

# Resolution of a Combinatorial Problem using Cultural Algorithms

Alberto Ochoa<sup>1,2</sup>

<sup>1</sup> Universidad Autónoma de Ciudad Juárez, Instituto de Ingeniería y Tecnología, Ciudad Juárez, México

<sup>2</sup> CIATEC (Centro Conacyt), León; México.

Email: megamax8@hotmail.com

Julio Ponce<sup>3</sup>, Arturo Hernández<sup>4</sup> & Liang Li<sup>5</sup>

<sup>3</sup> Artificial Intelligence Laboratory, Aguascalientes University; Aguascalientes, México

<sup>4</sup> Centro de Investigación en Matemáticas, Guanajuato, México

<sup>5</sup> Technical University of Singapore, Singapore City; Singapore.

Email: julk\_cpg@hotmail.com

**Abstract**— Many problems involve not structured environments which can be solved from the perspective of Bioinspired Algorithms (Cultural Algorithms). In this paper, a proposed algorithm is used to resolve a famous game known as Japanese puzzles, which are analyzed for obtain the optimal solution. The authors show that Japanese Puzzles are constrained combinatorial optimization problems, which can be solved using Cultural Algorithms. Other features, such the use of a belief space involve many proposed solutions and local search heuristics; can also be taught using these puzzles.

**Index Terms**— Multiobjective problem, Japanese Puzzles, Cultural Algorithms

## I. INTRODUCTION

Bioinspired algorithms (BAs) are a class of population-based search and optimization techniques that work on a principle inspired by nature: evolution of species [1]. BAs have proven to be very useful tools in a large number of applications in optimization, control, signal processing, or machine learning [1, 2] using different Bas to obtain a better resolution.

Japanese puzzles, also known as nonograms, are a form of logic drawing: the puzzler gradually makes a drawing on a grid, by means of logical reasoning. This task can be mimicked by using techniques from Artificial Intelligence. The solution is usually unique. Japanese puzzles are very popular in Latin America nowadays and are sold at every newspaper stand.

In a Japanese puzzle, the numbers in rows and columns represent how many blocks of cells must be filled in the grid, in the order in which they appear. With two or more numbers, the blocks of cells in the grid must be separated by at least one blank square. In addition, an objective function, which compares the number of 1s and 0s in each line (row or column) of a puzzle solution  $X$  with the desired number of 1s and 0s, can be considered. This problem is a constrained combinatorial optimization problem. The puzzler is provided with information about the horizontal and vertical arrangement of the black pixels along every line. Figure 1 shows an image and its corresponding horizontal and vertical description. For each line, the description indicates the sizes of the

segments of consecutive black pixels, in the order in which they appear on the line. The problem of solving Japanese puzzles is NP-complete; several complexity results concerning Japanese puzzles, e.g., on uniqueness, were derived in [5]. In the next section we will give a short description of the evolutionary algorithm applied.

## II. ALGORITHM OVERVIEW

We have adapted an evolutionary algorithm [3] for solving multiobjective problems when an evaluation function is given. The algorithm optimizes exclusively over the set of all images that satisfy the prescribed projections. At the end of each époque all candidate solutions have the prescribed horizontal and vertical projections. This requires a suitable crossover operator that is not only capable of mixing features from both solutions, but also of ensuring that the produced solution. Similar requirements apply to the mutation operator.

A Japanese puzzle is an interesting and addictive game, which takes the form of a  $N * M$  grid, with numbers situated on the left, top rows, and columns, see figure 1. In a Japanese puzzle, the numbers in rows and columns represent how many blocks of cells must be filled in the grid, in the order in which they appear. With two or more numbers, the blocks of cells in the grid must be separated by at least one blank square. A Japanese puzzle is a constrained combinatorial optimization problem: consider a Japanese puzzle defined in an  $N * M$  grid ( $N$  rows,  $M$  columns). Since the puzzles are formed by black (filled) and white (blank) squares to be fixed in the grid, the puzzle can be reformulated in terms of finding a  $N * M$  binary matrix, with the number of 1s given by the conditions in rows and columns, separated by at least one 0 when more than one condition is present. Our algorithm is based on a Cultural Algorithm [2] for another view on this approach: after every VIP a stochastic hillclimb is performed until the solution has reached a local optimum. In this way, population' belief always represent local optima in the search space.



