International Journal of Future Generation Communication and Networking
Vol. 8, No. 1 (2015), pp. 191-206
http://dx.doi.org/10.14257/ijfgcn.2015.8.1.20

IGAA: An Efficient Optimization Technique for RFID Network Topology Design in Internet of Things

Po-Jen Chuang and Wei-Ting Tsai

Department of Electrical Engineering
Tamkang University
Tamsui, New Taipei City, Taiwan 25137, R. O. C.
E-mail: pjchuang@ee.tku.edu.tw

Abstract

Most RFID applications in the Internet of Things (IoTs) use multiple readers to read the IDs of multiple tags and form the RFID network. In such a network, unguarded reader deployment may generate over-crowded readers, cause interferences and, as a result, increases the deployment cost while degrading tag detection. Seeing that desirable reader deployment is crucial for RFID system performance, this paper introduces an optimization-based IGAA approach which outperforms existing RFID topology designs by turning up more favorable reader deployment and system performance. The new approach employs an advanced multi-objective fitness function and improved genetic annealing algorithms (GAA) to pursue a better RFID topology design. By involving an improved gene-stirring operation to help preserve good genes and locate optimal solutions for reader deployment, it is simple in operation but effective in practice. Experimental evaluation shows that when compared with related approaches, IGAA can yield better solution quality with less search time.

Keywords: *RFID networks; topology design; optimization-based approaches; Genetic Annealing Algorithms (GAA); experimental evaluation.*

1. Introduction

The radio frequency identification (RFID), a widely used technology in practical applications [1-6], has been considered as one of the principal building blocks for realizing the Internet of Things (IoTs) concept [7-12]. In IoTs, most RFID applications use multiple readers to read the IDs of multiple tags and form the RFID network. For instance, a supermarket or logistics management usually needs multiple readers to read the multiple items in one area. To facilitate the operations, it is necessary to arrange those to-be-deployed RFID readers, including their positions and power levels, by an optimal topology design. Unguarded or unplanned reader deployment (for instance, readers are largely or randomly deployed) may generate over-crowded readers in the network and therefore brings up interferences. When the situation happens, the deployment cost will escalate sharply and the percentage of successful tag detection will tumble down.

As mentioned, interferences tend to happen when an RFID network is over-crowded with readers. There are two such interferences, the reader-to-tag and reader-to-reader interferences [13-15]. In Figure 1, when Tag 3 is read by both readers R1 and R2, the reader-to-tag collision may happen; while in Figure 2, if R1 and R2 are neighbors and Tag 3 can be read by R1, it can cause the reader-to-reader collision.

ISSN: 2233-7857 IJFGCN

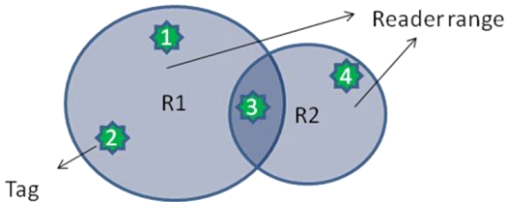


Figure 1. Illustration of the Reader-to-tag Interference

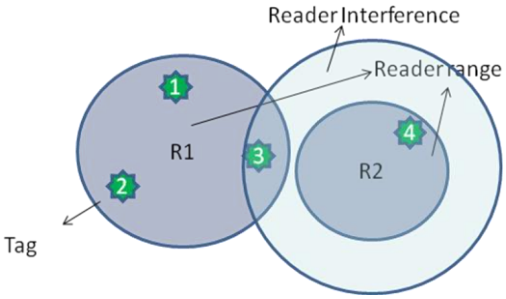


Figure 2. Illustration of the Reader-to-reader Interference

That is to say, when developing an optimal RFID topology design, we must take not only the reader deployment cost and the percentage of tag detection but also the above two interferences into account. Seeing that reader deployment plays a significant role in RFID system performance, we introduce a new RFID topology design in this paper to pursue more desirable reader deployment and system performance. To achieve the goal, we build the new topology design based on (1) the Genetic Annealing Algorithms (GAA) – a previous optimization approach of ours [16], (2) an advanced multi-objective fitness function and (3) the Improved Genetic Annealing Algorithms (IGAA). Simple in design and easy to implement, both GAA and IGAA optimization approaches employ a gene-stirring operation, a novel idea out of the annealing concept [17], to help preserve good genes, locate the optimal solutions for reader deployment, and ultimately upgrade the overall system performance.

Extensive simulation runs are conducted to evaluate and compare the performance – in terms of solution quality and solution search time – of three related optimization approaches: GA [18-22], GAA [16] and IGAA, all using the proposed multi-objective fitness function. Solution quality refers to the optimality of solutions – to be indicated by the fitness values, and solution search time refers to the speed of finding desirable solutions – indicated by the needed complexity. The collected simulation results show that (1) our advanced multi-objective fitness function can reduce the required cost for all of the three approaches, (2) GAA performs better than the original GA approach, and (3) among the approaches, the new IGAA proves to be an optimal choice as it yields the best performance.

