Improving Localization in Wireless Sensor Networks with An Evolutionary Algorithm

Vincent Tam, King-Yip Cheng and King-Shan Lui

Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam, Hong Kong.
{vtam,kycheng,kslui}@eee.hku.hk

Abstract— **Wireless sensor networks are highly useful for many location-sensitive applications including environmental monitoring, military applications, disaster management, etc. Localization in wireless sensor networks concerns about the precise estimation of node positions given a relatively small portion as anchor nodes with their absolute positions predetermined. Intrinsically, localization is an unconstrained optimization problem based on various distance/path measures. Most of the existing work focus on increasing the accuracy in position estimation typically by using different heuristic-based or mathematical techniques. On the other hand, there were many complex optimization problems successfully tackled by the nature inspired search algorithms including the ant-based or genetic algorithms. In this paper, we propose to adapt an evolutionary approach, namely a micro-genetic algorithm, and integrate as a post-optimizer into some existing localization techniques such as the Ad-hoc Positioning System (APS) to further improve their position estimation. Clearly, our proposed MGA is so adaptable that it can easily be integrated into other localization methods. More importantly, the remarkable improvements obtained by the prototype of our proposed evolutionary optimizer on certain anisotropic topologies of our simulation tests prompt for further investigation.**

I. Introduction

A wireless sensor network is a network consisting of thousands of sensors that span a large geographical region. These sensors are able to communicate with each other to collaboratively detect objects, collect information, and transmit messages. Sensor networks have become an important technology especially for environmental monitoring, military applications, disaster management, etc [5] [9]. However, as sensors are usually small in size, they have many physical limitations. For example, due to its limited size, a sensor does not have a very powerful CPU and is limited in computational power and memory. On the other hand, a sensor is powered by a battery instead of a power outlet. This limitation in energy puts extra constraints in the operations of sensors. As recharging is difficult, sensors should smartly utilize its limited energy in collecting, processing, and transmitting information.

In many applications, sensors have to know their geographical locations. Theoretically, Global Positioning System (GPS) can be used for a sensor to locate itself. In reality, it is not practical to use GPS in every sensor node because a sensor network consists of thousands of nodes and GPS becomes very costly. To solve the problem, many localization methods have been developed. Instead of requiring every node to have GPS installed, all localization methods assume only a few nodes (≥ 3) are equipped with GPS hardware. These nodes are called *anchor* or *beacon* nodes and they know their positions without communicating with other nodes. Other normal sensors then obtain distance information through talking to each other and derive their positions based on the information. A good localization protocol should reduce the error in position estimation by using reasonable number of messages. The computation and memory required should be limited as well due to the nature of sensor networks mentioned above.

Most of the existing work focus on increasing the accuracy in position estimation by using different mathematical techniques such as triangulation [8], multilateration [10], multidimensional scaling [12], [11], [6], convex optimization [3], etc. In these methods, information provided by every anchor node is used, without considering the position of that anchor. To the best of our knowledge, there is only limited research on studying the significance of anchor node selection, particularly for localization algorithms in anisotropic sensor networks [1]. In a previous work [2], we showed that the accuracy of a localization algorithm could be increased by wisely selecting a subset of anchor nodes rather than all anchor nodes when applying the Ad-hoc Positioning System (APS) [8], a distributed and hop-by-hop localization algorithm, to estimate node positions in anisotropic sensor networks [7]. Basically, anisotropic sensor networks possess challenging properties to certain localization algorithms such as the APS due to various limiting factors including the geographical shape of the involved region, different node density, irregular radio patterns, etc. In such challenging anisotropic cases, we identified the criteria of "good" anchor nodes and developed an algorithm for selecting these nodes. Clearly, our previous proposal of selecting "good" anchor nodes can be readily extended to other general sensor networks. After all, localization is intrinsically an unconstrained optimization problems(UOPs) [1] for which a large number of nature inspired algorithms including the ant-based algorithms [14], evolutionary algorithms (EAs) [15] or

[1]A network is isotropic if measurements in all directions are exhibiting the same properties; otherwise, it is anisotropic.