Subsequently, we again use the weighted network ow model to construct an adapted image C, which has the same value as M in as many entries as possible. This will result in a child image that is in A(R, S), resembles the first parent in a certain part and resembles the other parent in the rest of the image.

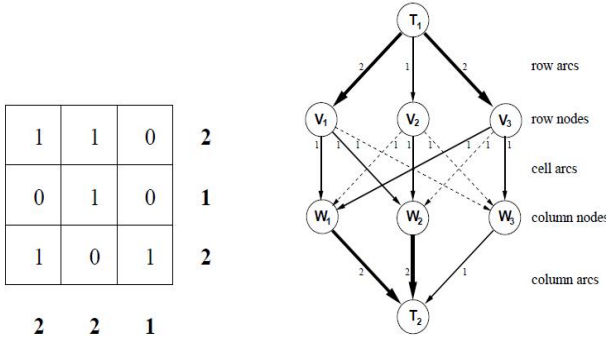


Figure 3: A 3\*3 image instance and its corresponding network ow.

To ensure that the adapted image has sufficient quality, we apply a local hillclimb operator after the crossover operation. Figure 4 show the basic principle of the hillclimb operator. Pairs of black and white entries are swapped as long as it is possible to improve the evaluation function in this way. The mutation operator uses similar principles. First, a mutation mask is generated which determines a small part of the image that will be distorted. By using the network ow method and subsequent hillclimbing an adapted image is generated that adheres to the prescribed projections. Over the set of all images that satisfy the prescribed projections over the set of all images that satisfy the prescribed projections. At the end of each époque all candidate solutions have the prescribed horizontal and vertical projections. This requires a suitable crossover operator, which is not only capable of mixing features from both proposed solutions.

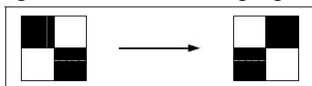


Figure 4: Swapping pairs of black and white pixels in the indicated way does not change the projections.

III. SOLVING JAPANESE PUZZLES

We now turn to the problem of solving Japanese puzzles. As summation of the segment sizes of a line yields the total number of black pixels on that line, Japanese puzzles can be considered to be a special form of the multiobjective problem, in which extra a priori information is available. We will construct an evaluation function which encapsulates all this extra information [6]. The evaluation function should reflect the deviation of a given image from the horizontal and vertical descriptions. We consider this deviation separately for each (horizontal or vertical) line. We can then obtain an evaluation function for the whole image by summation of the deviations of all lines. The minimal deviation possible, 0, should of course correspond to a solution of the puzzle. We refer to the evaluation of a pixel line  $\square$  which has description  $s$  by  $ds(\square)$ . Ideally, the function  $ds$  should make full use of all information available: the value of

the pixels on  $\square$  and the prescribed description  $s$  of  $\square$ . Let  $(\square_1, \square_2, \dots, \square_k)$  be the pixel values of  $\square$  (where  $k = n$  if  $\square$  is horizontal,  $k = m$  if  $\square$  is vertical). We call the operation of changing the value of one pixel of  $\square$  a bitflip operation. We now define  $ds(\square)$  to be the minimal number of bitflip operations required to make  $\square$  conform to  $s$ .

This definition of  $ds$  has several advantages. Firstly, it is a very intuitive way of defining the distance between a given line of pixels and its prescribed description and it uses all available information. If a line  $\square$  adheres to its description  $s$ , we have  $ds(\square) = 0$ , as desired. Secondly,  $ds$  has the property that performing a single bitflip operation on  $\square$ , yielding a line  $\square'$ , can change the value of  $ds$  by no more than 1. In a discrete sense,  $ds$  can be regarded as a fluent function of  $\square$ . Surprisingly,  $ds(\square)$  can be computed quite efficiently by means of dynamic programming. This is of course necessary since the algorithm requires many of these computations. Suppose that the prescribed description  $s$  of  $\lambda$  consists of  $h$  segments of black pixels,  $(s_1, \dots, s_h)$ . Without loss of generality, we may assume that starts and ends with a white pixel: adding white pixels at the beginning or end of a line does not change its description. We define a line segmentation  $\square^* = (\square^*_1, \dots, \square^*_{h+1})$ , corresponding to the number  $h$  of black =  $\begin{cases} 0 & \text{(white) if } i \text{ is odd} \\ 1 & \text{(black) if } i \text{ is even} \end{cases}$

