A Criterion-based Genetic Algorithm Solution to the Jigsaw Puzzle NP-Complete Problem

Francisco Gindre, David A. Trejo Pizzo, Gabriel Barrera, and M. Daniela Lopez De Luise

***Abstract*—The foremost objective of the present research project consists developing an Intelligent Robotic System (SIR for its name in Spanish, Sistema Inteligente Robótico) that solves an unknown jigsaw puzzle in a reduced amount of time. To fulfill the objective, SIR applies pattern recognition techniques as edge and feature detection in conjunction with Genetic Algorithms. Many authors have addressed the jigsaw puzzle problem. By comparing their work, the NP-Complete nature of the problem appeared as a common denominator on them. SIR relies on those experiences, results, conclusions and observations as a working base for a new approach to the problem. This approach involves a conversion of the puzzle model to a Graph Theory model. The study of this model plus the inclusion of a different analogy and solution approach is the main contribution of this work. It describes the theoretical and practical frameworks, current state of the project and future work.**

***Index Terms*—graph, jigsaw, puzzle, genetic, algorithm.**

I. Introduction

According to the American Jigsaw Puzzle Society, in 1760 John Spilsbury, a London engraver and mapmaker, produced the first jigsaw puzzle [10]. Since then, several different manufacturers around the world are manufacturing jigsaw puzzles in a variety of shapes, sizes and piece types. Despite all the different types of jigsaw puzzles that can be found, there is a lack of common classifying terminology that identifies uniquely each type of jigsaw puzzle. The background research for this project came across this issue. Certain common definitions aroused from different authors about the topic. Some of the classification terms are adopted by the present project and briefly described here. The term Standard jigsaw puzzle (see a sample figure 1.1) is used in [2] to describe puzzles made by cutting pictures printed on firm substrates into interlocking patterns of pieces. It should be noted that standard class only denotes the interlocking nature of the puzzle along with the basic manufacturing process; it does not refer to the size or shape of the piece or the puzzle itself.

F. Gindre is with Artificial Intelligence Group (AIGROUP), Universidad de Palermo, Ciudad Autonoma de Buenos Aires, Argentina (e-mail: fgindre@palermo.edu).

D. Trejo Pizzo is with Artificial Intelligence Group (AIGROUP), Universidad de Palermo, Ciudad Autonoma de Buenos Aires, Argentina (e-mail: dtrejo@palermo.edu).

G. Barrera is with Artificial Intelligence Group (AIGROUP), Universidad de Palermo, Ciudad Autonoma de Buenos Aires, Argentina (e-mail: gmbarrera@gmail.com).

M. D. Lopez De Luise is with Artificial Intelligence Group (AIGROUP), Universidad de Palermo, Ciudad Autonoma de Buenos Aires, Argentina (e-mail: aigroup@palermo.edu).

The term square jigsaw puzzle (Figure 1.2) refers to puzzles having square pieces featuring straight borders and uniform size which conform a grid that completes the solution image, as they lack of a curvilinear shape, it is possible (depending on the image) that this kind of jigsaw puzzles have multiple solutions to the same picture.

There is jigsaw puzzle superset called canonical puzzles, defined by [11]. It includes all puzzles having four edge pieces that can be rotated to four different orientations. The outcome of placing the pieces contiguously constitutes a rectangular grid conforming the resultant picture. The latter definition makes no distinction whether the border is curved or straight. Other approaches were found as well, conforming apictorial puzzles (figure 1.3 and 1.4) [9]. These particular puzzles have no picture or distinguishable chromatic features that could lead to their assembly; therefore their solution relies purely on piece's boundary shape.