swarm intelligence (SI) [13] have been successfully applied to tackle various instances of complex UOPs. Therefore, in this paper, we consider to further enhance our previously improved APS technique using an adapted micro-genetic algorithm (MGA) [4], [15], with a relatively smaller population size (often ≤ 30) to focus its search effort on limited resources such as the number of iterations allowed, as a post-optimizer. Our adapted MGA for localization consists of two major genetic operators, namely the descend-based mutation and crossover operators, trying to improve on the objective values of individual chromosomes, with each denoting a position estimation of the current node of interest until the best k chromosomes converged to the same solution or the maximum number of iterations exceeded. To demonstrate the feasibility of our proposal, we implemented a prototype of our adapted MGA, and evaluated the performance of the improved APS integrated with our adapted MGA against that of the improved APS using simulations on the challenging anisotropic networks. When compared to the nonlinear optimizer provided in the Matlab Optimization Toolbox, our adapted MGA-based optimizer achieved remarkable improvements on certain anisotropic topologies of our simulation tests, that prompt for more detailed investigation. In addition, our proposal of MGA-based post-optimizer is so generic that it can readily be integrated into other localization methods based on heuristics or mathematical techniques.

This paper is organized as follows. Section II reviews our previous study [2] on the significance of anchor node selection in the APS for localization in anisotropic sensor networks. In Section III, we describe our adapted MGA in detail. Section IV summarizes the experimental results of the improved APS integrated with the adapted MGA against those of the improved APS only. Lastly, we conclude our work in Section V.

II. Preliminaries

In this section, we firstly review the original APS algorithm [8] based on the triangulation technique used in GPS to estimate the positions of other sensor nodes with reference to the fixed positions of anchor nodes dispersed inside any general sensor network. Later, we will consider our previous proposal [2] to improve the position estimation through more careful anchor selection in such challenging cases of anisotropic sensor networks.

A. Ad Hoc Positioning System (APS)

APS [8] is one of the earliest methods developed for localization. It bases on the triangulation used in GPS and is a distributed protocol that requires reasonable computation, memory, and message overhead. APS assumes there are at least three anchor nodes in a sensor network. Each normal sensor tries to find out its distance to the anchor nodes. When the distance information to three or more anchor nodes is obtained, the sensor node can compute its own position using triangulation.

The key question APS answers is how a node finds out its distances to the anchors. Three methods are described: *DV-hop*, *DV-distance*, and *"Euclidean" propagation*. Among them, DV-hop and DV-distance receive most attention. Both DV-hop and DV-distance measure distance in a hop-by-hop manner. Each sensor is required to communicate with its immediate neighbors only. In DV-distance, an anchor node A starts out by sending a broadcast message that contains its identity and its geographical location. By the signal strength of the message, a neighbor of A can then determine its physical distance to A. Each neighbor then broadcasts a message indicating that its distance from A. A node that receives this message will determine its distance to A by adding its distance to its neighbor and the distance of its neighbor to A, which is carried in the message. Subsequently, every node in the network can identify the distance to A. DV-hop works in a similar manner except the physical distance is not derived from signal strength measurement but from *average distance per hop between anchor nodes*.

Theoretically, a normal node requires only distance and position information of three anchor nodes to perform triangulation. When there are more than three anchor nodes, a normal node can use the information of all the nodes to calculate its location or it can select only three. APS does not study this issue in detail. However, intuitively, different choices should yield different estimations of positions. An anchor node which is isolated in a remote area is not a good one since the distance estimated using DV-hop or DV-distance is usually deviate a lot from the real physical distance. We study the effect of selecting different anchor nodes using simulations.

B. The Impact of Anchor Selection in APS

Generally speaking, the selection of anchor nodes should be impactful to the precision of position estimation in most localization algorithms, especially true for the APS or specifically DV-distance algorithm in the presence of measurement error in anisotropic networks. Figure 1 gives a convincing example of