2. Background Study

2.1. The Multi-objective Fitness Function

A new optimization-based approach is recently proposed in [18] to give a favorable RFID topology design. Constructed on the genetic algorithms (GA), the approach evaluates each possible solution by a defined linear weighted multi-objective fitness function which covers six

objectives: overlapping of the reading area, the number of useless readers, the number of redundant readers, the number of tags located in the overlapped reading areas, the number of tags covered, and the number of readers located out of the deployment area. Each objective in the fitness function is defined as $f_i = 1/(1+\varepsilon_i^2)$, where ε_i is the difference between the target and current solutions in objective i. To keep ε_i an integer and meanwhile avoid unreasonably amplifying this difference, it is suggested that ε_i^2 be replaced by $|\varepsilon_i|$.

For the approach, as we can see from Figure 3, the values of y depict much bigger differences between $0 < x < 40$ and less differences (i.e., much smoother) around x = 100. To prevent our fitness function from encountering the same problem, we believe it is better to replace 1 in the denominator by 100. Based on the observation and other references [18-20], we redefine each objective in the fitness function to become $f_i = 1/(100+|\varepsilon_i|)$, instead of $f_i = 1/(1+\varepsilon_i^2)$, to remove possible biased effects due to a certain objective, i.e., to be more practical and reasonable. By doing so, we attain the fitness = $\sum w_i * f_i$, where f_i and w_i respectively represent objective i and its weight.

Illustrated in the following are the five objectives in [18]. Note that the last objective – the number of readers located out of the deployment area – is not counted in here because we consider it unreasonable in practice to deploy readers out of the should-be-known deployment area and therefore decide to exclude the objective from our multi-objective fitness function.

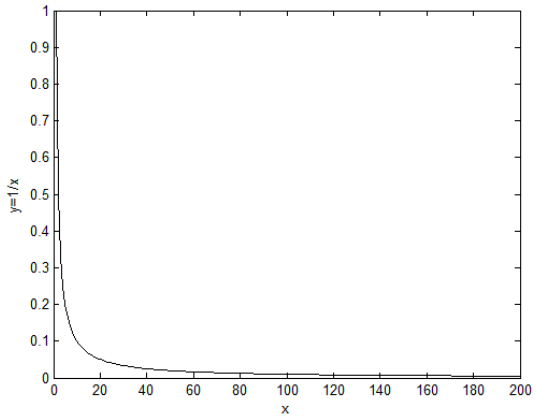


Figure 3. The Curve of Function *y* = 1/*x*

2.1.1. Overlapping of the Reading Area: Dense reader deployment may generate large overlapping of the reading area and consequently brings up the mentioned reader-to-reader and reader-to-tag interferences. Hence, we should limit the overlapping of the reading area upon deployment as Figure 4 shows. When it is practically difficult to keep all readers from overlapping, we can set the limit of the overlapping ratio to be 25%. That is, 25% overlapping of the reading areas is allowed. We then define this objective as

$f_1 = 1/(100+|\varepsilon_1|)$, where the value of ε_1 represents the exceeded overlapping beyond the limit.

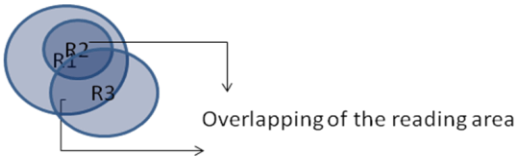


Figure 4. Overlapping of the Reading Area

2.1.2. The Number of Useless Readers: Useless readers will cause extra cost and also the reader-to-reader interference, as Figure 5 demonstrates. The objective is defined as

$f_2 = 1/(100+|\varepsilon_2|)$, in which ε_2 represents the number of useless readers.

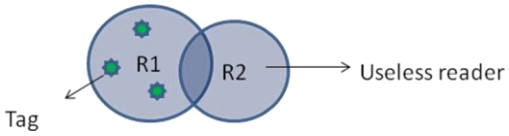


Figure 5. Useless Readers

2.1.3. The Number of Redundant Readers: Redundant readers also lead to extra cost and both the reader-to-reader and reader-to-tag interferences, as Figure 6 displays. This objective is defined as

$f_3 = 1/(100+|\varepsilon_3|)$, where ε_3 is the number of redundant readers.

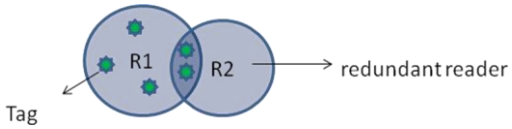


Figure 6. Redundant Readers

2.1.4. The Number of Tags Located in Overlapping Reading Areas: To avoid locating tags in the overlapping reading area is either difficult or costly (takes substantial deployment cost). But if too many tags are located in the overlapping area, it will trigger the reader-to-tag interference. Therefore we need to limit the number of such tags upon deployment. In Figure 7, we see that the number of tags allowed in the overlapping reading area between any two readers is set to be 2. The objective is defined as

$f_4 = 1/(100+|\varepsilon_4|)$, ε_4 indicating the number of tags located in overlapping reading areas beyond the limit.