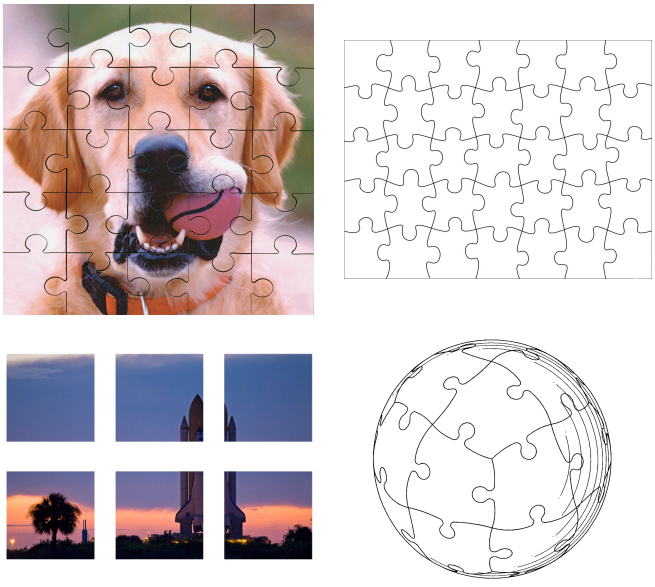

Figure 1.1 – Standard Jigsaw Puzzle (top left corner)
Figure 1.2 – Square Jigsaw Puzzle (bottom left corner)
Figure 1.3 – Apictorial Canonical Puzzle (top right Corner)
Figure 1.4 – Apictorial puzzle (bottom right corner)

The present paper describes a new approach to solve square jigsaw puzzles, developed as part of the SIR prototype, a project of the AIGroup research lab.

Many authors have tackled the jigsaw puzzle problem over the last four decades. In 1968, Freeman and Garder [9] addressed the assembly of *apictorial* jigsaw puzzles as they recognized the technical limitations at that time to gather

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

other useful information than the shape of the pieces. Their work conform a common ground for several other works [1][2][3][5][7][8][11][12], including this research project.

Capturing a jigsaw puzzle into a computerized model involves computer vision techniques such as shape description, partial boundary matching, pattern recognition, feature extraction, and heuristic matching. In addition, the problem complexity is made apparent by its characteristics: compatible to Wang's tiling problem [6] and NP-complete type [2][13][7].

It becomes an interesting field of study for areas such as non-linear optimization [4], constraint programming [2] and dynamic programming [12]. It has also a diverse set of application domains. Additionally to solving jigsaw puzzles, the task of reassembling complex objects has great importance in the fields of archeology, forensics, art restoration and engineering. Reconstruction of broken documents, archeological restoration and reconstruction of disassembled machines are complex and time consuming tasks that makes evident the need of an automatic solution to that issue [12][20].

As a response for that necessity, SIR's main goal is to build a low-cost robotics system with the ability of building an unknown square jigsaw puzzle in a reduced amount of time. SIR's design consists of three independent modules that interact to fulfill its goals. The first one is the Capture Module, which applies computer vision techniques to extract the jigsaw puzzle pieces from the environment (called playground). The second is the Logical Module, which is in charge of receiving the extracted pieces and providing a solution for the puzzle and an execution algorithm. The third component is the Robotics Module that executes that algorithm and places the puzzle pieces in the correct order. The present paper describes the design and development of the Logical model using a new approach based on Graph Theory [25], its background and its contribution to the subject. The following paragraphs on this section cover the background of the jigsaw puzzle assembly problem and describe main scientific community contributions.