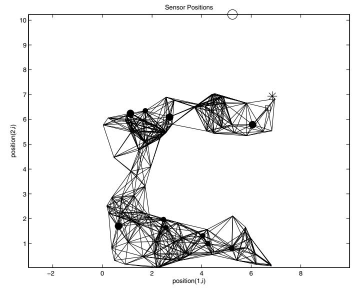

Fig. 1. An example showing the benefit of selecting the best 3 anchors rather than all anchors in APS for a particular node

the estimated positions by both the original and modified APS, that will exhaustively search for the best 3 anchors among all possible combinations of 3 anchors, for a particular sensor node against its true position in a specific case. For clarity of presentation, the square is used to denote the true position of the individual node of interest. The "*" symbol represents the estimated position by our modified APS whereas the circle is the estimated position by the original APS. Obviously, the estimated node position by the original APS is far from its true position for which the estimated position by our modified APS is much closer in this particular case. From our empirical experience, we find that similar cases occur for the 'corner' or its neighboring nodes around the two ends of the C-shape. In corners, the estimation error in the accumulated distance along the various paths from *all anchor nodes* as computed in the original DV-distance algorithm of the APS approach can be much more profound and misleading when compared to that of other positions in the C-shape, thus introducing a much higher imprecision in the estimated position as illustrated in Figure 1.

In view of the empirical results obtained for the modified APS, we proposed in our previous work [2] an *improved APS algorithm*, namely the *APS(Near-3)*, based on a simple yet effective heuristic as follows: the improved APS algorithm will always choose the *nearest 3* anchors for a faster response time with respect to each individual sensor node inside the original APS computation (i.e. the triangulation mechanism) used for its position estimation. To compare their performance, we implemented both the original and improved APS in the Matlab Version 7.0.1 running on a Pentium-IV 3-Ghz PC. Among 20 test cases of anisotropic topologies, the best resulting topology obtained by the improved APS gives $0.1527R$ as the average error whereas the worst result shows $0.5716R$, producing an overall averaged error of $0.3343R$ for all the 20 test cases. On the other hand, the original APS produces $0.7683R$ as the average error for the best case, with the worst result at $1.8731R$, and giving an overall averaged error of $1.0725R$ for all the 20 cases.

III. Our Adapted Micro-Genetic Algorithm

Evolutionary algorithms (EA) [4], [15] are local search methods capable of efficiently solving complex constrained or unconstrained optimization problems (OPT) [1]. As previously discussed, localization in wireless sensor networks is intrinsically an unconstrained OPT. However, in handling relatively sizable (with ≥ 500 variables) or complicated OPTs, the total computational costs can be drastically reduced by focusing the search only on a reasonably small population of chromosomes without much impact on the search efficiency. In fact, it has been reported that MGAs can find solutions with fewer iterations than evolutionary algorithms (EAs) with larger population sizes for some problems [4]. Therefore, in this paper, we consider a micro-genetic algorithm (MGA) based on a small population size (usually < 30) to tackle the challenging localization problems in wireless sensor networks. Basically, a variable in an OPT is usually represented by a gene in a MGA. A chromosome, consisting of all the genes, is used to denote a valuation for all the variables. As inspired by the nature, an EA or MGA basically performs a parallel local search in different parts of the search space through maintaining a population of chromosomes for iterative improvements over the successive generations. Figure 2 shows the pseudo-code

```
MGA($PZ, MZ, fitness()$)
  initialize a small $Population$ of $PZ$ chromosomes
  repeat
    select the best $MZ$ chromosomes $\in Population$
    $Population := \emptyset$
    repeat
      produce {offspring} by descend-based genetic opr.
      $Population := Population \cup \{offspring\}$
    until ($sizeof(Population) = PZ$)
  until ($Population$ is converged or resource limit is
      exceeded)
```

Fig. 2. The Convergence Procedure of our Adapted MGA

of our adapted MGA as a post-optimizer for further position refinement in any localization method. Given the population size PZ, the size MZ of the mating pool and the evaluation function $fitness()$, our adapted MGA initially sets up a small population. In each generation, our adapted MGA selects the best MZ chromosomes according to $fitness()$ from the current $Population$ to construct the mating pool in which some genetic operators such as mutation or crossover are applied to produce offspring to form the next generation. This "selection-and-reproduction" process is repeated until all the chromosomes have converged to the same local minima, or some predetermined resource limit is exceeded. A resource limit is usually defined in terms of CPU time or the maximum number of generations allowed. After all, our adapted MGA for localization aims at minimizing the best value returned by the objective function ($fitness()$) formally defined as follows.

$$fitness(x', y') = \sum_{i=0}^{3} (\sqrt{(x'-x_i)^2 + (y'-y_i)^2} - path\_dist_i)^2$$

where (x', y') represents the current position estimate, (x_i, y_i) is the absolute position of the anchor node i with $i = 0, 1, 2$ for the 3 nearest anchors, and $path\_dist_i$ is the relative path distance measure of anchor node i with respect to the current position estimate. Clearly, the $fitness()$ function gives the mean square error of the current position estimate (x', y') with respect to the positions of the 3 nearest anchor nodes and their relative path distance measures.

