through the recorded DataA’s broadcast cut-off time, the broadcast execution terminal collection’s situation to calculate this broadcast task’s priority, and to enter into the multi-buffer queue for queuing.

4) Once enter the buffer queue, ARM Agent records the DataA’s waiting time, and compares this waiting time whether to surpass the cut-off time of this task execution, the mobile computing terminal information which feeds back through other duty's ARM Agent judges this duty whether to exist the execution terminal invalidation question, and gives the data and the terminal invalidation diary.

5) The invalidation diary feeds back through the mobile communication upward channel to the mobile computation server, the server through the feedback expiration diary restarts or adjusts the broadcast duty (adjusts cut-off time, increases time cost, raises the priority level, changes execution terminal and so on).

6) The broadcast task which does not become invalid (data invalidation and execution terminal invalidation) will be broadcast, in the process of broadcast ARM Agent records the broadcast data’s execution situation: The execution time, the execution effect (whether or not execute in the execution terminal correctly) and so on.

7) The broadcast data which has not been able to execute correctly will be recorded in ARM Agent, when the DataA task has been completed, will check the DataA’s ARM Agent diary, the task which invalids or has not been broadcast completely will be calculated the invalidation cost, and converts the priority-rating to enter into the multi-buffer queue for queuing.

8) Regarding the broadcast scheduling problem because of the broadcast data invalidation, ARM Agent will return to the server ARM registration center, find this ARM Agent’s registration number to match, and produces a copy of the broadcast data DataA, after producing this copy, will produce the ARM Agent of this copy according to the situation of the copy, then enter the next new broadcast process.

This invalidation control strategy by adjusting the level of priority and the copied broadcasting data which is produced by ARM Agent to ensure the data can be implemented in the effective time and in the executive terminal. Therefore, is particularly suited to the mobile computing system which requests strictly to the data execution and the execution time.

V. PERFORMANCE ANALYSIS AND COMPARISON

In order to confirm the effectiveness of the multi-buffer queue and the invalidation control strategy based on IBM ARM proposed by this paper, construct and develop the proposed scheduling data broadcasting scheduling system on some correspondence company’s 3G platform. The entire platform including mobile computation server 1, PC machine 16, ordinary PDA 10, computable 3G mobile phones 15. Carries on the validity comparison in this platform with the literature [5] [6]’s model and the control strategy. And to confirm the mobile termination scale to the model and the invalidation strategy’s influence, proposes three plans such as in Table 1 in the construction platform. And launches the performance analysis experiment under these three plans.

Figure 4 shows the Invalidation rate comparison the different project by the model and strategy presented in this paper, the X axis for the system model simulation time (unit for minute), and the Y axis for an invalidation rate. As can be seen from the figure, along with the number of mobile computing terminals increases, there is increasing invalidation rate trend, this mainly concerns with the mobile computer server’s ability to accommodate.

Figure 5 shows the broadcasting data’s invalidation rate comparison between the model strategy proposed by this paper and the re-access method proposed by the
literature [5], the X axis for the system model simulation time (unit for minute), the Y axis for a invalidation rate. As can be seen from the figure, the invalidation rate of the model and the strategy proposed by this paper is quite stable along with the simulation time, but that of the model and the strategy proposed by the literature [5] fluctuates wildly, and the model and the strategy proposed by this paper compared to the method proposed by literature 5 has the remarkable effect in cut in the invalidation rate (probably to reduce about 25%).

Figure 6 shows the data broadcasting scheduling average response time comparison between model of this paper and user cache method proposed by literature 6, the X axis for the system model simulation time (unit for minute), the Y axis for the average response time (unit for second). As can be seen from the figure, the average response time of the model and strategy proposed by this paper has a certain improvement, which reduces approximately about 20% compared to the method proposed by literature 6.

VI. CONCLUSIONS AND FUTURE WORK

This paper embarks from the invalidation of equipment and broadcasting information because of wireless bandwidth, communication quality and too long scheduling queue that badly influence the system computing performance, designs a multi-buffer queue-scheduling model, each buffer queue has its own priority, when the pending data after the algorithmic analysis enters in the different queue according to the respective priority. By proving that can be drawn, the introduction of a buffer with a priority (a multi-buffer queue) in the buffer queue will effectively reduce the invalidation rate of the data broadcast schedule.

In addition to an invalidation control strategy of data broadcasting scheduling based on mobile Agent and IBM Application Response Measurement (ARM) is presented, this invalidation control strategy by adjusting the level of priority and the copied broadcasting data which is produced by ARM Agent to ensure the data can be implemented in the effective time and in the executive terminal. This achievement will have the promotion and application value in the military science (migration military command system), the third generation mobile communication (information jamming) and so on domain.

From the experiment simulation result can be seen that the model and strategy proposed by this paper in the control of the invalidation (the invalidation rate and average work response time) has a certain improvement, but the real application of this model and strategy in practice, but also needs that the invalidation data's buffer ability as well as ARM Agent’ executive efficiency can be improved. In present's work, we will study the data broadcasting buffer technology as well as the "second (or more)" scheduling compensation technology after data failure, and identify and maintain the mobile distribution buffer region, let the buffer region assign and maintain according to the role of the mobile termination in the mobile computation environment, thus provides system's buffer ability, reduces the performance expenses.

REFERENCES


**Research Background**

With the number increase of mobile computing equipment, the invalidation of equipment and broadcasting information because of wireless bandwidth, communication quality and too long scheduling queue that badly influence the system computing performance. This paper designs a multi-buffer queue-scheduling model, which classifies the priority of broadcasting data. A invalidation control strategy of data broadcasting scheduling based on mobile Agent and IBM Application Response Measurement is presented to guarantee data executing on the terminal in the right time. This model and strategy can apply in military affairs science (mobile military commanding system) and the third generation mobile communication (information jamming). Our works are supported by the Science Foundation Fund of Hubei Province (2006ABA218) and Chongqing City Mobile Telecommunication Technology Key Lab Open Fund of Chongqing University of Post and Telecommunication.

**Hu Wen-bin**, born in 1977. He has been teacher of Wuhan University since 2006. His main research interests are intelligent simulating and optimization, multi-agent system and swarm intelligent algorithm.
A Novel Iterative Multilateral Localization Algorithm for Wireless Sensor Networks

Zhang Shaoping¹, ²
¹School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
²School of Computer and Information Engineering, Jiangxi Agriculture University, Nanchang, China
E-mail: spzhang@mail.jxau.edu.cn

Li Guohui¹, Wei Wei¹ and Yang Bing¹
¹School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
E-mail: guohuili@mail.hust.edu.cn, sybo0111@163.com, yangbing@smail.hust.edu.cn

Abstract—In many applications of wireless sensor networks, location is very important information. It can be used to identify the location at which sensor readings originate, in routing and data storage protocols based on geographical areas and so on. Location information can come from manual setting or GPS device. However, manual setting requires huge cost of human time, and GPS setting requires expensive device cost. Both approaches are not applicable to localization task of large scale wireless sensor networks. In this paper, we propose an accurate and efficient localization algorithm, called iterative multilateral localization algorithm based on time rounds. This Algorithm uses time round mechanism and anchor nodes triangle placement scheme to reduce error accumulation caused by iteratively localization. And it also reduces location errors and prevents abnormal phenomena caused by trilateral localization through limiting the minimum number of neighboring beacon nodes used in different localizing time rounds. Experimental results reveal that this algorithm has high localization accuracy, even if in large range errors, it can achieve good result.

Index Terms—Wireless sensor networks, Location estimation, Time rounds, Iterative Multilateral Localization

I. INTRODUCTION

Wireless sensor network (WSN) consists of a large collection of sensor nodes that are highly constrained in terms of their computing power, communication capabilities, and battery power. Its applications cover a wide range from natural monitoring to ambient awareness, from military to surveillance. Basically, each sensor node will monitor its local environment and they collaborate as a whole to provide information about the sensor field.