Since [9], many other authors suggested and developed different approaches to reach the jigsaw puzzle solution. *Apictorial puzzles* [8] are used as a way of solving the part assembly problem. Its goal is to develop software that can reconstruct an object by comparing the physical features of its composing parts. There, Burdea and Wolfson make use of a robotics arm is used to assemble the white puzzle after the pieces are processed and the solution has been reached. The work introduced the concept of global solution to the scene. The solution algorithm implemented, called KBEST, recreates the assembly process done by human beings. As the first step it identifies the border pieces of the puzzles and assembles the exterior frame of the puzzle by the use of heuristics and a similar approach of the traveler salesman problem (TSP [4]), this process iterates over the inner frames of the puzzle using the k best solutions found in different iterations until a global solution is reached. This approach can solve puzzles up to 208 pieces. Iterative solution that also discards pictorial information is proposed in [14]. It addresses the boundary-matching problem by finding the Isthmus critical points of the borders and then iteratively tests all the possible matching borders. This approach is capable of assembling up to 24 pieces *standard jigsaw puzzles*. The KBEST approach shown in [8] is revisited in [5]. In that paper Chung proposes assembling *standard* jigsaw puzzles using both shape and color, claiming that relying on only one of the features, due to the similarity of the pieces would lead to incorrect solutions. The environment is captured by a 6mm camera using the CIE LAB Colorspace [28] and the puzzle pieces are extracted from the background color board they are placed on. They designed three algorithms: AP, that recreates the "human" approach by attempting to solve first the external frame and then moving to the inner part of the puzzle; TSP/AP that combines the AP approach with the strategy of dealing with the similarity of the jigs as if it is the distance in the TSP problem. The third algorithm combines the TSP approach with the KBEST algorithm. The performance of the last algorithm is the best of all three but it could only assemble certain puzzles with a limit of 54 pieces maximum. The main contribution is bringing the TSP issue to the problem. The TSP [4] approach in conjunction to the global solution [8] leads the way to the use of genetic algorithms [15][16].

This technique is used in [3] to solve a *square jigsaw puzzle* of m by n pieces using color information. The border color composition is the only information required in this approach. Each piece is captured in a 1-bit black and white palette, where each border is composed by the piece's external pixels. During this process the inner information is discarded, as it is not considered for the comparison. This approach could find the global solution of certain given set of segmented images up to an 8x8 dimension. Years later this technique is reused in [1] with a different perspective. The pieces are captured in RGB, containing more detailed information about each border. Also the assembly of the puzzle would be done not from individual pieces. Instead, pieces are first assembled in blocks with optimal distances, later those blocks are compared as if they were single pieces. The genetic algorithm (GA) optimizes distance between pieces to build optimal blocks that are compared with other blocks and stand-alone pieces. The main advantage of this method is that optimal blocks are not affected by further fortuitous operations on the chromosomes, reducing the random nature of the GA mating process. This approach is capable of assembling up to 16x12 puzzles. Tybon's Thesis [2] presents an interesting survey of the "state of the art" and the different methods that could be used to solve *square jigsaw puzzles.* It features a showcase of possible approaches to the puzzle problem solution that include Linear Programming [21], Evolutionary Programming and GA [15][16], Simulated Annealing (SA), Tabu search [18][19] and Constraint Driven Programming. Tybon implemented these different techniques and tested them using 2x2, 2x3, 2x4, 2x5, 3x4, 3x5 and 4x5 puzzles with unique solutions. The thesis concludes that the GA and the SA approaches are the most suitable ones. A recent

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

publication shows an alternative method for solving jigsaw puzzles by using dynamic programming and an optimal assignment procedure like the Hungarian procedure, showing better performance than the piece-by-piece. These techniques are used to solve an image-scrambling problem with tiles in a 10x10 layout [12] considering the texture of contiguous tiles.

The above background analysis summarizes different approaches to solve puzzle problems. According to [5][9][11][20] it is possible to infer that the extraction of the material pieces and their subsequent inclusion in the computerized model is not the key factor that limits the amount of pieces that those solutions can handle. On the contrary, the means taken to manage the NP-Completeness nature of the problem determined the scalability of those approaches. Based on this conclusion, SIR's logical module has the highest implementation priority. Therefore this publication focuses on that particular subsystem. The following section describes briefly the overall architecture of SIR contextualizing it location on the general workflow. Section 3 discusses the new approach design plus its advantages and setbacks. Lastly Section 4 illustrates future work for the next quarterly periods.

