This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2007 proceedings.

A Simulated Annealing Algorithm for RFID Reader Networks

Chun-Fu Lin
Department of Information Management
National Taiwan University (NTU)
Taipei, Taiwan, R.O.C

Frank Yeong-Sung Lin
Department of Information Management
National Taiwan University (NTU)
Taipei, Taiwan, R.O.C

Abstract—**In consideration of the reader collision problem in Radio Frequency Identification (RFID) networks, we present an algorithm that deploys RFID reader networks so that they are flexible and efficient. We formulate the deployment issue as 0/1 integer programming problem, and propose a simulated annealing-based heuristic algorithm to solve it.**

Keywords Reader Collision Problems, RFID Reader Networks, RFID, Simulated Annealing Approach

I. INTRODUCTION

An RFID reader reads/writes data from/to RFID tags in its interrogation zone. Since RFID readers and tags operate on the same frequency channel, and any tags that respond to the reader at the same time will collide with each other. Several reader-to-tag communication protocols have been proposed and adopted as anti-collision standards. For example, the Tree Walking Algorithm (TWA) and Slotted Aloha are used by EPC class 0 and ISO 18000 mode 3 respectively.

Besides the tag collision problem, Engels et al. [4] observed that there also exists a reader collision problem. Take the applications of SRFID (RFID tags with sensor capability), for example. When the surveillance area is larger than a reader's effective range, to ensure complete coverage, it is necessary to deploy more readers. However, some readers will collide with each other due to the inevitable overlap of their interrogation zones.

Waldrop et al. [9] proposed a distributed algorithm called "Color-wave" to address the reader collision problem. It allows each reader to decide its own "colors" (time slots) autonomously, based on the "kick" message exchanged between readers. If adjacent readers are not close enough for communication, the "kick" messages are suspended. Another distributed algorithm called "Reader Collision Avoidance (RCA)" described by Carbunar et al. [1], can operate without direct communication, but the time complexity depends on the total numbers of tags and readers. If these two numbers are large, the time complexity of RCA increases accordingly.

In contrast to the above approaches, we propose a centralized algorithm. There are two reasons for this. First, an adversary with knowledge of a distributed algorithm could join the corresponding network without authorization. Second, in consideration of the fairness, a reader with more tags in its zone should be allocated more time slots than others in one cycle. In EPC networks [8], an RFID reader is regarded as a front-end device between back-ends and tags (data sources). Data received by a reader is further integrated and processed by the back-ends for different purposes. The reader can connect to a back-end by a direct link, wireless, or ad-hoc network. In a non-dynamic environment, a centralized method is considered more efficient than distributed ones. In our scenario, back-end servers are involved in service processes and network security, as well resolving performance issues. One back-end is dedicated to calculating and broadcasting a new reader's access time schedule when the network topology changes, or when a reader's slot requests are modified.

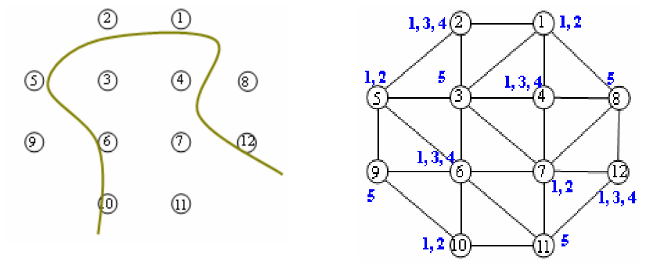
II. PROBLEM DESCRIPTION

The reader collision problem can be described as an N-coloring problem that is NP-complete, similar to the channel assignment problem in cellular networks. The mathematical model in [5] can be extended to solve this problem. The objective of our model is twofold: 1) to minimize the total number of assigned slots (cycle lengths) of all readers; and 2) to satisfy every reader's slot requests.

To improve flexibility, a server can forwardly alter a reader's slot requests according to the actual situation, such as the decreasing (increasing) number of tags or the percentage of the stationary tags in the reader's zone. Note that every recalculation and announcement of a new schedule raises the overheads of the network.