In WSN, location is used to identify the location at which sensor readings originate [1, 2], in communication route protocols based on geographical areas [3, 4], in data storage protocols based on geographical area partition [5, 6], and other services based on location. So location is one of the important issues in WSN. Location information can come from manual setting or GPS (Global Positioning System) device. However, manual location setting requires huge cost of human time, and GPS location setting requires expensive device cost. Both approaches are not applicable to localization task of large scale wireless sensor networks. In a network of thousands of nodes, it is unlikely that the designer determine the location of each node. In an extreme case, nodes may be dropped from the air and scattered about an unknown area. In order to localize per-node, a WSN usually consists of two category nodes: one is anchor nodes, which can get their location information through GPS or manual location, the other is unknown nodes, whose coordinates are unknown. Unknown nodes get their location information through anchor nodes and communication between nodes. In the following, we call nodes whose coordinates are known “beacon nodes” (including anchor nodes and unknown nodes which have been localized).

In this paper, we propose energy efficient, high accuracy distributed localization algorithm, called iterative multilateral localization algorithm based on time rounds (IMLBTR). This algorithm uses time round as localizing time unit, localizes round after round, and limits the minimum number of neighboring beacon nodes in different localizing time rounds. When the number of neighboring beacons of an unknown node equal to or more than the limited value, we apply all its neighboring beacon nodes to localize the unknown node. Upon an unknown node has been localized, it becomes a beacon node and sends its own location information to its neighboring nodes which will assist them to estimate their locations. In this iterative method, when using beacon nodes with location errors to localize other unknown nodes, it will produce greater error accumulations. Our proposed algorithm reduces localization errors from two aspects: (1) applying time rounds and anchor node triangle placement schemes to reduce error accumulations caused by iterative localization, (2) applying multilateral instead of trilateral...
localization to reduce localization errors and prevent abnormal phenomena caused by trilateral localizing.

The remainder of the paper is organized as follows: Section 2 discusses related work in localization for WSN. Section 3 gives detailed descriptions of our proposed IMLBTR. Section 4 describes our simulation and the performance analysis of the algorithm. Finally, we conclude in Section 5.

II. RELATED WORK

So far, many localization algorithms for sensor networks have been proposed to provide per-node location information. These algorithms take into account different factors to localization issues such as the network topology, device capabilities, localization accuracy and energy requirements. The localization schemes can be broadly classified into two categories: range-free schemes and range-based schemes. Range-free schemes do not require any technology or equipment to measure the distance or bearing between nodes, they just apply the communication among nodes to localize unknown nodes. The representative schemes are centroid [7], APIT [8] and DV-Hop [9]. The advantage of range-free schemes lies in their simplicity, as nodes do not need any additional device to measure range information. But they provide only coarse locations. In the following, we describe range estimation techniques and range-based localization schemes.

A. Range Estimation Techniques

Common techniques for distance or angle estimation include Time of Arrival (ToA), Time Difference of Arrival (TDoA), Angle of Arrival (AoA), and Received Signal Strength indicator (RSSI) [10].

The ToA technique measures the distance between nodes according to the signal traveling time. This technique requires precise time synchronization and high-speed sampling of the received signal. GPS [11] is a typical ToA-based localization system. The TDoA technique sends two different speed signals at the same time, and then uses their arrival time difference to calculate the distance between nodes. It requires low speed (for example, ultrasound) signal propagate device. AHLos (Ad-hoc localization system) [12] is a TDoA-based localization algorithm for WSN. The AoA technique measures the angle of arriving signal from anchor nodes, so it requires an antenna array at anchor nodes. APS (Ad-hoc Positioning System) [13] is an AoA-based localization algorithm for WSN. Although these techniques can accurately measure the distance or angle between nodes, as they require additional devices, they are not applicable to most of sensor networks.

The RSSI is based on the fact that the received signal power attenuates with distance. Thus, the distance between two nodes can be estimated according to the RSSI. But Radio frequency (RF) based range techniques are inherently dependent on the RF channel whose multipath fading and shadowing effects have a fundamental bearing on the accuracy of distance estimate. However, for WSN, RSSI-based range technique does not require any additional device, and the physical/medium access control (PHY/MAC) layer protocol of IEEE802.15.4 standard [14] defines a function of RSSI measurement in its protocol. In other words, if we construct a wireless sensor with the IEEE802.15.4 standard, a node can naturally measure the RSSIs from its neighboring nodes through communications. As a result, many researchers study the RSSI-based localization techniques, and propose a lot of the RSSI-based localization algorithm [15–19]. Literature [20] introduces a path loss model as follows:

\[
L(d) = \begin{cases} 
L_{ps}(d) & d \leq d_{bp} \\
L_{ps}(d) - 10 \cdot \alpha_2 \cdot \log_{10}(d/d_{bp}) + x & d > d_{bp}
\end{cases}
\]  

where 

\[
L_{ps}(d) = L_b + 10 \cdot \alpha_1 \cdot \log_{10}(d)
\]

Where \(L\), \(d\), \(d_{bp}\), \(\alpha_1\), \(\alpha_2\) represent path loss(dB), distance(m), breakpoint distance(m), power-distance gradient; before and after the breakpoint; respectively. \(x\) is the shadow fading component with a zero mean Gaussian probability distribution.

B. Range-based Localization Schemes

In range-based schemes, precise distance or angle measurements are made to estimate the location of nodes in the network. In the following, we describe some related range-based schemes.

The AHLos[12] scheme applies atomic multilateration, iterative multilateration and collaborative multilateration to localize unknown nodes. Unknown nodes which they have enough neighboring anchors estimate their locations through Atomic Multilateration. Once an unknown node estimates its location, it becomes a beacon and broadcasts its location to other neighboring nodes, enabling them to estimate their locations. This process repeats until all the unknown nodes that satisfy the requirements for multilateration obtain an estimate of their position. This process is defined as iterative multilateration which uses atomic multilateration as its main primitive. An unknown node may never have three neighboring beacons therefore it will not be able to estimate its position. When this occurs, a node may attempt to estimate its location by considering use of locations over multiple hops in a process referred to as collaborative multilateration.

The n-hop multilateration primitive [21] is also referred to as collaborative multilateration. It consists of a set of mechanisms that enables nodes found several hops away from beacon nodes to collaborate with each other and estimate their locations with high accuracy. Location estimates are obtained by setting up a global non-linear optimization problem and solving it using iterative least squares. This scheme addresses two issues which exist in the AHLos: (1) iterative multilateration is sensitive to beacon densities and can easily get stuck in places where beacon densities are sparse, (2) error propagation becomes an issue in large networks.

MDS-MAP [22] scheme applies multidimensional scaling (MDS) techniques, which are a set of data analysis techniques that display the structure of distance-like data as a geometrical picture, to estimate unknown
III. THE ITERATIVE MULTILATERAL LOCALIZATION ALGORITHM BASED ON TIME ROUNDS

This section gives detailed descriptions of our proposed IMLBTR. The network assumptions for this algorithm are as follows:

(1) Sensor nodes are static once they are placed. Each node supports a range estimation technique, RSSI for example, to estimate the distances to its neighboring nodes. We model range error as an independent Gaussian random distribution with zero mean and variance $E_r$.

(2) We can map the network region for the placement of anchor nodes.