II. Architecture, Workflow and Boundaries

As described in the previous section, SIR is designed as a system consisting of three independent modules interacting towards the solution of a square jigsaw puzzle. The first, the capture module: composed by a regular webcam and a software module that articulates a media framework to control the camera and triggers the capture. The design of this module contemns the inclusion of Gaussian and edge detection filters to be applied to the captured image along with feature detection techniques and algorithms for unique identification of the square jigsaw puzzle and its pieces. The output of this module is the set of pieces identified by order of appearance and location that will serve as the input and trigger for the subsequent module. The puzzle data is converted to an abstraction of that image capture changing the focus from a pixel matrix (image captured by webcam) to well formed and identified pieces of a constituted square jigsaw puzzle. Each piece is uniquely identified and their borders are considered the only region of interest required to solve the problem [1][2][3]. Puzzle dimension is yet unknown until the output of this module is made. The Logical Module uses a Genetic Algorithm to find a solution to the puzzle given by abstracting the original problem leading it to a graph optimization one (described in section 3). The output of this process produces the concrete solution and the puzzle dimension. The Robotic Module is the final step of the workflow. It's designed to consume the output of the previous module and execute the assembly of the real puzzle. The Robotic Module is composed by a Lego MindStorms NXT® robotic kit [22], assembled as a robotic main controller. It is connected through a Bluetooth® [23] interface with a laptop containing SIR software. The result from the previous module is parsed and executed, guiding the robot controller through the necessary steps to complete the assembly. SIR assumes the following problem constraints in order to control its boundaries: 1) the system is limited to square jigsaw puzzles, 2) puzzle pieces are placed on top of plain surface with a distinctive chromatics spectrum to make simpler feature extraction, 3) background color is the only information given to the system, 4) pieces only admit 90º rotations (straight rotations), 5) the execution is assumed in a controlled environment complying the former constraints.

III. A new Genetic Algorithm approach

Solving a jigsaw puzzle can be defined as the discovery of a global solution that minimizes all the distance comparison between all the pieces [1][2][3][4][8][11][12]. As a matter of fact, a square jigsaw puzzle containing soft gradients or large portions of solid colors can have multiple solutions that lead to rather local maxima results [2]. A perfect match in the majority of the puzzle plus different piece rotations on the gradient or solid area can generate these multiple global maxima. Large gradients, solid or homogeneous regions tend to create local minima or maxima solutions, causing regular algorithms to converge to false positive results, if any was reached. Sub-optimal solutions are the main reason for applying GA. The factorial-exponential magnitude of the problem would require long computational time as the number of pieces increases minimally as explained in [2]. Genetic Algorithms solve these problems. Mutation, crossover and selection guarantee that local minima or maxima are breached expanding the search span, without having to iteratively validate all possible combinations [15][26]. The following sub-sections explain some important drawbacks that arise from this same technique.

As previously stated, jigsaw puzzle solving is a non-linear, constraint based optimization problem. Global distance between pieces is the target function, were the global solution must optimize edge-to-edge distance. [2] Shows that the influence constraints have on the overall result, while [3] proves effectiveness in solving some types of puzzles in despite of the limitations imposed by the analogy chosen to determine chromosome and population.

In addition to the above criteria, the NP-Completeness and distance optimization characteristics of the problem made the research to be redirected an abstraction that presents severe similarities to the TSP [4]. The TSP approach was found among the most effective in solving large puzzles [5]. In the proposed new approach (used in SIR's design) the jigsaw puzzle is represented as a multi-graph [25](without loops), where each node stands for a piece of the puzzle. Each edge establishes a border-to-border adjacency between two pieces. The GA representation was redesigned to adequate to the graph approach. The architecture of the algorithm has to support the unknowns, uncertainties and constraints present in the original problem. The uncertainty upon the amount of pieces involved in the puzzle helped the idea of the chromosome representing a candidate solution to the jigsaw puzzle to withstand over other considerations. Ergo, chromosome length is tightly coupled to the dimension of the puzzle. An overview of the proposed model, presents a fixed length chromosome. Each gene represents an edge of the graph and

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

the sides of the pieces involved in that adjacency. Details of the GA component are discussed after detailing common constraints that apply to both puzzle and graph models.

Square jigsaw puzzles have clear rules concerning adjacencies. Besides the four corner pieces remaining static in any solution, the amount of pieces presenting 2, 3 and 4-sided adjacencies [6] vary depending on the puzzle dimension. These variables are coupled to the aspect of the resulting puzzle and consequently the existence of an optimal solution for it. The other aspect of puzzle constraints is invalid adjacencies. When not properly validated, mutation and crossover operations can cause adjacencies that are either not physically or logically possible. In example, a piece of the puzzle cannot be matched with itself or present an invalid one-sided adjacency. Spurious solutions may arise when not properly validated spreading the error to subsequent iterations, setting the algorithm out of it boundaries. This led to the second drawback: the complexity of validating individuals and solutions.