Figure 7. Tags Located in the Overlapped Area

2.1.5. The Number of Tags Covered: Assume that a tag can be successfully identified if it is covered by a reading area. Given any number of readers in the deployment area, it will be desirable to cover as many tags as possible, as shown in Figure 8. This objective is defined as

$f_5 = 1/(100+|\varepsilon_5|)$, ε_5 referring to the number of uncovered tags which should be minimized.

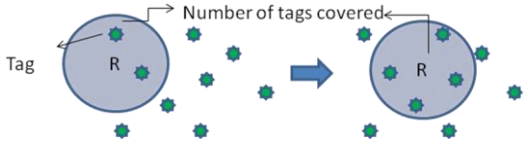


Figure 8. The Number of Tags Covered

2.2. The Optimization Algorithms

Optimization algorithms can be employed to solve quite a number of problems, including task matching and scheduling, reader network planning and so on. The following is the brief introduction of the two optimization algorithms involved in this investigation.

2.2.1. The Genetic Algorithms (GA) [23]: GA has three major operations: selection, crossover and mutation. It uses the selection process, which covers selection and copying, to select better fitness values into the next generation (common mechanisms include roulette wheel and tournament), the crossover process to mix the species in order to produce a better next generation (common mechanisms include a single point, pairs of points and even crossover), and the mutation process to avoid the missing of excellent species.

GA operates in the following steps:
Step1: Initialize the population size
Step2: Calculate the fitness value of each species according to the objective function
Step3: Go through the selection process according to the fitness values
Step4: Go through the crossover process to produce the next generation
Step5: Take the mutation process to avoid local optima
Repeat steps 2 to 5 until convergence or reaching a stop condition. Figure 9 gives the flowchart of GA.

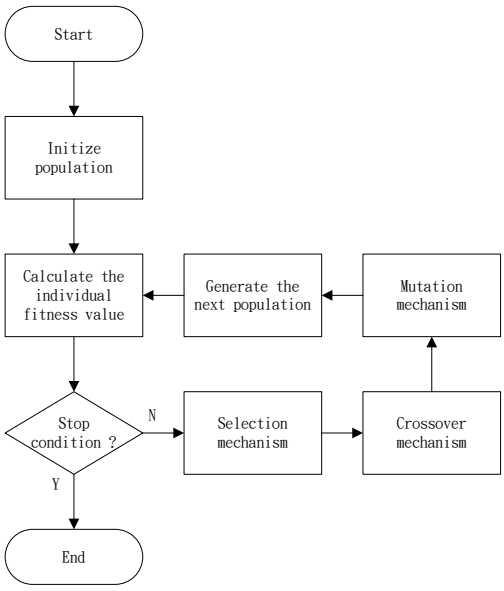


Figure 9. The Flowchart of GA

2.2.2. The Genetic Annealing Algorithms (GAA) [16]: A previous optimization approach of ours, GAA is the hybrid practice of GA, SA (simulated annealing) [17] and GESA (guided evolutionary simulated annealing) [24]. It involves only one operation, the stir operation, and four parameters: the number of genes to be stirred (N), the decreased number of genes to be

stirred (n), the number of stirs (M), and the decreased number of stirs (m). G = N/n = the number of generations. M decides the frequency of stirs while m decides the decreasing stir frequency, in the stir operation.

Denoted by Stir(X, N), where X represents any solution and N is the number of genes to be stirred, the stir operation works as follows.

(1) Randomly select N genes (characters) from an X.

(2) Change the values of the selected genes or their positions in the chromosomes.

(3) Completely stir the genes in the chromosomes to bring up all possible solutions scattering over the search space.

GAA operates in the following steps:

Step1: Set the four parameters N, n, M and m

Step2: Randomly generate a solution X

Step3: Generate a new solution Y after executing the stir operation for X

Step4: Select Y if F(Y) is better than F(X) or select X if otherwise

Repeat steps 3 and 4 M times. Then, decrease N by n and repeat steps 3 and 4 M = M-m times, until N becomes 0. Figure 10 shows the flowchart of GAA.