The two descend-based genetic operators for our adapted MGA are described as below.

- descend-based mutation operator: for each chromosome in the current population, the operator will invoke the $rand()$ function in C returning a probability p ranging from $0 \ldots 1$. If ($p > 0.5$), the mutation operator will generate a random point (x_r, y_r) in a straight line formed by the current position estimate (x', y') and the previous position estimate (x'_{old}, y'_{old}). In case the newly generated mutate point (x_r, y_r) produces a descend in the objective value returned by $fitness()$ when compared to that of the current estimate (x', y'), (x_r, y_r) will replace the current position estimate (x', y') as the new chromosome; otherwise, the current chromosome will remain unchanged. Essentially, the descend-based mutation operator will look for opportunistic improvement in each position estimate as denoted by each chromosome in the small population of our adapted MGA over successive generations.

- descend-based crossover operator: basically, the whole population of chromosomes are readily sorted in descending order according to their objective values. The crossover operator looks for any opportunistic improvement between any pair of chromosomes sequentially extracted from the sorted list of chromosomes. In this regard, we usually set the PZ to be even. In case not, the last pair can simply be formed from the last and first chromosomes of the sorted list. For each pair $(i, i+1)$ of chromosomes, when the probability value p as returned by $rand()$ is > 0.5, the crossover operator will proceed to consider the mid-point as the crossover point of their position estimates. Similar to the mutation operator, in case there is any decrease in the objective value of this crossover point when compared to that of the worse chromosome $(i+1)$ in the pair, the chromosome $(i+1)$ will be replaced by the new crossover point; otherwise, nothing occurs. The crossover operator will also be applied to the whole population over successive generations of our adapted MGA.

It is worth noting that unlike the conventional EA, our adapted MGA does not employ any selection criterion like the Roulette Wheel mechanism [4] to remove those less favorable chromosomes from the current population. For better efficiency, we simply assume all the chromosomes will remain and be improved whenever possible by our two descend-based genetic operators over successive generations.

IV. Experimental Results

This section considers the empirical evaluation of our proposal of adapted MGA as post-optimizer to further enhance our previously improved APS algorithm [2] using the nearest 3 anchor nodes in anisotropic sensor networks. To demonstrate the effectiveness of our adapted MGA, we generated 5 random instances of C-shape topologies of anisotropic sensor networks to evaluate the performance of both the improved APS and the improved APS integrated with our adapted MGA as the post-optimizer for position refinement. Both the improved APS and the improved APS integrated with our adapted MGA are based on the DV-distance algorithm [8] as described previously. For all test cases, the ratio of anchors is 10% of the total number N of sensor nodes. Besides, following [8], we assume a uniform distribution of 10% measurement errors across the anisotropic sensor network to test on the robustness of the localization algorithms for comparison. For all the subsequent discussion, the performance of each individual localization algorithm is always considered in terms of the average estimation error as measured in R for the involved radio range. The improved APS algorithms are implemented in the Matlab Version 7.0.1 whereas our adapted MGA is implemented in C and compiled using the 'mux compiler provided in the Matlab as external C functions for invocation by the concerned Matlab functions. Their simulation results are obtained on a Pentium IV desktop PC installed with a 3 Ghz processor and 512 Mbytes RAM under the MS Windows XP Operating System. Besides, it should be noted that we arbitrarily set the convergence condition of our adapted MGA as whenever the best 3 chromosomes are converged to the same solution, the whole population is regarded as "converged" for quick experimentation. In addition, the maximum number of generations for each invocation of our adapted MGA to improve on the current position estimate is set to $1,000$. Therefore, from our empirical observation, our adaptd MGA is very efficient for execution (usually < 1 minute in the wall-clock time in total to handle all 100 nodes), even though we have not *yet* explicitly considered the time performance of our adapted MGA in the following paragraphs due to the limited time we have for experimentation.