Our objective is to allocate additional slots to particular readers and overlapping the slots without increasing the cycle length. In some applications, such as factories or container yards, objects are usually stored in one spot or moved to another spot according to some reckoning patterns. Instead of reducing the number of requests sent to one reader by increasing the number sent to another reader, additional slots are pre-assigned. Overlapping slots can be used in alternate sequence of readers, or be activated (deactivated) by the server by sending mask bits. Figure 1 (a) illustrates a simple route passing through or close to readers 10, 5, 2, 1, 4, 7, and 12. If the server decides to allocate one additional slot to each reader on the route, the possible time schedule could be calculated as shown in Figure 1 (b). In this case, the server does not need to recalculate and broadcast a new schedule to all readers every time goods are moved along the route.

We formulate the above-mentioned problem as 0/1 integer programming problem, and propose a general mathematical framework to model it.



(a) A route map in an irregular network, reproduced from [5].

(b) One possible slot assignment for the route in (a). The slot requests vectors are (2, 3, 1, 3, 2, 3, 2, 1, 1, 2, 1, 3)

Figure 1.

III. PROBLEM FORMULATION

The following notations are used in the proposed model.

Given parameters:

- L: Set of readers.
- F: Set of time slots.
- c_j: Number of slot requests of reader j.
- α_j: Ratio of overlapped slots to the total number of slots assigned to j.
- A_{jk}: The adjacency matrix of readers. If the interrogation zones of reader j and reader k overlap, A_{jk} will be set to 1; otherwise 0.

Decision variables:

- B_{ij}: The overlapped slot matrix. If slot i assigned to reader j is also assigned to any of j's neighbors, B_{ij} is set to 1; otherwise 0.
- x_{ij}: If slot i is assigned to reader j, x_{ij} is set to 1; otherwise 0.
- y_i: The slot assignment array. If slot i is assigned to any reader, y_i is set to 1; otherwise 0.

Problem (IP)

$$Z_{IP} = \min \sum_{i\in F} y_i \qquad \text{(IP)}$$

Subject to:

$$\sum_{i\in F} B_{ij} \le \alpha_j \sum_{i\in F} x_{ij} \qquad \forall j \in L \quad (1)$$

$$(x_{ij} + x_{ik}) A_{jk} \le B_{ij} + A_{jk} \qquad \forall i \in F, \forall j \in L, \forall k \in L \quad (2)$$

$$c_j \le \sum_{i\in F} x_{ij} \qquad \forall j \in L \quad (3)$$

$$x_{ij} \le y_i \qquad \forall i \in F, \forall j \in L \quad (4)$$

$$y_i \le \sum_{j\in L} x_{ij} \qquad \forall j \in L \quad (5)$$

$$x_{ij} = 0 \text{ or } 1 \qquad \forall i \in F, \forall j \in L \quad (6)$$

$$B_{ij} = 0 \text{ or } 1 \qquad \forall i \in F, \forall j \in L \quad (7)$$

$$y_i = 0 \text{ or } 1 \qquad \forall i \in F \quad (8)$$

The physical meanings of the constraints are as follows: Constraint (1) ensures that the number of overlapped slots of reader j is not larger than j's threshold α_j. Note that if all α_j=0, this problem is exactly an N-coloring problem. Constraint (2) requires that slot i, which is assigned to reader j, is also assigned to any neighbor of j; then B_{ij} is set to 1. Constraint (3) requires that the slot assignment must satisfy the requests of every reader. Constraints (4) and (5) require that y_i must be set to 1 when slot i is assigned to any reader j. Constraints (6), (7) and (8) are integer constraints for the decision variables x_{ij}, B_{ij}, and y_i respectively.

IV. ALGORITHMS

The simulated annealing (SA) [7][2] algorithm is a general method for solving difficult combinatorial optimization problems. Dunque-Anton et al. [3] used SA to solve the channel assignment problem in a cellular network gracefully. We apply the concept of SA in our algorithm to approximately solve the same problem in RFID networks.