To avoid possible bias in the search span, GA's are initialized with random populations [26][27]. The current approach presents an exception to that rule. Initial populations would be created in a controlled random process. Meaning that the chromosomes that initialize the algorithm must be randomly picked but yet respecting the constraints of the problem. Other operations such as crossover and specially mutation require being tightly

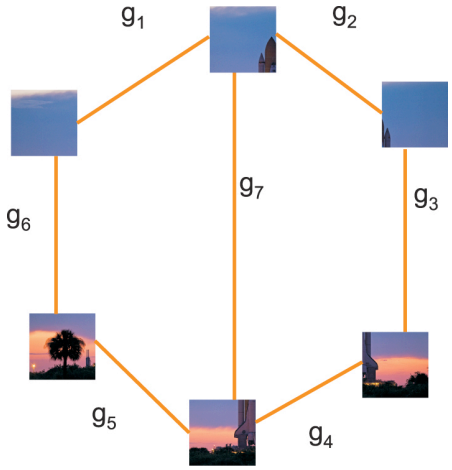

Figure 3.1 – Square Jigsaw Puzzle as a Graph

controlled and validated to ensure the GA does not outbound the given problem. This validation is done during selection and other offspring generation operations to guarantee the consistency of the entire process.

According to experimental work published in [3], Genetic Algorithms present proficient results when the puzzle to solve is up to 8x8 pieces, although computational times tend to increase considerably. The method chosen in [3] assumes fixed and known dimensions of the puzzle to solve where the solution is represented as a population formed by rows acting as individuals. Recent research conducted by the same authors in [1] improved puzzle size resolution to a maximum of 16x12. Their method consists in putting pieces that fit well together and doing a block comparison. To establish the first blocks, the method requires previous knowledge of the type of puzzle to be assembled. Along with other techniques, GA is presented as an alternative to converge to a solution of a jigsaw puzzle in [2]. Unlike [3], populations are formed with sets of solutions to the puzzles. Validations discussed above in this same section are taken in consideration by the author. These experiences lead to the design of the graph abstraction (Figure3.1). The graph theory model, offers versatility of application along with a wider search span. As discussed previously, by representing the puzzle solution as a set of adjacencies between nodes (arcs or edges of the graph) it is feasible to represent any possible layout for the jigsaw puzzle, including those that are invalid. The graph representation of the puzzle solution (chromosome) must comply with the following constraints.

1) Constraints in the graph model

Invalid adjacencies must be detected and consequently discarded. Given an m by n pieces puzzle, the numbers of edges (genes) of a chromosome are dictated by the function:

$$A(m,n) = 2mn - n - m \; . \qquad (1)$$

4-edge nodes respond to the function:

$$j(m,n) = (m-2)(n-2) \; . \qquad (2)$$

3-edge nodes are determined by this other function:

$$k(m,n) = 2(m-2) + 2(n-2) \; . \qquad (3)$$

2-edge nodes (corners) are always four. Chromosome and gene validations continue to be done in order to ensure consistency during breeding and selection. Code complexity to do these operations is reduced thanks to Graph Theory properties such as vertex degree, graph size and adjacency matrixes. A chromosome is surveyed by representing it as an Adjacency Matrix [25]. The grade of the node determines the type of adjacency that represents (2-edge, 3-edge or 4-edge). The adjacencies are summarized and compared against the expected values returned by functions (1), (2) and (3). Any chromosome not complying the constraints must be rendered *infertile*. The proposed abstraction includes validation methods mentioned in [2] as it also avoids complex evaluation of individuals and solution as separate processes and defines more parameters to narrow down the GA's search breadth.