A. Maximum Likelihood Estimation

The IMLBTR uses Maximum Likelihood Estimation (MLE) to calculate unknown nodes’ coordinates. The principle of MLE is as follows:

If an unknown node has $n$ ($n \geq 3$) neighboring beacons, it uses these beacons’ coordinates to estimate its coordinates. Let $(x, y)$ represents the unknown node’s coordinates; $(x_i, y_i) (1 \leq i \leq n)$ represents coordinates of its $i$’th neighboring beacon node; $d_i (1 \leq i \leq n)$ represents the distance between the unknown node and $i$’th neighboring beacon node. According to Euclidean distance formula, nonlinear equations are formulated as follows:

$$\begin{align*}
(x-x_1)^2 + (y-y_1)^2 &= d_1^2 \\
(x-x_2)^2 + (y-y_2)^2 &= d_2^2 \\
&\vdots \\
(x-x_n)^2 + (y-y_n)^2 &= d_n^2
\end{align*}$$

Through an equation minus the next equation, they are converted into overdetermined linear equations as follows:

$$\begin{align*}
2(x-x_1)(x-x_2) + 2(y-y_1)(y-y_2) &= d_1^2 - d_2^2 + x_2^2 - x_1^2 + y_2^2 - y_1^2 \\
2(x-x_1)(x-x_3) + 2(y-y_1)(y-y_3) &= d_1^2 - d_3^2 + x_3^2 - x_1^2 + y_3^2 - y_1^2 \\
&\vdots \\
2(x-x_1)(x-x_n) + 2(y-y_1)(y-y_n) &= d_1^2 - d_n^2 + x_n^2 - x_1^2 + y_n^2 - y_1^2
\end{align*}$$

This system of equations has the form $AX = b$ and can be solved using the matrix solution for least square method given by $X = (A^TA)^{-1}A^Tb$, where

$$A = \begin{pmatrix}
2(x_2-x_1) & 2(y_2-y_1) \\
2(x_3-x_1) & 2(y_3-y_1) \\
&\vdots \\
2(x_n-x_1) & 2(y_n-y_1)
\end{pmatrix}, \quad X = \begin{pmatrix}
x \\
y
\end{pmatrix}$$

and

$$b = \begin{pmatrix}
d_1^2 - d_2^2 + x_2^2 - x_1^2 + y_2^2 - y_1^2 \\
d_1^2 - d_3^2 + x_3^2 - x_1^2 + y_3^2 - y_1^2 \\
&\vdots \\
d_1^2 - d_n^2 + x_n^2 - x_1^2 + y_n^2 - y_1^2
\end{pmatrix}$$

B. The Triangular Placement Scheme of Anchor Nodes

Anchor nodes are expensive resources in WSN. If they are deployed randomly in the network region, their utilization rate is very low. Therefore, it is necessary to place anchor nodes elaborately. The ideas of the triangular placement scheme of anchors come from two aspects. (1) Due to a small number of anchors, if they are placed randomly in network region, the probability that an unknown node has three or more neighboring anchor nodes is very low, so few unknown nodes can be localized directly by using anchors. As a result, we group anchors. Each group consists of three anchors, forms an equilateral triangle to place into network region. The size of triangular area should assure that some unknown nodes can communicate with all anchors in the group so that they can use these anchors to localize directly. (2) In order to reduce the error accumulation, it should reduce the number of iterations. We apply a triangle placement scheme of anchors to reduce the number of iterations. Considering a triangle region in Figure 1, it is divided into four sub-regions A, B, C, and D. In triangle placement scheme (Fig.1 (a)), three group anchors are placed in region A, B, C, unknown nodes in region D can be localized by applying beacon nodes which are localized in region A, B, C. Whereas in non-triangle placement scheme (Fig.1 (b)), three group anchors are placed in region A, C, D, unknown nodes in region B can be localized only by applying beacon nodes which are localized in region D. Obviously, the average number of iterations that unknown nodes require in region D in triangle placement is less than that in region B in non-triangle placement. Therefore, the average localization error in triangle placement is lower than in non-triangle placement.
communication radius. Figure 2 shows the triangle placement of each group. When $L > \sqrt{3} R$ (Fig. 2(a)), which the communication area of three anchors have no common area, no unknown nodes can be localized directly by these anchors. When $R < L < \sqrt{3} R$ (Fig. 2(c)), all unknown nodes in the grey area can be localized directly according to anchors, there must be some nodes in the grey area. Therefore, $L$ is determined by network connectivity (the average number of neighboring nodes of each node). Let $con$ represents network connectivity. According to network connectivity and the size of grey area, we give a set of reference values of $L$ as follows:

$$L = \begin{cases} 
1.0R & con < 12 \\
1.1R & 12 \leq con \leq 18 \\
1.2R & con \geq 18 
\end{cases} \quad (4)$$

Secondly, for a given network area, we construct its minimum enclosing rectangle or hexagon, divide it into equilateral triangle area, then place a group of anchors into every other triangle area in row major order, as show in Figure 3. The triangle of grouped anchors has the same direction and centroid with triangle area. Let $L'$ represents triangle area side. Figure 3 shows an example of anchors placement. Where $R=30$, $L=1.1R$, $L'=2L$. We can calculate each anchor’s coordinate by using $L$ and $L'$.

In practical application, if the locations that anchors are placed do not have large place errors, it has no effect to our proposed algorithm. Experiments in section 4.1 illustrate this.

C. Overview of IMLBTR

The triangular placement scheme can reduce error accumulations by reducing the number of iterations. We can further reduce the localization errors by the time round scheme. The ideas of the time rounds scheme come from two aspects: (1) Trilateral localization not only produces large location errors but also easily lend to an abnormal phenomena. Due to localization error, when three nodes with location errors, which the real locations of these nodes close to a line, are applied to localize another unknown node, this maybe produces abnormal phenomena, as show in Figure 3. A, B, C represent three nodes’ real locations, D represents an unknown node’s real location, $A'$, $B'$, $C'$ represent locations of A, B, C after localization, $D'$ is location of D after localization using $A'$, $B'$, $C'$. In order to reduce localization errors and prevent the abnormal phenomena, we should use neighboring beacon nodes as many as possible to localize unknown nodes. (2) There are error accumulations in Iteration localization, the more the iterations, the larger the error accumulations. In order to reduce error accumulations, we should use beacon nodes whose iterations are as fewer as possible to localize unknown nodes.

After all nodes have been deployed, they first execute some initial operations, such as synchronizing system time, applying a range estimation technique to estimate distances between neighboring nodes, setting system parameters, and so on. Next, all nodes start the localization algorithm at the same time. Our proposed algorithm introduces time round scheme, localizes round after round, and it limits the minimum number of neighboring beacon nodes that localization requires in different time round. Each round is divided into two phases: the first phase is localizing phase, all unknown nodes, which the number of their neighboring beacon nodes equals to or more than the requirement in this round, are localized; the second phase is location data transmission phase, all nodes which are localized in this round send their own location data to their neighboring nodes. In the first round, localizing based on anchors, all unknown nodes, whose neighbor nodes have three or more anchors, are localized. Upon an unknown has been localized, it becomes a beacon node and sends its own location data to its neighboring nodes. In subsequent each round, localizing based on beacon nodes, all eligible unknown nodes are localized. And as the same, upon an unknown has been localized, it becomes a beacon node and sends its own location data to its neighboring nodes.

Through using time round scheme, IMLBTR assures that unknown nodes are localized by using beacon nodes whose iterations are as fewer as possible, therefore, it reduces error accumulations. Additionally, it limits the minimum number of neighboring beacon nodes that localization requires in different time round so as to use multilateral instead of trilateral localization to reduce localization errors and prevent abnormal phenomena.
D. Algorithm Description

IMLBTR must set two parameters. (1) The size of time rounds. In a time round period, each node must complete tasks as follows: calculating its own location and send it to its neighbors, receiving all location data packets sent by its neighboring nodes whose locations are calculated in this round. Calculating time, comparing to communicating time, can be negligible. Let \( T \) represents the size of a time round. So \( T \) is determined by network connectivity and the time of sending a location data packet. Assuming network connectivity is \( con \), the time of sending to receiving a location data packet is \( t \), we use \( T = 2 \times con \times t \) to represent the size of time round. (2) The minimum number of beacon nodes that localization requires in different time round (represented by \( minBs \)). We set an upper limit (represented by \( UpperLimit \)) to \( minBs \). In the first round, localizing only based on anchors, so \( minBs \) assign to 3. With the increasing in the number of iterative time rounds (represented by \( Rounds \)), the number of beacon nodes is increasing too. So when \( Rounds \) increase 1, \( minBs \) is also an increase, but at most \( UpperLimit \), i.e. when \( minBs \) is greater than \( UpperLimit \), it is assigned to \( UpperLimit \). Additionally, some nodes, which the number of their neighboring beacon nodes can not be equal to or greater than \( minBs \), can not be localized for ever. So we set a maximum round for multilateral localization (represented by \( maxMR \)). When \( Rounds \) is equal to or more than \( maxMR \), \( minBs \) is assigned to 3. Additionally, a few of nodes, which the number of their neighboring beacon nodes never equals to 3, can not be localized for ever. Therefore, we set a maximum round for localization (represented by \( maxRound \)). When \( Rounds \) equals to \( maxRound \), the algorithm terminates. According to these rules, the relationship between \( minBs \) and \( Rounds \) can be represented as the following function:

\[
minBs = \begin{cases} \frac{Rounds + 2}{UpperLimit} & 1 \leq Rounds \leq UpperLimit - 2 \\ \frac{UpperLimit - 2}{maxMR} \times Rounds < maxMR \leq maxRound \\ \minBs & Rounds > maxRound \end{cases}
\]