The SA algorithm, detailed in Table 1, applies a stochastic search process after the initial configuration. If the latter is close to the optimal value, there is a high probability that SA will find the optimal solution.

We use another heuristic algorithm, S, shown in Table 2, to derive an initial configuration (lower bound) for our problem. To reduce the number of comparisons, Algorithm S computes the slot arrangement according to each reader's id in ascending order. Initially, the slots do not overlap, and the algorithm tries to move each reader's assigned slots as far forward as possible without violating the overlapping constraints.

A random sequence in each loop is generated at the beginning of the SA cooling process. The first reader of the sequence selects one of its assigned slots and moves it to any available slot at random. This forces the other readers in the sequence to exam their current assignments. An affected slot will be either moved forward or backward in accordance with the constraints. This process creates a new configuration. The SA algorithm then calculates the energy of the new configuration and compares it with the previous one. To avoid getting trapped in a local minimum, a new configuration with more energy might be accepted by applying the Metropolis criterion.

After r iterations, the configuration with the minimum energy is found and saved as the best solution. When the frozen temperature t_f, is reached, the algorithm stops.

The energy, E, is defined as follows:

$$E = \sum_{i\in F} y_i$$

TABLE I. THE PSEUDO CODE FOR SA

```
1   Initialization.
    1.1 Use algorithm S to find an feasible slot arrangement. Calculate the deployment energy: $E_{new} \leftarrow$ current energy E.
    1.2 Save the current configuration as the initial solution. $E_{min} \leftarrow E_{old} \leftarrow E_{new}$.
2   Repeat steps (3) - (11) until $t <= t_f$.
3   Repeat steps (4) – (10) r times.
4   Generate p and a reader sequence set $S = (s_1, s_2 \ldots s_n)$ randomly, where $0<p<1$, $n=|L|$.
5   Set $j=s_1$. Select one assigned slot $x_{ij}$, $i \in F$ and $x_{ij}=1$ randomly. Move $x_{ij}$ to an available random slot $x_{kj}$, $k \in F$ and $x_{kj}=0$.
6   Adjust for a new deployment.
    6.1 For $j=s_m$, $m=2$, repeat steps (6.2) until $m=n$.
    6.2 For each assigned slot $x_{ij}$, $i \in F$ and $x_{ij}=1$, try to move $x_{ij}$ to a new available slot $x_{kj}$, $k \in F$ and $x_{kj}=0$ without violating Constraints (1)&(2). If k is found, then move $x_{ij}$ to $x_{kj}$.
7   Calculate new deployment energy $E_{new}$ and energy difference $\Delta E \leftarrow E_{new}-E_{old}$
8   If $E_{new}<E_{old}$ or $\mathbf{exp}(-\Delta E/t) > p$, then go to step (9); otherwise, restore old deployment and return to step (4).
9   Accept the new arrangement. $E_{old} \leftarrow E_{new}$.
10  If $E_{old} < E_{min}$ , then $E_{min} \leftarrow E_{old}$.
11  $r \leftarrow r*\varepsilon$; $t \leftarrow t*\delta$,.
12  $Z_{IP} \leftarrow E_{min}$.
```

TABLE II. THE PSEUDO CODE FOR S

```
1.  Initialization.
    1.1 For j=1, repeat steps (1.2)-(1.3) until j=n.
    1.2 Set $x_{ij} \leftarrow 1$, $\forall$ $i \in \{k, C_j+k\}$.
    1.3 Set $k \leftarrow k+C_j+1$.
2   Adjustment.
    2.1 For j=2, repeat steps (2.2) until j=n.
    2.2 For every $x_{ij}$, $x_{ij}=1$, and $i \in F$, try to find a new position $x_{pj}$, $x_{pj}=0$, and $p \in \{1,i-1\}$, so that $x_{ij}$ can move to $x_{pj}$ without violating Constraints (1)&(2) with all of j's left-hand-side readers k, $k \in \{1, j-1\}$.
```