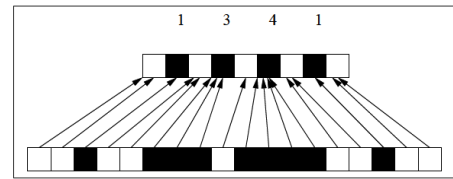


Figure 6: A line description (top), the corresponding line segmentation (middle), and an actual line that adheres to the description (bottom). The arrows indicate the links between the line and the segmentation.

The entries of the line segmentation correspond to the alternating black and white segments in the description (where we consider the white segments to be implicitly present). Figure 6 shows a line description  $s$ , its corresponding line segmentation  $\square^*$  and a realization of  $s$ , which is a line  $\square$  that adheres to  $s$ . The arrows indicate for each pixel of  $\square$  to which entry of  $\square^*$  it corresponds. If pixel  $i$  corresponds to entry  $j$  of  $\square^*$ , we say that pixel  $i$  is linked to entry  $j$  of the line segmentation. We call the corresponding mapping a link mapping. If  $\square$  does not correspond to  $s$ , we can also link every pixel  $i$  consecutively to an entry  $j$  of  $\square^*$ . In that case, however, there are pixels in  $\square$  that do not match the color of the entries of  $\square^*$  to which they are linked. Some bitflip operation will have to be applied to these pixels in order to make  $\square$  conform to  $\square^*$ , with the given links. Valid link mappings must satisfy several requirements: consecutive pixels should be linked either to the same entry of  $\square^*$  or to consecutive entries of  $\square^*$ . The number of pixels linked to each black entry  $\square^*_j$  must be  $s_j/2$ . There must be at least one pixel linked to each white entry of  $\square^*$ .

A valid partial link mapping from  $(\square_1, \dots, \square_i)$  to  $(\square^*_1, \dots, \square^*_j)$  satisfies the same requirements as a total valid link mapping. In addition, if pixel  $i$  is linked to a black entry  $\square^*_j$ , the pixels  $(\square_{i-s_j/2+1}, \dots, \square_{i-1})$  should also be linked to entry  $j$ . We now introduce  $\delta(i, j)$  (for  $1$  to  $i$  until  $k$ ,  $1$  to  $j$  until  $2h + 1$ ), the minimal number of bitflip operations, over all valid partial link mappings from  $(\square_1, \dots, \square_i)$  to  $(\square^*_1, \dots, \square^*_j)$ , that has to be applied to  $\square$  in order to make it conform to the partial link mapping (so that all pixels have the same colors as the entries to which they are linked). We remark that  $\delta(k, 2h+1)$  is the minimal number of bitflip operations that is required to transform  $\square$  into a line that adheres to  $s$ . We can directly use it as the evaluation function for our algorithm. Fortunately,  $\delta(i, j)$  can be computed efficiently by means of dynamic programming. For both the case that  $j$  is odd and the case that  $j$  is even, it is possible to construct a recurrence relation which computes  $\delta(i, j)$  from other values  $\delta(i', j')$  where always  $i' < i$ . By using a nested loop, which iterates over all pairs of  $(i, j)$  in the right order, each value  $\delta(i, j)$  can be computed from table-values that are already known.

IV. RESULTS AND CONCLUSIONS

We implemented the evaluation function and performed several test runs. We used a population of agents of 500 for the experiments. Figure 7 shows the test images that we used. The example has size  $20 \times 20$  which is known by to have a unique solution. We performed one test run for this image. The image was reconstructed perfectly on a Pentium IV at 2.4 GHz in 80 minutes after 27 époques of the algorithm. Note that the reconstruction is quite different from the original image, yet it adheres to nearly all line descriptions. Although the reconstruction process took a long time to complete, this is still a very positive result, since this image is very hard to reconstruct using only logic reasoning. Branching seems inevitable for that type of image.

We have adapted an evolutionary algorithm [3] for solving multiobjective problems when an evaluation function is given. The algorithm optimizes exclusively over the set of all images that satisfy the prescribed projections. At the end of each époque all candidate solutions have the prescribed horizontal and vertical projections. This requires a suitable crossover operator that is not only capable of mixing features from both solutions, but also of ensuring that the produced solution. Similar requirements apply to the mutation operator.