Table I compares the performance of our originally improved APS and the improved APS integrated with our adapted MGA in terms of the mean position error on different combinations of the total number N of sensor nodes as 100 or 150, and the physical dimension R of the square area for spreading the sensors with $R = 1.2r, 1.6r, 2.0r$ where r denotes the predetermined communication range of each sensor node. Since our adapted MGA is used as an post-optimizer, with the initial, but sometimes possibly fairly gross, position estimate provided by the Matlab nonlinear optimizer, our adapted should always be able to improve on the original position estimates obtained by the improved APS method. To the worst, our adapted MGA may simply return the same position estimate in case it fails to search for a better solution. Therefore, we can consistently see improvements, no matter minor or major, in the mean position error as obtained from our adapted MGA when compared to that of the originally improved APS in Table I. For a better understanding and fairer comparison, we highlight only those significant improvements (i.e. $\geq 10\%$ of improvement) as obtained by our adapted MGA in bold-faced figures. As shown in Table I, among those highlighted cases with $\%$ of improvement $\geq 10\%$, the highest improvement is 46.88% whereas the lowest is 11.48%, giving an overall average of 14.32% for our adapted MGA on all the test cases. For the topolgies with $N = 100$, the averaged $\%$ of improvement is increasing as $5.49\%, 12.03\%, 16.45\%$ respectively for $R = 1.2, 1.6, 2$, giving a group averaged improvement for $N = 100$ as 11.32%. On the other hand, for $N = 150$, the averaged $\%$ of improvement is decreasing as $17.50\%, 17.33\%, 14.84\%$ for $R = 1.2, 1.6, 2$ respectively, giving a group averaged improvement for $N = 100$ as 16.56%. The specific reason(s) for such increasing/decreasing trend of averaged $\%$ of improvement at different settings prompts for further investigation. However, we observed that there is a general increase in the group averaged $\%$ of improvement from 11.32% to 16.56% when the total number N of nodes goes from 100 to 150. This clearly demonstrates the effectiveness of our adapted MGA in further refining the position estimates as provided by the originally improved APS method. After all, whether such general increase in the group averaged $\%$

N = 100	Trial				
R = 1.2	1	2	3	4	5
Improved APS	**0.2074**	0.6554	0.1879	0.2560	0.3970
Adapted MGA	**0.1828**	0.6176	0.1852	0.24382	0.3828
% of impr. by MGA	**11.88%**	5.78%	1.46%	4.75%	3.57%
R = 1.6					
Improved APS	**0.0813**	0.3697	**0.0453**	0.1598	0.2702
Adapted MGA	**0.0582**	0.3488	**0.0366**	0.1534	0.2626
% of impr. by MGA	**28.47%**	5.67%	**19.20%**	4.00%	2.83%
R = 2					
Improved APS	**0.0534**	0.2814	**0.0163**	0.0811	0.1742
Adapted MGA	**0.0320**	0.2736	**0.0108**	0.0799	0.1674
% of impr. by MGA	**40.09%**	2.74%	**34.05%**	1.47%	3.92%
N = 150, R = 1.2					
Improved APS	**0.6421**	0.4682	**0.3715**	**0.2604**	0.4964
Adapted MGA	**0.3550**	0.4574	**0.3198**	**0.2097**	0.4610
% of impr. by MGA	**44.72%**	2.31%	**13.91%**	**19.46%**	7.12%
R = 1.6					
Improved APS	**0.3817**	0.3279	**0.2110**	**0.1047**	0.3281
Adapted MGA	**0.2028**	0.3145	**0.1787**	**0.0927**	0.2989
% of impr. by MGA	**46.88%**	4.07%	**15.30%**	**11.48%**	8.91%
R = 2					
Improved APS	**0.2445**	0.2378	0.1294	**0.0709**	0.2398
Adapted MGA	**0.1419**	0.2293	0.1169	**0.0602**	0.2305
% of impr. by MGA	**41.97%**	3.55%	9.67%	**15.09%**	3.90%

TABLE I

PERFORMANCE OF THE IMPROVE APS AGAINST THAT INTEGRATED WITH OUR MGA ON VARIOUS ANISOTROPIC TOPOLOGIES

of improvement will continue for $N = 200$ or larger, and more importantly the detailed explanation accounted for such increase would again prompt us for more experiments and detailed analysis.