V. COMPUTATIONAL RESULTS

To evaluate the solution quality of our proposed algorithm, we use a large-scale linear programming and mixed integer programming problem solver "GLPK" (GNU Linear Programming Kit [6]) to derive the optimal values of our test data. Because of the limited capability of our hardware platform (3 Pentium 4 PCs, all equipped with 2.4GHz CPU and 512M RAM), only test sets of v1~v7 can derive the optimal values with GLPK in reasonable time. The parameters of the SA algorithm are $\delta=0.75$, $\varepsilon=1.3$, $t=0.1$ and $r=|L|x|L|$. The frozen temperature t_f is set to t/30.

Experiments I & II are performed on a network with 12 readers, as shown in Fig. 1 (b). Experiment I simulates the N coloring problem by setting all α_j to 0, while Experiment II simulates overlapping by setting all α_j to 0.5. Nine test sets, listed in Table V (a), are used in the experiments. Table III shows that SA obtains the optimal values in test sets $v_1 \sim v_7$. The lower bound values derived by Algorithm S are improved by SA in test sets v_6, v_7 in Experiment I and in v_7, v_8 in Experiment II.

TABLE III. RESULTS OF EXPERIMENTS ON FIGURE 1 (B). |L|=12.

	| F|	Experiment I			Experiment II		
		GLPK	S	SA	GLPK	S	SA
v_1	16	4	4	4	4	4	4
v_2	24	6	6	6	5	5	5
v_3	24	7	7	7	6	6	6
v_4	31	9	9	9	7	7	7
v_5	38	11	11	11	9	9	9
v_6	34	12	13	12	9	9	9
v_7	63	19	22	19	13	15	13
v_8	126	N/A	39	39	N/A	30	29
v_9	195	N/A	76	76	N/A	55	55

Experiments III & IV are performed on a network with 25 readers, as shown in Fig. 2. Like the first two experiments, all α_j are set to 0 and 0.5 respectively. Five test sets, listed in Table V (b), are used in the experiments. As the results in Table IV show, SA improves the lower bounds of test set t_5 in Experiment III and test sets t_1, t_5 in Experiment IV.

TABLE IV. RESULTS OF EXPERIMENTS ON FIGURE 2. |L|=25.

	| F|	Experiment III			Experiment IV		
		GLPK	S	SA	GLPK	S	SA
t_1	34	N/A	6	6	N/A	5	4
t_2	49	N/A	9	9	N/A	7	7
t_3	67	N/A	10	10	N/A	8	8
t_4	154	N/A	32	32	N/A	22	22
t_5	296	N/A	48	44	N/A	35	32

VI. DISCUSSION

The execution time of SA is closely related to the size of the solution space and grows exponentially according to the values of |L| and |F|. In our experiments, the execution time varied from seconds to hours. Algorithm S can derive the initial configuration in a few seconds. For system administrators, there is a tradeoff between the solution quality and efficiency. Take one test case of test set t_5 in Experiment III as an example. In this case, Algorithm S took 0.75 seconds to derive an initial value of 48; and SA took 5601.437 seconds to improve the lower bound to 44. It took more than 1.5 hours to improve the solution quality of 8.3%.

For the sake of efficiency, it is reasonable that administrators should use Algorithm S instead of SA. Note that, in Algorithm S, different renaming orders of reader's ids may derive different lower bounds. Administrators can run Algorithm S with different reader id orders many times. However, in large-scale networks, the initial value of the lower bound may not be close to the optimum. Here, the SA algorithm can improve the lower bound derived by Algorithm S. The results of our experiments on small-scale networks show that SA can approximate the optimal values.

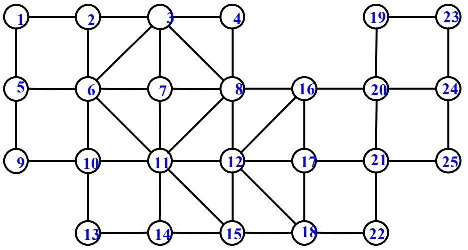


Figure 2.