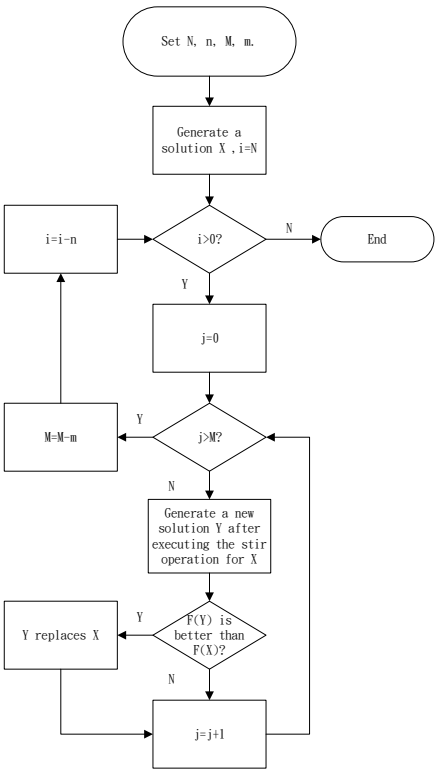


Figure 10. The Flowchart of GAA

2.2.3. GA and GAA: Table 1 lists the features of GA and GAA for easier contrast. As GAA has a limited number of iterations, we need to adjust the number of genes to be stirred (N) so that the generations can outnumber the genes. For example, If N = 30 and the decreased number of genes to be stirred n = 1, GAA will produce at most 30 generations. To produce more than 30 generations, it needs to adjust N.

Table 1. Characteristic Comparison between GA and GAA

	GA	GAA
Replacement of genes	Complex	Simple
Probability to keep genes	Fixed	Annealing
Probability not to fall into local optima	Lower	Higher
Number of iterations	Unlimited	Limited

We adjust the number of genes to be stirred, use the above mentioned multi-objective fitness function to simulate GA and GAA, and run up to 200 iterations. Figure 11 gives the fitness values obtained in each iteration. As we can see, in earlier iterations, GAA does not yield much better search capability than GA because it has more genes to be stirred. But, in later iterations, it is able to generate more favorable fitness values due to its special features.

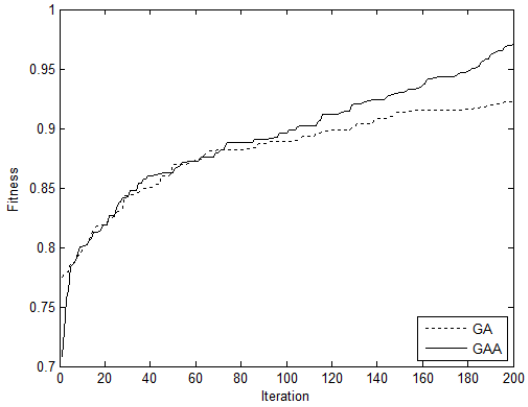


Figure 11. Fitness values for GA and GAA in each Iteration

3. The Proposed Approaches

3.1. The Advanced Multi-objective Fitness Function

As the cost objective function in existing literature [19] considers only the total price (related to the total power) of deploying readers, it may lead to biased reader deployment in multi-objective optimization. More specifically, it may lead to biased consideration and hence two biased deployment results: (1) covered tags will decrease extensively – because the cost issue prevails and (2) remaining readers become useless – because more readers with high power have been deployed (e.g., intending to deploy 10 readers but end up deploying only 4 with the maximum power). To avoid such biased effects, we thus define a more proper cost objective $f_6 = 1/(100+|\varepsilon_6|)$, where

$$\varepsilon 6 = \sum_{i=1}^{n} \frac{\text{the coverage area of reader } i}{\text{the number of tags covered by reader } i}$$

while n is the number of deployed readers. The value of ε_6 indicates the average reader coverage area per tag, related to the density of covered tags (the smaller ε_6 is, the higher the density). Minimizing ε_6 can reduce the cost and increase the covered tags, as the 3 cases in Figures 12-14 (resulting from our cost objective f_6) illustrate. In Case 1, for a deployed reader R1, reducing its coverage area (from left R1 to right R1 in Figure 12) to cover the same number of tags can generate better deployment. By better deployment, we indicate R1 needs a smaller coverage area, lower power and reduced cost. That is, minimizing ε_6 can reduce the average coverage area and power consumption per reader, and as a result bring down the overall deployment cost.

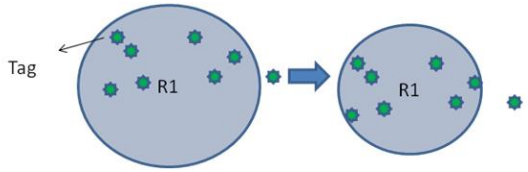


Figure 12. Case 1: achieved by cost objective f_6

To achieve our cost objective f_6, we may change the deployment result from deploying one reader to two readers. As Case 2 in Figure 13 shows, the reduced coverage area from left R1 to right R1 and R2 can still cover the same number of tags. Assuming the cost of two low-power readers is lower than that of a high-power reader – which is usually true in practice, such a deployment change can help reduce the overall deployment cost.