V. CONCLUSION

Wireless sensor networks are widely applicable to many practical applications including environmental monitoring, military applications, disaster management, etc. in which sensors may need to know their geographical locations. Accordingly, various localization algorithms have been developed given a few anchor nodes with their own positions precisely determined. Undoubtedly, most of the existing work focus solely on increasing the accuracy in position estimation by using one single heuristic-based or mathematical techniques. An example is the the Ad-hoc Positioning System (APS) [8] proposed by Niculescu and Nath, that works by the triangulation mechanism used in the Global Positioning System (GPS). Extending from our previous work [2] in which we critically considered the impacts of careful anchor selection in solving the localization problems, we proposed in this paper a generic micro-genetic algorithm (MGA) for integration into various localization algorithms as a post-optimizer to further position refinements whenever possible. Our adapted MGA for localization makes use of two key genetic operators, namely the descend-based mutation and crossover operators, aiming at opportunistic decreases in the objective values obtained for the new mutate out of the current position estimate, or the crossover point for any pair of existing chromosomes over successive generations. To demonstrate the feasibility of our MGA proposal, we implement the improved APS algorithm in the Mathlab Version 7.1 with our adapted MGA as external C functions interfacing to Matlab functions, and obtain their simulation results on a set of 60 randomly generated C-shape topologies. Our simulation results clearly demonstrate the effectiveness of our MGA proposal as the post-optimizer in reducing the average estimation error to almost half (precisely 46.88%) of that attained by the originally improved APS method. More importantly, our adapted MGA, without any prior assumption/knowledge about the underlying localization method, is so generic that it can readily be integrated into various localization algorithms for further position refinement.

This work opens up a number of interesting directions that are worth exploring. First, as pointed out in Section IV, we would conduct more experiments and careful analysis to explain in detail about the general increase in the % of improvement when the number of nodes increases, and more importantly for the specific trend across the various values of R with the same N. Besides, it should be interesting to study about the integration of our generic MGA proposal into other localization algorithms. Lastly, it is worth exploring on fine tunning of the two descend-based genetic operators, possibly integrated with more sophisticated selection criteria, in order to further increase the overall accuracy of position estimation.

ACKNOWLEDGMENT

The authors would like to thank Professor Yi Shang for his fruitful discussion.

REFERENCES

[1] *Nonlinear Programming (2nd Edition)*. Athena Scientific, 1999.

[2] K. Cheng, V. Tam, and K. Lui. Improving aps with anchor selection in anisotropic networks. In *Proceedings of the International Conference on Networking and Services (ICNS'05)*, October 2005.

[3] L. Doherty, K. Pister, and L. Ghaoui. Convex position estimation in wireless sensor networks. In *IEEE Infocom*, 2001.

[4] G. Dozier, J. Bowen, and D. Bahler. Solving small and large scale constraint satisfaction problems using a heuristic-based microgenetic algorithm. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, 1994.

[5] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. In *IEEE Pervasive Computing*, pages 59 – 69, 2002.

[6] X. Ji and H. Zha. Sensor positioning in wireless sensor networks using multidimensional scaling. In *IEEE Infocom*, 2004.

[7] H. Lim and J. C. Hou. Localization for anisotropic sensor networks. In *IEEE Infocom*, 2005.

[8] D. Niculescu and B. Nath. Ad hoc positioning system (aps). In *IEEE Globecom*, pages 2926 – 2931, 2001.

[9] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. In *Communications of the ACM*, volume 43, pages 51 – 58, 2000.

[10] A. Savvides, H. Park, and M. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[11] Y. Shang and W. Ruml. Improved MDS-based localization. In *IEEE Infocom*, 2004.

[12] Y. Shang, W. Ruml, and Y. Zhang. Localization from mere connectivity. In *ACM MobiHoc*, 2003.

[13] SwarmINTEL.com. Swarm intelligence resources. In *website available at: http://www.swarmintelligence.net*, 2004.

[14] V. Tam, S. Koo, and K. Sugiyama. Evolving artificial ant systems to improve layouts of graphical objects. In *Proceedings of the 8th Pacific Rim International Conference on Artificial Intelligence (PRICAI'04)*, pages 955 – 956, 2004.

[15] V. Tam and P. Stuckey. Improving evolutionary algorithms for efficient constraint satisfaction. In *The International Journal on Artificial Intelligence Tools*, volume 28, pages 363 – 383, 1999.