The upper limit (\( UpperLimit \)) of the minimum number of beacon nodes has larger effect on localization errors. When \( UpperLimit \) increases, as localization requires more beacon nodes to participate in, localization errors will be lower. However, the algorithm needs more time rounds to execute, and so it will lead to greater error accumulations. So we should make a tradeoff between localization errors and error accumulations. Experiments in section 4.2 show the effect of \( UpperLimit \) on localization errors, and reveals that the best value of \( UpperLimit \) is 5. Additionally, we do large numbers of experiments. Experimental results reveal that \( maxMR \) never exceeds 10 and \( maxRound \) no more than 20. Therefore, in this paper, \( maxMR \) is set to 10 and \( maxRound \) is set to 20.

IMLBTR is implemented through event-driven method. The algorithm uses two events: periodic timer trigger event and receiving location data packet event. We name them \( localizationTimer \) and \( receivingLocation \) respectively. After Initialization, all nodes start localization algorithm at same time. Upon a node receiving a location packet (firing \( receivingLocation \)), it updates neighboring node list with the location data. Upon a node’s \( localizationTimer \) fired, the node executes following operations: if it is an anchor node, it directly sends its location data to neighboring nodes, stops \( localizationTimer \), terminates the algorithm; otherwise, if it meets the conditions that this time round requires, then it calculates its own location and send location data to its neighboring nodes, then stops \( localizationTimer \), terminates the algorithm, else waits the next time fired. Algorithm 1 provides a formula description for localizing procedure of a node.

Algorithm 1: Iterative Multilateral Localization Algorithm Based on Time Rounds
Input: Neighboring list with neighboring nodes’ id and distance
Output: the location of this node
1. initializations;
2. Rounds=0;
3. start \( LocalizationTimer \) at \( T0 \), every \( T \) time fired;
4. UPON \( LocalizationTimer \) FIRED
5. BEGIN
6. \( Rounds = Rounds + 1 \);
7. IF this node is a anchor node THEN
8. send location data to its neighboring nodes;
9. stop \( LocalizationTimer \);
10. END IF
11. IF \( Rounds = maxRounds \) THEN
12. stop \( LocalizationTimer \);
13. return can not be localized information;
14. END IF
15. \( minBs = \) calculate \( minBs \) of this Rounds;
16. \( beacons = \) Count the number of beacon nodes from neighboring list;
17. IF \( beacons \geq minBs \) THEN
18. calculating the location of the node;
19. send location data to its neighboring nodes;
20. stop \( LocalizationTimer \);
21. ELSE
22. \( / / \) waiting next time round;
23. END IF
24. End
25. UPON receivingLocation FIERD
26. BEGIN
27. update neighboring list with received location;
28. END

IV. EXPERIMENTAL RESULTS
In this section, we verify our proposed algorithm and analyze the simulation results from different conditions or perspectives. In these simulations, all unknown nodes are placed randomly with a uniform distribution within a \( 200m \times 200m \) square area. The communication radius of each node is 30m. We model range errors as an independent Gaussian random distribution with zero mean and variance \( \sigma \). A range error \( d^* \), assuming the true distance is \( d \), a random measured error drawing from a normal distribution \( d^* - N(0, \sigma) \) , and so the measured distance is \( d + d^* - N(0, \sigma) \). Localization errors are normalized to \( R \), i.e. assuming \( d^* \) is the distance between estimated coordinates and true coordinates, the localization error is \( d^*/R \).
A. Localization Errors When Varying Anchors Placement

In this experiment, we study the effect of anchors placement on the localization errors of our proposed algorithm. The anchor nodes are placed (a) randomly with a uniform distribution within network area, called it randomly placement, (b) on a square grid with some placement errors, called it grid placement, (c) on an equilateral triangle based on our proposed triangular placement scheme with some placement errors, called it triangle placement, and (d) on an equilateral triangle based on our proposed triangular placement scheme with no placement errors, called it ideal triangle placement. We model placement errors for the grid and triangle placement as Gaussian noises. With a placement error $E_p$, a random value drawing from a normal distribution $R \times E_p \times N(0,1)$ is added to the node’s ideal position.

Assuming the ideal location is $(x_i, y_i)$, the placement is 20% ($E_p = 0.2$) the simulation location $(x, y)$ can be calculated as follows:

$$x = x_i + E_p \times R \times |N(0,1)| \times \cos \alpha$$
$$y = y_i + E_p \times R \times |N(0,1)| \times \sin \alpha$$

(6)

Where $\alpha$ is a random degree of $[0,2\pi)$ with uniform distribution.

![Figure 5. The Effect of Placement of anchors](image)

Figure 5 shows the experiment results with the range error $E_p = 0.1$ and 200 unknown nodes. From the experiment results, we conclude that: the localization errors caused by anchors random placement are far larger than that caused by grid or triangle placement. When anchors are placed randomly, due to irregular placement, a few of unknown nodes can be localized directly by using anchors, and so producing larger error accumulation; when anchors are placed in grid or triangle with 20% placement error, localization accuracy is very high and close to the accuracy of ideal triangle placement; when the number of anchors is greater than 24 (about 10% of The total number of nodes), with the increasing in the number of anchors, localization accuracy despite the increasing but in small change.

B. Localization Errors when Varying UpperLimit

IMLBTR uses time round scheme to reduce error accumulations, limits the minimum number of beacons that localization requires in different time rounds, and sets an upper limit (UpperLimit) for the minimum number. In this simulation, we study the effect of the upper limit (UpperLimit) on the localization errors. 21 anchors are placed on equilateral triangles with 20% placement error, 200 unknown nodes are placed randomly in network region.

Figure 6 shows the experiment results when varying UpperLimit. When UpperLimit=3, in fact, the minimum number of beacons that localization requires in all rounds is 3, localization errors are very large, especially in large range errors. This is because trilateral localization has large errors and is prone to produce the abnormal phenomena. Consequently, unknown nodes which are localized by using three beacons have large location errors. Once these nodes become beacons and are used to localize other unknown nodes, it will produce greater error accumulations. When UpperLimit=5, it has best localization accuracy. When UpperLimit>5, the localization accuracy is slightly lower. From the experiment results, we conclude: in our proposed IMLBTR, in the first round, minBs=3; in the second round, minBs=4; and in the subsequent rounds, minBs=5, until the number of time rounds equals to or more than maxTR, minBs=3.

![Figure 6. The Effect of UpperLimit on Localization Errors](image)

C. Localization Errors When Varying Network Connectivity

In this simulation, we study the effect of network connectivity on localization errors. We place 24 anchors on equilateral triangles in network area with 20% placement error; and randomly place 120, 150, 180, 200, 220, 250 unknown nodes in network area, corresponding to the connectivity of 9.2, 11, 13, 14.1, 15.5 and 17.4 respectively.

![Figure 7. The Effect of Connectivity on Localization Errors](image)

Figure 7 shows the experiment results, when connectivity is lesser than 10, localization errors are slightly larger. This is because some unknown nodes,
which the number of their neighboring beacon nodes is lesser than minBs in every time rounds, are not localized until the number of time rounds equals to or more than maxTR; when connectivity is greater than 10, localization errors only have slightly change. From the experiment results we conclude that the network connectivity has slight effect on localization errors.

D. The Effect of Network Area Shape on Localization Errors

In this simulation, we study the effect of network area shape on localization errors. Figure 8 shows a C-shape network with 150 unknown nodes and 18 anchor nodes. 150 unknown nodes are placed randomly in network area, 18 anchors are placed on equilateral triangles with 20% placement error.