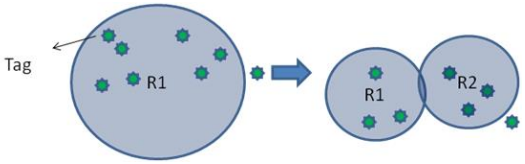


Figure 13. Case 2: Achieved by Cost Objective f_6

Figure 14 shows that achieving our cost objective f_6 may bring a deployed reader, such as R1 here, to cover more tags. This may not cut down the cost of the deployed reader but we can still expect the overall deployment cost down because of the obtained more covered tags.

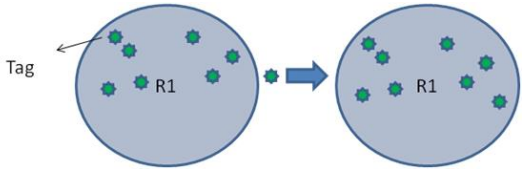


Figure 14. Case 3: Achieved by Cost Objective f_6

3.2. The Proposed Improved Genetic Annealing Algorithm (IGAA)

The performance differences of GA and GAA reveal two problems with GAA: (1) generating limited iterations and (2) yielding less search capability in the middle of iterations. To increase iterations, we can take a proper decimal number less than 1 as the decreased number of genes to be stirred n (1>n>0) to extend the number of iterations. For example, with the initial number of stir genes N = 10 and n = 1, GAA will have at most 10 iterations, but if we set n = 1/20 = 0.05, we can extend the number of iterations from 10 to 200. As N is an integer, it will decrease by 1 after each 20 iterations. We use the notation (10,...,10), (9,..,9), …, (2,..,2), (1,..,1) to illustrate the value changes of N, where each parenthesis indicates the values of N for 20 iterations.

The two problems with GAA are actually related. That is, the solution for the first problem may worsen the second problem because when we take smaller n to produce more iterations, the stir operations will generate large-scale mutations and unexpectedly degrade the search capability. To improve the situation, we decrease n by a new approach: moving from *constant* decrement to *cyclic* decrement, to narrow down possible large-scale mutations. The new approach is called Improved GAA or IGAA because it removes the disadvantages of GAA. To help illustrate IGAA, Figure 15 recaps the operations of Stirs and the four parameters (N, n, M, m) in GAA [16].

```
for( i=N ; i>0 ; i-=n ){
        for( j=0 ; j<M ; j++ ){
                Y = stir(X,i);
                if( F(X) > F(Y) ) Y=X;
        }
        M-=m;
}
```

Figure 15. The Operations of GAA

Figure 16 gives the operations of IGAA. As the figure shows, IGAA can arbitrarily set iter to be any number of iterations, in contrast to GAA which needs to set n between 0 and 1 to increase the number of iterations. By changing the values of N, it also avoids unnecessary large-scale mutations in GAA. To see how IGAA works, we use k (Figure 16) instead of i (Figure 15) to indicate N in each iteration, and calculate k by function transform(i, iter) (Figure 17). To give an example, assuming iter = 200, N = 10, n = 1 and i is initialized to be iter = 200, the value changes of k for 200 iterations in IGAA will be (10,9,8,…,3,2,1,10,9,…), (9,8,7,…,3,2,1,9,8,…), …, (2,1,2,1,…), (1,1,…) – in contrast to (10,...,10), (9,..,9), …, (2,..,2), (1,..,1) in GAA with 1>n>0. It is clear that IGAA moves from *constant* decrement to *cyclic* decrement for each 20 iterations, to narrow down possible large-scale mutations. Figure 18 shows the flowchart of IGAA.

```
for( i=iter ; i>0 ; i-=n ){
        k=transform(i,iter);
        for( j=0 ; j<M ; j++ ){
                Y = stir(X,k);
                if( F(X) > F(Y) ) Y=X;
        }
        M-=m;
}
```

Figure 16. The Operations of IGAA

```
int transform(i,iter){
        temp=(i-1)/(iter/N)+1;
        k=(i-1)%temp+1;
        return k;
}
```

Figure 17. Function Transform()

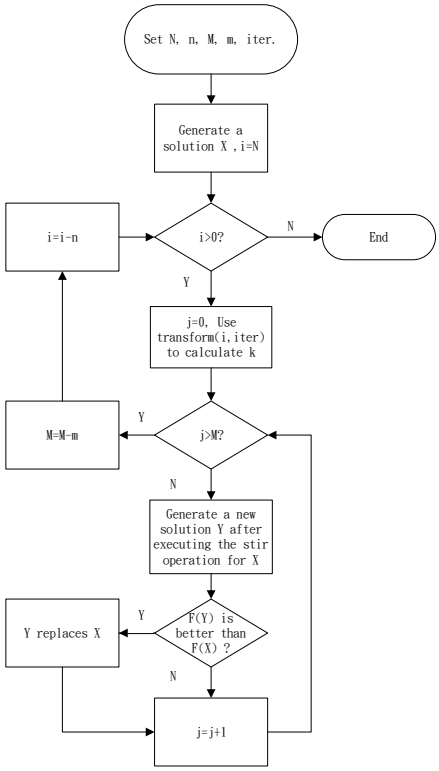


Figure 18. The Flowchart of IGAA

4. Experimental Evaluation

We have carried out extensive simulation runs to evaluate the performance of related optimization-based RFID network topology designs, including GA, GAA and IGAA. Table 2 lists the involved simulation parameters [18].