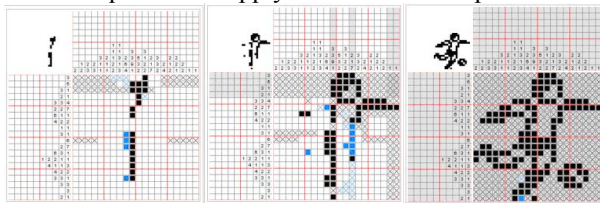


Figure 7: Resolution of a Japanese Puzzle using our algorithm.

Future research includes a detailed analysis of the performance on puzzles of different difficulty levels and different types (e.g., more than two colors) [7]. The fact

that the evolutionary algorithm from [1] can also be used to solve Japanese puzzles, which it was not specifically designed for, clearly demonstrates its versatility.

REFERENCES

- [1] K.J. Batenburg. An evolutionary algorithm for discrete tomography. Discrete Applied Mathematics ICW Congress, 2004.
- [2] K.J. Batenburg and W.J. Palenstijn. A new exam timetabling algorithm. In T. Heskes, P. Lucas, L. Vuurpijl, and W. Wiegierinck, editors, Proceedings of BNAIC 2003, the Fifteenth Belgium-Netherlands Artificial Intelligence Conference, pages 19-26, 2003.
- [3] Z. Michalewicz. Genetic Programs + Data Structures = Evolution Programs. Springer-Verlag, third edition, 1996.
- [4] H.J. Ryser. Combinatorial properties of matrices of zeros and ones. Canadian Journal of Mathematics, 9:371-377, 1957.
- [5] N. Ueda and T. Nagao. NP-completeness results for nonogram via parsimonious reductions. Technical Report TR96-0008, Department of Computer Science, Tokyo Institute of Technology, 1996.
- [6] A. Ochoa et al. Applying Cultural Algorithms and Data Mining to organize elements in a Social Dyoram, In proceedings of ENC'08-Workshop of Data Mining, 2008.
- [7] A. Ochoa et al. Una comparativa de la Inteligencia Grupal desde la perspectiva del Cómputo Evolutivo. In Proceedings of CIICC'08, Aguascalientes, México; 2008.

**Alberto Ochoa Ortiz-Zezzatti** (Bs'94–Eng.Master'00–PhD'04–Postdoctoral Researcher'06 & Industrial Postdoctoral Research'09). He joined the Juarez City University in 2008. He has 1 book, and 7 chapters in books related with AI. He has supervised 8 PhD theses, 11 M.Sc. theses and 27 undergraduate theses. He participated in the organization of COMCEV'07, COMCEV'08, HAIS'07, HAIS'08, HAIS'09, ENC'06, ENC'07, ENC'08 and MICAI'08. His research interests include evolutionary computation, natural processing language, anthropometrics characterization and Social Data Mining.

**Julio Ponce** is currently a PhD student in Evolutionary Computation from the UAA in Aguascalientes, Mexico. His research interests include Artificial Intelligence, Evolutionary Computation, and Data Mining. M.S. Ponce was organizer member of the COMCEV'05, COMCEV'07, and reviser member in CISCI'07, CISCI'08, CISCI'09 and COMCEV'07.

**Arturo Hernández Aguirre** received a BSc in Electronics from Universidad Autónoma Metropolitana in 1982, and an MSc and a PhD in Computer Science from Tulane University in 1998 and 2001, respectively. Hernández joined the Center for Research in Mathematic in 2001, where he works as a researcher at the Computer Science Department. He has authored over 16 journal papers, 100 conference papers and six book chapters. His research interests include evolutionary computation and bio inspired algorithms for global and constrained optimization.

**Liang Li** (BS '95–Master'99–PhD'03). She obtained a PhD from Salford University in Manchester, United Kingdom with a scholarship from Commonwealth Foundation. She realized in 2007 an Intership in Imperial Tokyo University in Japan; actually she has over 47 papers related with Artificial Intelligence in the developing of Intelligent Systems. She participated in the Technical Committee of HAIS'08.

Her research interests include Hybrid Artificial intelligent Systems.