2) Elements of the Genetic Algorithm

The fitness function is defined as:

$$F(x) = \frac{1}{2mn - m - n} \cdot \sum_{1}^{A(m,n)} (P(x_i)) \qquad (4)$$

Where P(x) is the percentage of edge-to-edge (border) coincidence between two pieces and $f(s_i)$ is a function that returns two vertices (pieces) involved in the s_i graph edge. The parameter x represents the chromosome input.

A *Population* is defined as a set of *fertile chromosomes*; it is evolved until convergence is detected. A *chromosome* represents a graph that illustrates a possible solution to the puzzle. *Chromosomes* are deeply analyzed by certain

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

criteria. If any criterion is not met, the individual is rendered infertile by assigning 0 (zero) to its fitness value. The low fitness value causes its suppression from the *population*. A *Gene* represents an edge connecting two pieces of the puzzle together in the graph abstraction model. They are evaluated in the fitness function to determine how well adapted an individual is. Recursively, genes are validated with different criteria and failure to comply a criterion would render the chromosome infertile. Genes are composed of two Alleles. Each one of those represents a border of a piece of the puzzle.

One constraint to this problem has been mentioned and not yet explained. A jigsaw puzzle is considered valid if the detection process finds a non-prime number of pieces in the playground. Meaning that, single row/column puzzles are not considered as valid puzzles in this paper or in the project. SIR's workflow requires a valid number of pieces to produce a resolution to a puzzle. Based on that number divisors and multiples, different instances of the GA are executed with the corresponding M and N parameter defining the target M by N dimension for that instance. The fittest chromosomes of each instance are placed into an elite population until the whole process concludes. Afterwards, a ranking selection is executed on the elite population to obtain the optimum solution to the puzzle.

IV. Heuristic Fitness Classification

Abstracting the puzzle into a graph model presents various advantages towards the piece-driven approach chosen in precedent works [1][2][3][8][111][12][13][20]. The graph model enlarges the GA's search span. Making the algorithm is less biased, therefore the odds of falling into local minima solutions is reduced. Also provides a flexible setting that supports different puzzle topologies than the one chosen for the current stage of the project. That flexibility is achieved by changing the criteria that define valid puzzle characteristics.

These features have setbacks attached to them. Leaving the "piece matrix" representation of the puzzle makes the model difficult to represent in both data structure and reality. This difficulty has a residual effect on the GA's workflow (figure 4.1 describes the workflow, which is based on Fig. 15.1 [27]). It is possible for the *mutation* or *cross-over* operations to dismantle a fit chromosome population, lowering its fitness value or even rendering it infertile by detaching genes that have maximum local fitness (minimum edge-to-edge distance).

Because of the nature of the problem, genes that have minimum edge-to-edge distance are more likely to be present on the puzzle optimum solution as well. To avoid these genes being removed, the GA's workflow includes a way of *tagging* genes that surpass a defined edge-to-edge distance. SIR contemplates two thresholds that can be adjusted to different edge-to-edge distance values. This creates two new classes of genes: *candidate genes* and *sticky genes. Candidate* refers to genes that have a percentage of edge-to-edge similarities that is considered "acceptable" as a part of a fit chromosome but identified as "not optimal". Sticky refers to "optimal" edge-to-edge distance two borders.

Sticky or Candidate genes are less likely to be affected by genetic operations. This heuristic allows operations *not* to dismantle fit genes during mutation or crossover. Tagging process is done by the fitness function. This capability can be switched on and off during the GA's workflow depending on the convergence evaluation.