Table 2. The Simulation Parameters

Readers	10	**w1**	0.2
Reader area	32m*32m	**w2**	0.2
Tags	30	**w3**	0.2
Tag area	20m*20m	**w4**	0.1
Overlapping ratio limit	0.25	**w5**	0.2
Tag limit in overlapping	2	**w6**	0.1

We first calculate the cost of GA and GAA before and after using our new cost objective to check its effect. The stop criterion is set at 200 iterations, and the results (i.e., the cost differences of the two approaches) are plotted in Figure 19. We define the cost as the total power of the deployed readers, and find that the new cost objective helps both GA and GAA drop the cost notably, each from 44.9 to 35.5 and 49.8 to 38.6.

Based on the results, we henceforth simulate and compare the three target approaches under the practice of the advanced fitness function and new cost objective.

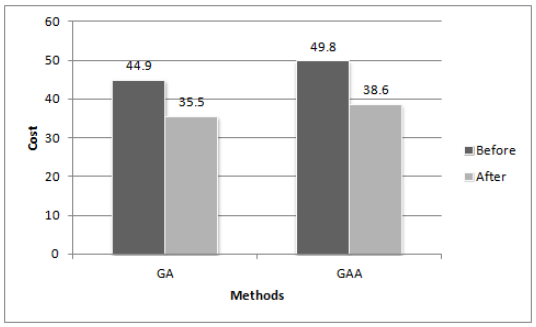


Figure 19. Cost Performance before and After using Our New Cost Objective

Figure 20 gives the fitness values obtained in each iteration for GA, GAA and IGAA. Among the approaches, IGAA yields the best fitness value in each iteration, thanks to its improved reader deployment. The performance of GAA and GA stays close for the first 80 iterations and then starts to differ, with the advantage going to GAA.

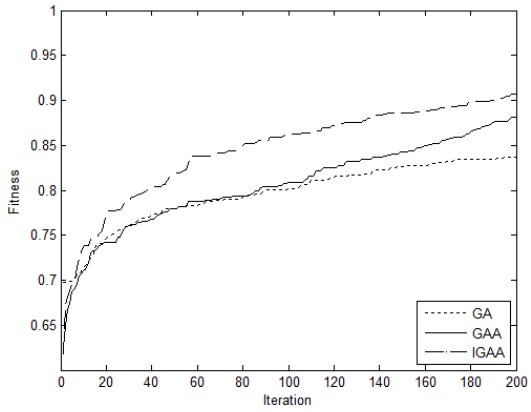


Figure 20. Fitness Values Obtained in each Iteration for the Approaches

Figure 21 exhibits the fitness values obtained upon convergence. Our simulation assumes convergence happens when the fitness value deference between two back-to-back iterations drops below a preset small value continuously for 20 iterations. Recall that in our objective function $f_i = 1/(100+|\varepsilon_i|)$, ε_i is the difference between the target and current solutions for objective i and will become 0 when reaching objective i. As the smallest difference of ε_i/f_i between two back-to-back iterations is 1/(1/100-1/101), we thus preset the small value = (1/100-1/101). In Figure 21, we see that GAA generates slightly smaller fitness values than GA – the consequence of the problems mentioned in Section 3.2. By contrast, IGAA which improves on the problems generates remarkably better fitness values upon convergence (i.e., better solution quality) than GAA and GA.

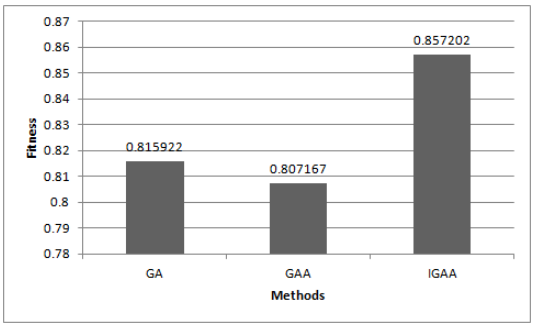


Figure 21. Fitness Values Obtained upon Convergence

Figure 22 displays complexity. We define complexity as the average processing time per iteration, to indicate the solution search time or the simplicity of operations. The results show that GAA and IGAA take much less complexity than GA because they both involve only the simple stir operation in each iteration. We also notice that IGAA, which obtains N from function transform(), takes higher complexity than GAA.