Figure 9 shows the experiment results when varying the range errors. From the experiment results we conclude that the network area shape has no effect on localization errors. This is because each node only uses its neighboring nodes’ location to calculate its own location, do not require nodes which are beyond its communication range.

E. IMLBTR Performance Analysis

First, we analyze the energy consumption of our proposed algorithm. In IMLBTR, all nodes only exchange location information with their neighboring nodes. Each node sends itself location data to neighboring nodes only once, and receives all neighboring nodes’ location data. Comparing with some distributed localization algorithm, the n-hop multilateration primitive requires multi-hop nodes’ locations, dwMDS needs exchange neighboring node repeatedly and broadcast their anchors’ locations to entire network. Therefore, the energy that they consumed for localization is far more than that IMLBTR consumed. And so our proposed algorithm is energy efficient localization algorithm.

Secondly, we analyze the number of anchors that IMLBTR requires. In IMLBTR, as anchor nodes are placed with triangle placement scheme, the number of anchors is related to the size of network area and network connectivity. When connectivity is about 12, the ratio of anchors to total nodes is lesser than 10%, and the greater the connectivity, the lesser the ratio of anchor nodes to total nodes. Accordingly, IMLBTR is applicable to dense wireless sensor networks.

Finally, we analyze the localization accuracy of our proposed algorithm. When network connectivity is lesser than 10, the localization accuracy is slightly low. And a few of nodes, which the number of neighboring beacons is lesser than 3, can not be localized. But when network connectivity is greater than 10, the localization accuracy is very high. When the range error is small(Er=0.05), the localization accuracy is about 92%; and when the range error is large(Er=0.3), the localization accuracy is about 60%. Therefore, IMLBTR is applicable to RSSI-based range technique.

V. CONCLUSION

In this paper, we propose an iterative multilateral localization algorithm based on time rounds for wireless sensor network. It uses anchor triangle placement and time round schemes to reduce error accumulation caused by iterative localization, and limits the minimum number of neighboring beacon nodes that localization requires in each time rounds to reduce localization errors and to prevent abnormal phenomena caused trilateral localization. Consequently, it is a high accuracy localization algorithm, even if in large range errors, it can achieve good localization result. This algorithm is an energy efficient range-based localization algorithm and applicable to the RSSI-based range technique.

ACKNOWLEDGEMENT

This work was supported by the National High-Tech Research and Development Plan of China under Grant No. 2007AA01Z309, the National Natural Science Foundation of China under Grant No. 60373000, and the Natural Science Foundation of Hubei Province of China.

REFERENCES

Analysis of Queuing Behaviors with Self-similar Traffic in Wireless Channels

Liu Chang
Wireless Information Network Lab, University of Science and Technology of China, Hefei, China
Email: changool@mail.ustc.edu.cn

Qin Xiaowei, Zhang Sihai and Zhou Wuyang
Wireless Information Network Lab, University of Science and Technology of China, Hefei, China
Email: {qinxw, shzhang, wyzhou}@ustc.edu.cn

Abstract—Many measurement studies showed that network traffic usually exhibits self-similar nature, meanwhile, wireless fading channels lead to the time-variant service rates of network nodes. Compared with classical queuing systems with Markov sources or constant service rates, the queuing system with self-similar input in a wireless fading channel is more complicated, and it is very difficult to obtain closed-form formulae for the queuing performances. In this paper, we study queuing behaviors with self-similar traffic input in wireless channels by establishing a queuing system model, and present a convenient and efficient algorithm to estimate the queue length distributions. Simulation results validate the accuracy of the proposed estimation algorithm. Finally, we analyze effective service rates of wireless channels based on the estimation algorithm, and draw some meaningful conclusions.

Index Terms—self-similar traffic, wireless channel, queuing system, queue length distribution, estimation algorithm, effective service rate

I. INTRODUCTION

With rapid development of wireless technology, demand for wireless data services is continuously increasing. How to provide quality of service (QoS) guarantees plays an important role in the design of wireless networks. Since queuing behavior is closely related to the QoS guarantees, it has attracted much attention as a research topic [1-7].

Since the characteristic of self-similarity of network traffic was found by Leland et al. in 1993 [8], a large number of study results indicate that network traffic in a variety of modern communication networks exhibits self-similar nature [9], such as in computer networks [10], in Ad hoc networks [11] and in wireless LANs [12]. As self-similar traffic exhibits scale-invariant burstiness, the analytical models developed under short-range-dependent traffic are no longer valid in the presence of self-similar traffic, for example, classical Markov-based queuing analysis is not suitable any more [13].

There have been many studies on queuing performance with self-similar traffic. Norros proposed the Fractional Brownian Motion (FBM) model to characterize self-similar traffic [14], and obtained the queue length distribution. The formulae of average queue length, queue length variance, average delay and jitter were derived in [3]. In [1], the loss probability with self-similar traffic in a finite partitioned buffer was analyzed. All the literatures above are based on the assumption that the service rates of the queuing systems are constant. However, in wireless environments, wireless features such as fading and Doppler spread lead to the variable channel capacity, that is to say, the service rate of the queue is a variable. Although [4] considered the variable service rate, which was characterized by a FBM process, the practical features and parameters of wireless channels were still not discussed.

In [6], [7], [15] and [16], the authors studied the queuing performances in practical fading channels based on the theories of effective bandwidth and effective capacity, but all of them didn’t investigate the queuing behavior with self-similar traffic. Ref. [6] assumed that the source rate is constant, and [7] adopted an ON-OFF Markov source, which was also used in [15] and [16] with a first-order autoregressive (AR) source.

In this paper, we consider a wireless network model as shown in Fig. 1. The confluent traffic flow at node A exhibits self-similarity, and is forwarded to node B over a wireless channel. Our interest is focused on the queuing behavior of node A. We establish a queuing system model, based on which further analysis of queuing performances with self-similar traffic input in wireless channels is conducted, and then a convenient and efficient algorithm is proposed to estimate the queue length distributions. By taking samples of the queue, our estimation algorithm works well with no need for the source and the channel parameters. Then, we analyze effective service rates of different wireless channels based on the proposed estimation algorithm.

The rest of this paper is organized as follows: The queuing system model is introduced in section II. In section III, we study the queuing behaviors based on the model and present an estimation algorithm for the queue length distributions. Section IV validates the accuracy of the proposed estimation algorithm by simulation. We analyze effective service rates of wireless channels in section V. Finally, section VI concludes the paper and points out future research directions.
II. QUEUING SYSTEM MODEL

To investigate the queuing behavior of node A in Fig. 1, we establish a queuing system model including three sub-modules: the traffic model, the queuing model and the wireless channel model, as shown in Fig. 2. We will specify these three sub-modules respectively.

A. Self-similar Traffic Model

Many analytical models have been developed to characterize self-similar traffic flows. Among these models, FBM model is identified as an efficient way \cite{1}, \cite{2}, \cite{3}, \cite{4}, \cite{12}, and hence adopted in this paper. A FBM traffic flow \cite{12} can be expressed as:

\[ A(t) = mt + \sqrt{am} Z_m(t), \quad t > 0, \]

where \( A(t) \) denotes the amount of traffic arriving in time interval \([0, t]\), \( m \) is the mean traffic arrive rate, and \( a \) is the variance coefficient of \( A(t) \). In this expression, \( a = \sigma^2 / m \), where \( \sigma^2 \) is the variance of traffic in a time unit. \( Z_m(t) \) is a standard FBM process with: (i) \( Z_m(0) = 0 \), (ii) \( \text{Var}(Z_m(t)) = t^{2H} \), and (iii) \( \text{Cov}(Z_m(t), Z_m(s)) = (t^{2H} + |t-s|^{2H} - |t-s|^{2H}) / 2 \), where \( H \in [0.5,1) \) is the Hurst parameter. The closer \( H \) is to one, the greater the degree of self-similarity is.