VII. CONCLUSION

We have presented an SA-based algorithm that solves the problem caused by overlapping slots as well as the reader collision problem. The computational results indicate that the SA approach is effective and the solution is near optimal. The main contribution of the proposed approach is that it improves the flexibility and efficiency of RFID reader networks. It also provides an alternative solution for system administrators to consider when a tradeoff between the solution quality and efficiency is necessary.

REFERENCES

[1] B. Carbunar, M.K. Ramanathan, M. Koyuturk, C. Hoffmann, and A. Grama, "Redundant Reader Elimination in RFID Systems," *Proceedings of Sensor and Ad Hoc Communications and Networks, IEEE SECON 2005*, pp. 176- 184, September. 2005

[2] V. Cerny, "Thermodynamnical Approach To The Traveling Salesman Problem: An Efficient Simulation Algorithm," *J. Opt. Theory Appl.*, vol.45, pp.41-51, 1985.

[3] M. Dunque-Anton, D. Kunz, and B. Ruber, "Channel Assignment For Cellular Radio Using Simulated Annealing," *IEEE Transactions on Vehicular Technology*, vol.42, pp.14-21, 1993.

[4] D.W. Engels, S.E. Sarma, "The Reader Collision Problem," *Systems, Man and Cybernetics, 2002 IEEE International Conference* on vol.3, pp. 6-9, 2002

[5] Frank Y.S. Lin, "Quasi-static Channel Assignment Algorithms for Wireless Communication Network," *Information Networking 1998 (ICOIN-12) Proceedings, Twelfth International Conference*, pp. 434–437, 21-23 Jan. 1998

[6] GLPK (The GNU Linear Programming Kit), http://www.gnu.org/software/glpk/glpk.html

[7] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing," *Science*, vol.220, no.4598, pp.671-680, May 13, 1983.

[8] Sun Microsystems, "Sun EPC Network Architecture, A technical White paper."

[9] J. Waldrop; D.W. Engels; S.E. Sarma, "Colorwave: a MAC for RFID reader networks," *Wireless Communications and Networking, 2003. WCNC 2003*. 2003 IEEE vol.3, pp.1701 – 1704, 16-20 March 2003

TABLE V. THE TEST SETS

(a)

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
c_1	1	2	1	2	4	3	6	8	9
c_2	2	3	3	2	3	1	6	10	11
c_3	1	1	2	3	2	4	2	11	11
c_4	1	3	4	2	2	2	4	7	13
c_5	2	2	2	4	5	5	5	6	22
c_6	1	3	2	2	3	2	2	10	23
c_7	1	2	1	1	3	6	6	9	22
c_8	1	1	2	2	4	1	7	12	7
c_9	1	1	1	2	2	2	7	14	21
c_{10}	1	2	3	4	5	2	7	11	32
c_{11}	1	1	1	3	3	4	5	10	11
c_{12}	2	3	2	4	2	2	6	18	13

(b)

	t_1	t_2	t_3	t_4	t_5
c_1	1	3	4	8	14
c_2	2	2	4	9	6
c_3	1	1	3	12	12
c_4	2	3	2	12	3
c_5	1	1	4	10	11
c_6	2	1	2	7	13
c_7	1	2	2	4	13
c_8	1	1	3	8	14
c_9	1	3	2	7	8
c_{10}	1	2	4	2	5
c_{11}	1	1	3	5	13
c_{12}	2	3	1	4	6
c_{13}	1	2	2	3	5
c_{14}	1	2	3	4	10
c_{15}	2	2	4	8	9
c_{16}	2	1	4	9	20
c_{17}	2	3	2	8	18
c_{18}	1	3	2	5	17
c_{19}	1	1	3	6	22
c_{20}	1	2	3	7	20
c_{21}	1	2	1	4	21
c_{22}	2	1	1	2	8
c_{23}	1	1	2	1	8
c_{24}	1	3	3	3	9
c_{25}	2	3	3	6	11