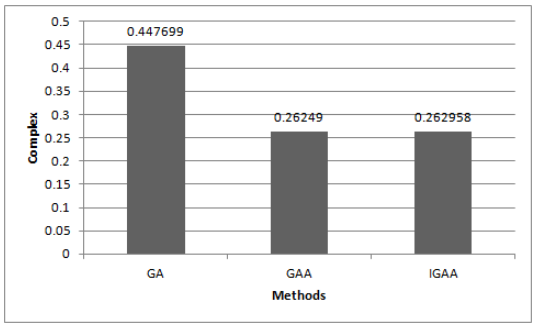


Figure 22. Complexity for the Approaches

Figure 23 gives the number of iterations upon convergence for the approaches, to indicate their ability to jump out of local optima or the probability not to fall into local optima. Note that we assume convergence happens when the fitness value deference between two back-to-back iterations drops below a preset value (1/100-1/101) continuously for 20 iterations. Therefore, when an approach has higher ability to jump out of local optima, it may widen the difference of fitness values between iterations and result in later convergence (i.e., reaching convergence with increased iterations). That is to say, the number of iterations upon convergence is not a proper indication for search capability. As Figure 21 exhibits, IGAA generates the biggest number of iterations but also significantly the best fitness value upon convergence among the approaches. Compared with GA, GAA takes a smaller number of iterations and also smaller fitness value upon convergence due to its limited number of iterations. The key point is, if the three approaches are to reach the same fitness value, we believe our IGAA will take the least number of iterations.

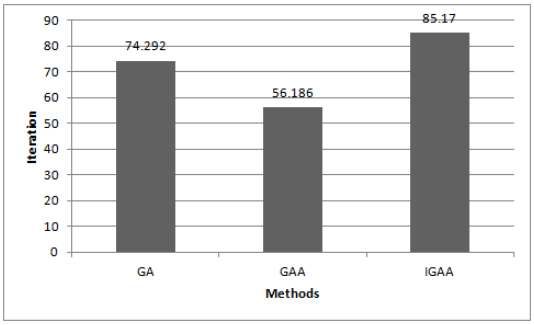


Figure 23. The Number of Iterations Upon Convergence

5. Conclusions

Observing that reader deployment plays a significant role in RFID system performance, this investigation introduces a new RFID topology design to pursue more desirable reader deployment and enhanced system performance. The new approach is built on genetic annealing algorithms (GAA) but has improved over existing design loopholes to ensure better practice. To achieve the goal, we develop an advanced multi-objective fitness function and the improved genetic annealing algorithms (IGAA). The IGAA approach is simple in design and easy to implement. It employs an improved gene-stirring operation to help preserve good genes, locate the optimal solutions for reader deployment and, as a result, uplift the overall system performance. As the obtained simulation results have demonstrated, our advanced multi-objective fitness function can reduce the required cost for all of the target approaches, and the proposed IGAA approach outperforms the other target approaches in both solution quality (better fitness values) and solution search time (less complexity).

Acknowledgement

This paper is a revised and expanded version of a paper entitled ‘On RFID Network Topology Design for Internet of Things’ presented at FGCN 2014on December 20-23, 2014 at Hainan China.

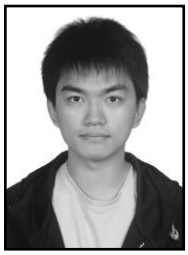
References

[1] L. Sanchez and V. Ramos, “Towards a New Paradigm for RFID Identification: Should We Cluster RFID Tags Or Not?”, Proceedings of the IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications, **(2013)**, pp. 444-451.
[2] S. Ma, Y. Zhang, and D. Wang, “Distributed Work Flow System for RFID Integration and Application”, Proceedings of 2009 IEEE Youth Conference on Information, Computing and Telecommunication, **(2009)**, pp. 62-65.
[3] F. Hussian, S. Durrani, J. Farooqi, H. Mahmood, G. Junjua, I. Jattala, and N. Ikram, “RFID & WSN Based Integrated Maternity Ward Monitoring System”, Proceedings of the 16th International Multi Topic Conference, **(2013)**, pp. 43-48.
[4] W. Chun, E. Noel and K..W. Tang, “The Tag Duplication Problem in An Integrated WSN for RFID-Based Item-level Inventory Monitoring”, Proceedings of the 5th International Conference on Networked Sensing Systems, **(2008)**, pp. 59-62.
[5] L. Wang, L. D. Xu, Z. Bi, and Y. Xu, “Data Cleaning for RFID and WSN Integration”, IEEE Transactions on Industrial Informatics, vol. 10, no. 1, **(2014)**, pp. 408-418**.**
[6] T. H. Oh, Y. B. Choi, and R. Chouta, “Supply Chain Management for Generic and Military Applications using RFID”, International Journal of Future Generation Communication and Networking, vol. 5, no. 1, **(2012)**, pp. 61-76.