B. Wireless Channel Model

In order to analyze queuing behaviors with self-similar traffic in wireless channels, we adopt three fading channel models, which are two-state Markov channel model, Rayleigh fading channel model and Rician fading channel model, respectively. We will describe them as below:

1) Two-state Markov channel model

In wireless environments, wireless features such as fading and Doppler spread lead to the variable channel capacity. And the wireless channel can be modeled as a two-state Markov chain for simplicity, as shown in Fig. 3.

The channel capacity in state G (good) is larger than that in state B (bad). \( p \) is the transition probability from G to B and \( q \) is the transition probability from B to G.

In this paper, we assume that the channel capacity is equal to a fixed value \( C_G \) with channel state G, and is \( C_B \) with state B (\( C_G > C_B \)). The channel state changes at each sample interval.

2) Rayleigh fading channel model

Rayleigh fading model is a commonly used channel model to describe the fading feature of wireless channels. The instantaneous capacity of the Rayleigh fading channel at the \( n \)th sample interval can be expressed as:

\[ C_n = B \log_2 \left( 1 + \left| h_n \right|^2 \times \bar{\gamma} \right), \]

where \( B \) is the channel bandwidth, \( h_n \) is the channel gain and \( |h_n| \) is a Rayleigh random variable. Average signal-to-noise ratio (SNR) \( \bar{\gamma} = S / (N_0 B) \), \( S \) and \( N_0 \) are the average transmit power and the power spectral density of the noise, respectively. It is well known that the envelope of the sum of two quadrature Gaussian noise signals obeys a Rayleigh distribution, so that \( h_n \) can be generated by a first-order AR model \cite{5}:

\[ h_n = k h_{n-1} + v_n, \]

where the noise \( v_n \) is zero-mean complex Gaussian with variance of \( 1 - k^2 \) per dimension, and is statistically independent of \( v_{n-1} \). The coefficient \( k \) can be determined by

\[ \kappa = 0.5^{1/T_c}, \]

where \( T_s \) is the sampling interval, \( T_c \) is the coherence time, and can be computed by

\[ T_c = \frac{9}{16 \pi f_m}, \]

where \( f_m \) is the maximum Doppler shift.

3) Rician fading channel model

When there is a dominant stationary signal component present, the small-scale fading envelope distribution is Rician. The instantaneous capacity of the Rician fading channel at the \( n \)th sample interval can still be expressed as (2), but \( |h_n| \) is a Rician random variable here. \( |h_n| \) can be generated as follows \cite{17}:
Here $x_i$ and $y_i$ are samples of zero-mean stationary Gaussian random processes each with unit variance. The parameter $K$ is the Rician factor, which is defined as the ratio between the deterministic signal power $A^2$ and the variance of the multipath $2\sigma_m^2$. It is given by $K = A^2/2\sigma_m^2$.

C. Queuing Model

We consider a single-input single-output (SISO) queuing model as shown in Fig. 2, and assume that the buffer size is infinite and the transmitter has perfect knowledge of the channel gain $h_n$ at each sample interval. Therefore, rate-adaptive transmissions and strong channel coding can be used to achieve error-free transmission. Thus, the service rate of the queue is equal to the instantaneous channel capacity $C_m$.

For simplicity, we adopt a fluid model [3], [4], [6], where the size of a packet is assumed to be infinitesimal. Let $R_n$ and $Q_n$ denote the amount of traffic arrives and the queue length at the $n$th sample interval, respectively. The units of $R_n$ and $Q_n$ are bits in this paper. Thus, we have:

$$Q_{n+1} = (Q_n + R_n - C_nT_n)^+,$$

where $(x)^+ = \max(0,x)$.

III. QUEUE LENGTH DISTRIBUTION AND ESTIMATION ALGORITHM

The arrival and service processes of the queue presented in section II are both complicated, therefore, it is difficult to obtain closed-form formulae to describe the queuing behaviors such as the queue length distribution. In this section, based on background knowledge and simulation results, a convenient and efficient algorithm is proposed to estimate the queue length distribution.

A. Background Knowledge

Some conclusions of queue length distributions in relatively simple scenarios are presented as follows:

**Conclusion I**: If the input of a infinite buffer is a FBM process, and the service rate is constant, the queue length approximately follows a Weibull distribution [14], which is given by

$$P(Q > B) \approx \exp(-\alpha B^\varepsilon),$$

where $\varepsilon = 2 - 2H$, 

$$\alpha = \frac{(C - m)^{2H}}{2H^{2H} (1 - H)^{2H} \sigma^2},$$

$B$ is the queue length, and $C$ is the constant service rate.

**Conclusion II**: If the arrival and service processes of an infinite buffer are Markov processes, the queue length approximately follows an exponential distribution [6], [18], which is given by

$$P(Q > B) = \gamma \exp(-\theta B),$$

where $\gamma = P(Q > 0)$ is the probability that the buffer is nonempty, $\theta$ is a certain positive constant called the QoS exponent. $\gamma$ and $\theta$ are difficult to be calculated if the arrival and service processes are complicated, but can be estimated instead [6], [19]. Take $N$ samples of the queue, let $S_n$ denote whether a packet is in service ($S_n \in \{0,1\}$). $Q_n$ denotes the number of bits in the queue, then the estimated values of $\gamma$ and $\theta$ can be computed as

$$\hat{\gamma} = \frac{1}{N} \sum_{n=1}^{N} S_n,$$

$$\hat{\theta} = \frac{\hat{\gamma}}{\hat{q}},$$

where $\hat{q} = \frac{1}{N} \sum_{n=1}^{N} Q_n$.

B. Queuing Behavior with Self-similar Input in Rayleigh Fading Channel

In order to investigate the queueing behaviors with self-similar traffic input in wireless channels, we first simulate the system depicted in Fig. 2 to obtain some preliminary results.

1) We adopt the fast Fourier transform (FFT) method [20] to generate two independent traffic flows, which can be characterized by FBM models. The parameters are set to be $m_1 = m_2 = 50$, $\sigma_1^2 = \sigma_2^2 = 100$, $H_1 = 0.80$, and $H_2 = 0.85$.

2) We adopt Rayleigh fading channel as our channel model here. The parameters of the Rayleigh fading channel are set as follows: channel bandwidth $B_r = 10kHz$, average SNR $\bar{\gamma} = 10dB$, maximum Doppler shift $f_d = 30Hz$ and sampling interval $T_s = 2ms$.

By simulations, the curve of actual queue length distribution and two contrast curves are shown in Fig. 4(a) and (b) with $H = 0.80$ and 0.85, respectively. The two contrast curves are defined as bellow:

**Contrast curve 1**: Simulation curves by substituting (10) (11) into (9), which approximately follow an exponential distribution;

**Contrast curve 2**: Simulation curves by using the mean of the channel capacity $\bar{C} = \frac{1}{N} \sum_{n=1}^{N} C_n$ to replace $C$ in (8), and the curves approximately follow a Weibull distribution.
It can be observed in Fig. 4 that as \( B \) increases, the actual probability \( P(Q > B) \) decreases much more slowly than curve (ii), which decreases exponentially with \( B \). The results show that in Rayleigh fading channel, the queue length distribution with self-similar traffic is different from that with Markov traffic. Comparing curve (i) with curve (iii), it can also be observed that the actual probability is larger than that with a constant service rate, but at the same time, curve (i) and (iii) exhibit a similar shape, especially when \( H \) is larger.

Moreover, we can draw the following conclusions from Fig. 4(a) and (b):

1) the larger the Hurst value is, the more slowly the probability \( P(Q > B) \) decreases with \( B \);
2) the larger the Hurst value is, the more significantly it impacts the queue length distribution, and the less significantly the channel condition does.