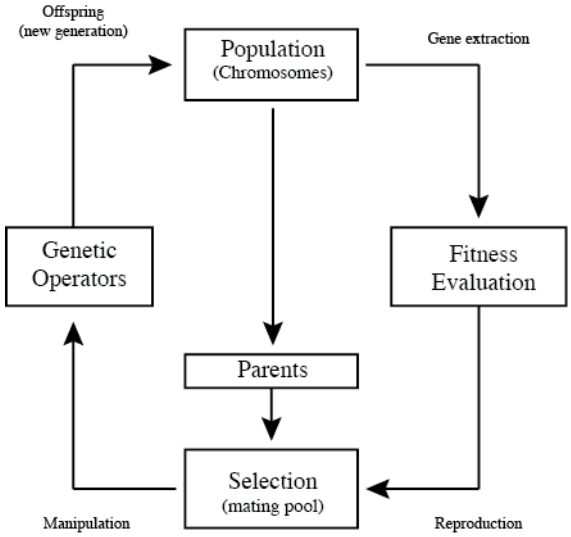

Figure 4.1 GA's workflow with out heuristic fitness usage

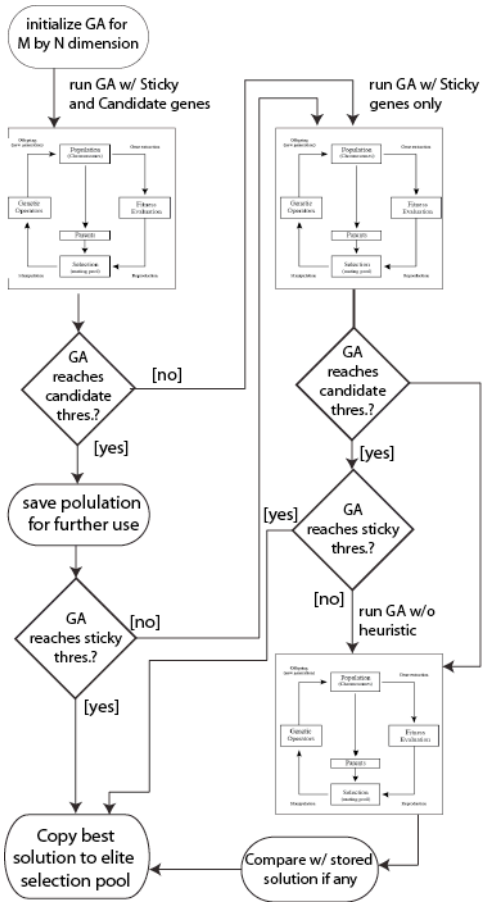

Figure 4.2 – Execution of GA using heuristic fitness classification

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

In Theory, tagging genes helps the GA to converge to optimum solutions faster. It also makes the unbiased graph based workflow, biased. Since *candidate* genes are not optimum edge-to-edge distance adjacencies, it is possible that populations evolve to sub-optimal convergence states. Anticipating that consequence, convergence is analyzed taking in consideration the fact that *locking* sub-optimal genes during the manipulation phase, affects the global solution fitness negatively. Figure 4.2 describes the overall execution of the GA using the *heuristic fitness classification.*

The workflow described in figure 4.2, is executed for every possible puzzle dimension that can be conformed from the number of pieces that were extracted by the *capture module*. A GA instance is set up an initialized for a given M by N dimension. The workflow described in figure 4.1, is executed using a fitness function and operations that can handle tagged genes. Firstly, genes are tagged as *candidate* or *sticky* according to the thresholds the GA was set up with. The overall execution goes from a biased one, to a *non-biased* one. If the heuristic methods do not help the GA to converge to optimal solutions, their usage is reduced until the last execution of the GA by itself with no use of the tagging. When sub-optimal convergence is detected, the resultant population is stored. If the algorithm fails to converge to better results, the best solution of that stored population will be selected as the best result of that given dimension.

V. Current State and Future Work

The project described in the present work, is currently in development. As SIR's *logical module* reaches completion, the different combination of the heuristic settings will be tested and their performance compared to determine whether the usage of heuristic fitness classification is a key factor in helping optimal convergence of the GA. The robotic stage is the hindmost step of the workflow and the project itself. The necessary equipment has been acquired and tested. For license and compatibility purposes, the Lego® Brick was converted to a *Java Virtual Machine* supported by the LEjOS open source framework [24]. The *Robotic Module* development, testing and deployment are scheduled for Q3 2010.

Acknowledgment

The authors would like to express their thanks to the Faculty of Engineering members that supported them and the project daily and specially, Dean Esteban Di Tada whose interest, opinion and support are of invaluable help.