[7] L. D. Xu, W. He, and S. Li, "Internet of Things in Industries: A Survey", IEEE Transactions on Industrial Informatics, vol. 10, no. 4, **(2014)**, pp. 2233-2243.
[8] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Integration of Agent-based and Cloud Computing for the Smart Objects-oriented IoT", Proceedings of the IEEE 18th International Conference on Computer Supported Cooperative Work in Design, **(2014)**, pp. 493-498.
[9] I. Ganchev, Zhanlin Ji, and M. O'Droma, "A Generic IoT Architecture for Smart Cities", Proceedings of the 25th IET Irish Signals & Systems Conference and China-Ireland International Conference on Information and Communications Technologies, **(2014)**, pp. 196-199.
[10] T. Yashiro, S. Kobayashi, N. Koshizuka, and K. Sakamura, "An Internet of Things (IoT) Architecture for Embedded Appliances", Proceedings of the 2013 IEEE Region 10 Humanitarian Technology Conference, **(2013)**, pp. 314-319.
[11] W. Hu and T. Cheng, "Simulative Research on the Function of Internet of Things Basing on the Changing of Topological Structure", International Journal of Future Generation Communication and Networking, vol. 6, no. 2, **(2013)**, pp.169-178.
[12] P.-J. Chuang and W.-T. Tsai, "RFID Network Topology Design for Internet of Things", Proceeding of the 8th International Conference on Future Generation Communication and Networking, **(2014)**.
[13] Y. Kang, M. Kim, and H. Lee, "A Hierarchical Structure Based Reader Anti-collision Protocol for Dense RFID Reader Networks", Proceedings of the 13th International Conference on Advanced Communication Technology, **(2011)**, pp. 164-167.
[14] C.-Y. Chiu, C.-H. Ke, and K.Y. Chen, "Optimal RFID Networks Scheduling Using Genetic Algorithm and Swarm Intelligence", Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, **(2009)**.
[15] X. Zeng and J. Tian, "Solution to Both Frequency Interference and Tag Interference of RFID System", Proceedings of the 3rd International Conference on Intelligent Control and Information Processing, **(2012)**.
[16] P.-J. Chuang, C.-H. Wei, and Y.-S. Chiu, "GAA: A New Optimization Technique for Task Matching and Scheduling in HCSs", Journal of Information Science and Engineering, vol. 21, no. 2, **(2005)**, pp. 287-308.
[17] Z. P. Lo and B. Bavarian, "Job Scheduling on Parallel Machines Using Simulated Annealing", Proceedings of the 1991 IEEE International Conference on Systems, Man, and Cybernetics, vol. 1, **(1991)**.
[18] O. Botero and H. Chaouchi, "RFID Network Topology Design Based on Genetic Algorithms", Proceedings of the 2011 IEEE International Conference on RFID-Technologies and Applications, **(2011)**.
[19] Y. Gao, X. Hu, L. Peng, H. Liu, and F. Li, "A Differential Evolution Algorithm Combined with Cloud Model for RFID Reader Deployment", Proceedings of the 4th International Workshop on Advanced Computational Intelligence, **(2011)**.
[20] M. Ma, P. Wang, and C,-H, Chu, "A Novel Distributed Algorithm for Redundant Reader Elimination in RFID Networks. Proceedings of the 2013 IEEE International Conference on RFID-Technologies and Applications, **(2013)**.
[21] X.-M. Zhang, Q.-K. Yan, and J. Li, "The Planning of Workshop RFID Network Based on Modified Genetic Algorithm Using Metropolis Rule", Proceedings of the IEEE 18Th International Conference on Industrial Engineering and Engineering Management, **(2011)**.
[22] H.-L. Ding, W. W. Y. Ng, P. P. K. Chan, D. S. Yeung, and D.-L. Wu, "RFID Reader Deployment Using RBFNN with L-GEM", Proceedings of the 2010 International Conference on Machine Learning and Cybernetics **(2010)**.
[23] M. Mitchell, "An Introduction to Genetic Algorithms", MIT Press, Cambridge.
[24] C. Y. Shen, Y. H. Pao, and P. Yip, "Scheduling Multiple Job Problems with Guided Evolutionary Simulated Annealing Approach", Proceedings of the 1st IEEE Conference on Evolutionary Computation, **(1994)**.

Authors

Po-Jen Chuang received the B.S. degree from National Chiao Tung University, Taiwan, R.O.C., in 1978, the M.S. degree in Computer Science from the University of Missouri at Columbia, U.S.A., in 1988, and the Ph.D. degree in Computer Science from the Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, U.S.A. (now the University of Louisiana at Lafayette), in 1992. Since 1992, he has been with the Department of Electrical Engineering, Tamkang University, Taiwan, where he is currently a Professor. He was the department chairman from 1996 to 2000. His main areas of interest include parallel and distributed processing, fault-tolerant computing, computer architecture, mobile computing, network security and cloud computing.



Wei-Ting Tsai received his B.S. degree in Electrical Engineering in 2013 from Tamkang University, Taiwan, where he is currently studying toward his M.S. degree. His research interests include parallel and distributed processing, internet of things and mobile computing.