C. Estimation Algorithm for the Queue Length Distribution

We have observed by simulation that curve (i) and curve (iii) exhibit a similar shape in Fig. 4, so that we use a Weibull distribution to approximate the queue length distribution with self-similar traffic input in Rayleigh fading channel as bellow:

\[
P(Q > B) \approx \exp\left(-\eta B^\phi\right). \tag{12}
\]

Due to the complexity of the arrival and service processes of the queue, we adopt an estimation algorithm to calculate \( \eta \) and \( \phi \) in (12) approximately. Considering that \( \phi \) is related with the Hurst parameter of the traffic flow, we set \( \phi = 2 - 2H \) according to (8). if \( H \) is unknown, it can be estimated by the periodogram-based method [8]. Thus, we have:

\[
\hat{\phi} = 2 - 2\hat{H}, \tag{13}
\]

where \( \hat{H} \) is the estimated value of \( H \). Take \( N \) samples of the queue, let \( Q_n \) denote the queue length at the \( n \)th sample interval, and then the average queue length can be estimated by:

\[
\hat{q} = \frac{1}{N} \sum_{n=1}^{N} Q_n . \tag{14}
\]

From (12), the theoretic mean value of the queue length can be expressed as:

\[
\overline{Q} = \Gamma\left(\frac{1}{\phi} + 1\right) \eta^{\frac{1}{\phi}}, \tag{15}
\]

where \( \Gamma(\cdot) \) is the Gamma function. Substituting \( \hat{\phi} \) and \( \hat{q} \) into (15), we obtain the estimated value of \( \eta \) as:

\[
\hat{\eta} = \hat{q} e^{2\hat{H} - 2} \left[ \Gamma\left(\frac{3 - 2\hat{H}}{2 - 2\hat{H}}\right)\right]^{\frac{1}{2 - 2\hat{H}}} . \tag{16}
\]

Finally, we can estimate the queue length distribution by substituting \( \hat{\phi} \) and \( \hat{\eta} \) into (12), as follows:

\[
\hat{P}(Q > B) \approx \exp\left(-\hat{\eta}B^{\hat{\phi}}\right). \tag{17}
\]

Our estimation algorithm for the queue length distribution is summarized below:

**Step 1:** Take \( N \) samples of the queue, meanwhile, record the amount of traffic arrives and the queue length at each sample interval;

**Step 2:** Estimate the Hurst parameter of the input traffic flow by the periodogram-based method, and calculate \( \hat{q} \) by (14);

**Step 3:** Calculate \( \hat{\phi} \) and \( \hat{\eta} \) by (13) and (16), respectively;

**Step 4:** Estimate the queue length distribution by (17).

It is worth noticing that our estimation algorithm works by taking samples of the queue only, and needn’t to know the source and the channel parameters, which are difficult to get in actual environments. Although the algorithm is obtained from Rayleigh fading channel, we show that it can estimate queue length distributions in other wireless channels in the following section.
IV. SIMULATION RESULTS

We simulate the queuing system depicted in Fig. 2, and validate the accuracy of the proposed estimation algorithm via comparing the actual queue length distributions with the estimated results. We first estimate the queue length distributions with different self-similar traffic flows in Rayleigh fading channel, and then show that our estimation algorithm also works well in two-state Markov channel and Rician fading channel.

A. Performance of the Estimation Algorithm in Rayleigh Fading Channel

In this simulation, we still adopt the FFT method to generate four independent flows, and set $m = 50$ and $\sigma^2 = 100$ for all of them, but $H = 0.66$, $0.71$, $0.80$, $0.85$, respectively.

The parameters of Rayleigh fading channel are chosen as presented in section III-B: channel bandwidth $B_c = 10kHz$, average SNR $\bar{\gamma} = 10dB$, maximum Doppler shift $f_d = 30Hz$ and sampling interval $T_s = 2ms$. The number of samples is $N = 1.3 \times 10^5$. Actual and estimated results are shown in Fig. 5, where the horizontal axis represents the queue length and the vertical axis denotes the corresponding probability.

From Fig. 5, we can see that for the traffic flows with different Hurst values, the estimated results all match the corresponding actual results closely. That is to say, for a queuing system with self-similar traffic in Rayleigh fading channel, the proposed estimation algorithm can predict the queue length distributions accurately.

B. Performance of the Estimation Algorithm in Two-state Markov Channel

We aim at investigating the accuracy of the proposed estimation algorithm in a two-state Markov channel. As described in section II-B, we set the channel capacity $C_0 = 40kbps$ and $C_2 = 15kbps$, the state transition probability $p = 0.5$, $q = 0.5$, and sampling interval $T_s = 2ms$. The parameters of traffic flows are the same as the ones in section IV-A. The queue length distributions with different self-similar flows are shown in Fig. 6.

We note that the estimated results still match the corresponding actual ones. Thus, our estimation algorithm is efficient in a two-state Markov channel with different self-similar flows.

C. Performance of the Estimation Algorithm in Rician Fading Channel

This scenario is intended to examine the accuracy of the developed estimation algorithm in Rician fading channel. The parameters of the simulation are: Rician factor $K = 3dB$, channel bandwidth $B_c = 10kHz$, average SNR $\bar{\gamma} = 9dB$, sampling interval $T_s = 2ms$. We still use the traffic flows generated in section IV-A. The results are shown in Fig. 7. It shows that the proposed estimation algorithm also hold for Rician fading channel.
In summary, the results in Fig. 5-7 have shown that queue lengths with self-similar traffic input in different wireless channels (two-state Markov channel, Rayleigh and Rician fading channels) follow Weibull distributions.
approximately, and our proposed algorithm can estimate the queue length distributions accurately. In the following section, we make further analysis of the queuing behaviors based on the estimation algorithm.

V. EFFECTIVE SERVICE RATES OF WIRELESS CHANNELS

From Fig. 4, we can see that the actual probability is larger than the probability by using the mean of the channel capacity $\bar{C} = \frac{1}{N} \sum_{n=1}^{N} C_n$ to replace $C$ in (8). It means that the utilization ratio of a stable channel (channel capacity is constant) is larger than that of a wireless fading channel. So it is meaningful to investigate the effective service rates of wireless channels.

From (12) to (17), we know that queue length distributions with self-similar traffic in wireless channels depend on two parameters: $\varphi$ and $\eta$. $\varphi$ is only related with the self-similar traffic, while $\eta$ is related with the traffic and the channel condition. For the same traffic input, different channel conditions are reflected by different values of $\eta$. According to (8) and (12), we define:

$$\eta = \frac{(C_{\text{eff}} - m)^{2H}}{2H^{2H}(1 - H)^{2H} \sigma^2},$$

where $C_{\text{eff}}$ is called effective service rate, and defined as follows.

**Effective service rate:** With the same self-similar traffic input, if the queue length distribution in a wireless fading channel is the same as that in a constant-rate channel, the service rate of the constant-rate channel is $C_{\text{eff}}$.

$C_{\text{eff}}$ reflects the effective service rate of a wireless fading channel with a certain traffic flow. By solving (18), we have:

$$C_{\text{eff}} = \left[2H^{2H}(1 - H)^{2H} \sigma^2 \eta \right]^{\frac{1}{2H}} + m.$$

Based on the simulation results in section IV, $C_{\text{eff}}$ can be calculated as shown in Table I-III.

Considering the self-similarity of traffic and the fading characteristic of channels in wireless networks, this paper have studied the queuing system with self-similar traffic and variable service rate in wireless channels. A convenient and efficient algorithm has been proposed to estimate the queue length distributions for this queuing system. By taking samples of the queue, our estimation algorithm works well with no need for the source and the channel parameters. Simulation results have shown that the proposed estimation algorithm can predict queue length distributions accurately in different wireless channels (two-state Markov channel, Rayleigh fading channel and Rician fading channel). We have further analyzed effective service rates of wireless channels based on our estimation algorithm, and have demonstrated that in a certain wireless channel, if the input traffic flows have the same mean and variance, the effective service rate increases with the value of the Hurst parameter.

In the future, we plan to use our estimation algorithm in resource management, meanwhile, analyze the delay and packet loss characteristics with self-similar traffic input in wireless channels.