References

[1] Murakami T., Toyama F. Shoji K. , Miyamichi J. : "Assembly of Puzzles by Connecting Between Blocks", (2008).
[2] Tybon R.: "Generating solutions to the Jigsaw problem." Griffith University, (2004).
[3] Toyama F., Fujiki Y. , Shoji K. , Miyamichi J. : "Assembly of puzzles using a genetic algorithm." 16th International Conference on Pattern Recognition, (2002).
[4] Papadimitriou, C.H., Yannakakis M. : "Shortest paths without a map". Proc. 16th ICALP. Lecture notes in computer science (Springer-Verlag.), (1989).
[5] Chung M. G, Fleck, M. M, Forsyth D. A.: "Jigsaw puzzle solver using shape and color." Proceedings of ICSP (IEEE), 877-880, (1998).
[6] Wang J.: Introduction to NP-Completeness, 91.502 Foundations of Computer Science, (2006).
[7] Ville Lukkaril: The square tiling problem is NP- complete for deterministic tile sets. TUCS Technical Report No 754, March, (2006).
[8] Burdea G. D., Wolfson H. J.: Solving jigsaw puzzles by a robot. IEEE Trans. Robotic. Autom. v5 i6. 752-764, (1989).
[9] Freeman H. and Garder L.: Apictorial Jigsaw Puzzles: The Computer Solution of a Problem in Pattern Recognition, IEEE Trans, on Electronic Comp., Vol EC-13, No. 2, 118-127, April (1964).
[10] McAdam D.: History of Jigsaw puzzles. American Jigsaw Puzzle Society. http://www.jigsaw-puzzle.org/jigsaw-puzzle-history.html.
[11] Yao F.H., Shao G.F.: A shape and image merging technique to solve jigsaw puzzles, Pattern Recognition Letters 24 (2003) 1819-1835. (2003).
[12] Alajlan, N.: Solving Square Jigsaw Puzzles Using Dynamic Programming and the Hungarian Procedure. American Journal of Applied Sciences 6(11): 1942-1948, (2009).
[13] Altman T.: Solving the jigsaw puzzle problem in linear time. Applied Artificial Intelligence, 3,(1989).
[14] Webster R. W., LaFollete P. S., Stafford R. L.: Isthmus Critical Points for Solving Jigsaw Puzzles in Computer Vision. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. 21, NO. 5, 1271-1278. SEPTEMBER, (1991).
[15] Holland, J. H.: Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor,(1975).
[16] Koza, J.: Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press. ISBN 0-262-11170-5, (1975).
[17] Kirkpatrik S., Gelatt C. D., Vecchi M. P.: Optimization by simulated annealing, Science, (1983).
[18] Glover F.: Tabu search- Part I ORSA Journal on computing,(1990).
[19] Glover F.: Tabu search- Part II ORSA Journal on computing,(1990).
[20] Sagiroglu M. S., Ercil A.: A texture based matching approach for automated assembly of puzzles. Proceedings of 18th International Conference on Pattern Recognition, (PR'06), IEEE Xplore Press, Hong Kong, pp: 1036-1041. DOI: 10.1109/ICPR.2006.184, (2006).
[21] Dredsner E. C.: Técnicas cuantitativas: el management científico aplicado a las decisiones en la economía de empresas. tercera edición, Editorial Universo, Buenos Aires, (1998).
[22] Lego® MindStorms NXT. http://mindstorms.lego.com/eng/default.aspx .
[23] Bluetooth Technology. http://www.bluetooth.com
[24] LeJOS, Java for Lego MindStorms. http://lejos.sourceforge.net/
[25] Balakrishnan, V. K.: Graph Theory, McGraw-Hill; 1st edition. (1997).
[26] Bigus J. P., Bigus J.: Constructing Intelligent Agents with JavaTM: A Programmer's Guide to Smarter Applications, John Wiley & Sons, (1999).
[27] Amit K.: Artificial Intelligence and Soft Computing Behavioral and cognitive modeling of the human brain, CRC Press, (2000).
[28] Schanda J.: Colorimetry. Understanding the CIE System. Wiley-interscience, (2007).