VI. CONCLUSIONS

<table>
<thead>
<tr>
<th>TABLE I.</th>
<th>$C_{\text{eff}}$ IN RAYLEIGH FADING CHANNEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$ (bit/ms)</td>
<td>58.303</td>
</tr>
<tr>
<td>$H$</td>
<td>0.66</td>
</tr>
<tr>
<td>$C_{\text{eff}}$ (bit/ms)</td>
<td>51.991</td>
</tr>
<tr>
<td>$C_{\text{eff}}/C$</td>
<td>0.8917</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE II.</th>
<th>$C_{\text{eff}}$ IN TWO-STATE MARKOV CHANNEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$ (bit/ms)</td>
<td>55.065</td>
</tr>
<tr>
<td>$H$</td>
<td>0.66</td>
</tr>
<tr>
<td>$C_{\text{eff}}$ (bit/ms)</td>
<td>52.387</td>
</tr>
<tr>
<td>$C_{\text{eff}}/C$</td>
<td>0.9514</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE III.</th>
<th>$C_{\text{eff}}$ IN RICIAN FADING CHANNEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$ (bit/ms)</td>
<td>56.467</td>
</tr>
<tr>
<td>$H$</td>
<td>0.66</td>
</tr>
<tr>
<td>$C_{\text{eff}}$ (bit/ms)</td>
<td>53.434</td>
</tr>
<tr>
<td>$C_{\text{eff}}/C$</td>
<td>0.9463</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENT

This work was supported by National Basic Research Program of China (973 Program: 2007CB310602), 863 project (SQ200802L1480497), and USTC Research Fund for Young Teachers.

REFERENCES


Chang Liu was born in 1983. He received his B.S. degree in Electrical Engineering (EE) from University of Science and Technology of China (USTC), Hefei, China, in 2006. Since 2008, he has been working toward the Ph.D. degree in EE at USTC. His research interests include radio resource management in heterogeneous networks, network traffic modeling.

Xiaowei Qin was born in JiangSu, China, in 1979. He received the B.S. and Ph.D. degrees from the Department of Electrical Engineering and Information Science, University of Science and Technology of China (USTC), Hefei, China, in 2000 and 2008, respectively.

Since 2002, he has been a member of staff in Wireless Information Network Laboratory at USTC. His research interests include optimization theory, service modeling in future heterogeneous networks, and self-organized networks.

Sihai Zhang received the B.S. degree in Computer Science (CS) from Ocean University of China, in 1996, and the M.S. and Ph.D. degrees in CS from University of Science and Technology of China (USTC), Hefei, China, in 2002 and 2006, respectively.

He is currently a lecturer in the department of Electronic Engineering and Information Science, USTC. His research interests include evolutionary computation, wireless communication network, and social network.

Wuyang Zhou received his B.S. and M.S. degrees from Xidian University, Xi’an, China, in 1993 and 1996, respectively, and Ph.D. degree in Electrical Engineering (EE) from University of Science and Technology of China (USTC), Hefei, China, in 2000.

He is currently a professor in the department of Electronic Engineering and Information Science, USTC. His research interests include error control coding, radio resource allocation, and wireless networking.
Call for Papers and Special Issues

Aims and Scope.

Journal of Networks (JNW, ISSN 1796-2056) is a scholarly peer-reviewed international scientific journal published monthly, focusing on theories, methods, and applications in networks. It provides a high profile, leading edge forum for academic researchers, industrial professionals, engineers, consultants, managers, educators and policy makers working in the field to contribute and disseminate innovative new work on networks.

The Journal of Networks reflects the multidisciplinary nature of communications networks. It is committed to the timely publication of high-quality papers that advance the state-of-the-art and practical applications of communication networks. Both theoretical research contributions (presenting new techniques, concepts, or analyses) and applied contributions (reporting on experiences and experiments with actual systems) and tutorial expositions of permanent reference value are published. The topics covered by this journal include, but not limited to, the following topics:

- Network Technologies, Services and Applications, Network Operations and Management, Network Architecture and Design
- Next Generation Networks, Next Generation Mobile Networks
- Multimedia Communications, Communications QoS
- Information, Communications and Network Security, Reliability and Performance Modeling
- Network Access, Error Recovery, Routing, Congestion, and Flow Control
- BAN, PAN, LAN, MAN, WAN, Internet, Network Interconnections, Broadband and Very High Rate Networks,
- Wireless Communications & Networking, Bluetooth, IrDA, RFID, WLAN, WMAX, 3G, Wireless Ad Hoc and Sensor Networks
- Data Networks and Telephone Networks, Optical Systems and Networks, Satellite and Space Communications

Special Issue Guidelines

Special issues feature specifically aimed and targeted topics of interest contributed by authors responding to a particular Call for Papers or by invitation, edited by guest editor(s). We encourage you to submit proposals for creating special issues in areas that are of interest to the Journal. Preference will be given to proposals that cover some unique aspect of the technology and ones that include subjects that are timely and useful to the readers of the Journal. A Special Issue is typically made of 10 to 15 papers, with each paper 8 to 12 pages of length.

The following information should be included as part of the proposal:

- Proposed title for the Special Issue
- Description of the topic area to be focused upon and justification
- Review process for the selection and rejection of papers.
- Name, contact, position, affiliation, and biography of the Guest Editor(s)
- List of potential reviewers
- Potential authors to the issue
- Tentative time-table for the call for papers and reviews

If a proposal is accepted, the guest editor will be responsible for:

- Preparing the “Call for Papers” to be included on the Journal’s Web site.
- Distribution of the Call for Papers broadly to various mailing lists and sites.
- Getting submissions, arranging review process, making decisions, and carrying out all correspondence with the authors. Authors should be informed the Instructions for Authors.
- Providing us the completed and approved final versions of the papers formatted in the Journal’s style, together with all authors’ contact information.
- Writing a one- or two-page introductory editorial to be published in the Special Issue.

Special Issue for a Conference/Workshop

A special issue for a Conference/Workshop is usually released in association with the committee members of the Conference/Workshop like general chairs and/or program chairs who are appointed as the Guest Editors of the Special Issue. Special Issue for a Conference/Workshop is typically made of 10 to 15 papers, with each paper 8 to 12 pages of length.

Guest Editors are involved in the following steps in guest-editing a Special Issue based on a Conference/Workshop:

- Selecting a Title for the Special Issue, e.g. “Special Issue: Selected Best Papers of XYZ Conference”.
- Sending us a formal “Letter of Intent” for the Special Issue.
- Creating a “Call for Papers” for the Special Issue, posting it on the conference web site, and publicizing it to the conference attendees.
- Information about the Journal and Academy Publisher can be included in the Call for Papers.
- Establishing criteria for paper selection/rejections. The papers can be nominated based on multiple criteria, e.g. rank in review process plus the evaluation from the Session Chairs and the feedback from the Conference attendees.
- Selecting and inviting submissions, arranging review process, making decisions, and carrying out all correspondence with the authors. Authors should be informed the Author Instructions. Usually, the Proceedings manuscripts should be expanded and enhanced.
- Providing us the completed and approved final versions of the papers formatted in the Journal’s style, together with all authors’ contact information.
- Writing a one- or two-page introductory editorial to be published in the Special Issue.

More information is available on the web site at http://www.academypublisher.com/jnw/.
### REGULAR PAPERS

<table>
<thead>
<tr>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measuring the Botnet using the Second Character of Bots</td>
<td>98</td>
</tr>
<tr>
<td><em>Zhitang Li, Jun Hu, Zhengbing Hu, Bingbing Wang, Liang Tang, and Xin Yi</em></td>
<td></td>
</tr>
<tr>
<td>Control Strategy of Scheduling Invalidation of Data Broadcasting</td>
<td>106</td>
</tr>
<tr>
<td><em>Wen-bin Hu, Zhengbing Hu, Fu-qiang Yang, Cun-lian Fan, Lei Wang, and Xin-long Tan</em></td>
<td></td>
</tr>
<tr>
<td>A Novel Iterative Multilateral Localization Algorithm for Wireless Sensor Networks</td>
<td>112</td>
</tr>
<tr>
<td><em>Shaoping Zhang, Guohui Li, Wei Wei, and Bing Yang</em></td>
<td></td>
</tr>
<tr>
<td>Analysis of Queuing Behaviors with Self-similar Traffic in Wireless Channels</td>
<td>120</td>
</tr>
<tr>
<td><em>Chang Liu, Xaowei Qin, Sihai Zhang, and Wuyang Zhou</em></td>
<td></td>
</tr>
</tbody>
</table>