韓國國防經營分析學會誌
제 29 권, 제 2 호, 2003. 12. 31.

A Genetic Algorithm Approach to the Fire Sequencing Problem

Ojeong Kwon*

Abstract

A fire sequencing problem is considered. Fire sequencing problem is a kind of scheduling problem that seeks to minimize the overall time span under a result of weapon-target allocation problem. The assigned weapons should impact a target simultaneously and a weapon cannot transfer the firing against another target before all planned rounds are consumed. The computational complexity of the fire sequencing problem is strongly NP-complete even if the number of weapons is two, so it is difficult to get the optimal solution in a reasonable time by the mathematical programming approach. Therefore, a genetic algorithm is adopted as a solution method, in which the representation of the solution, crossover and mutation strategies are applied on a specific condition. Computational results using randomly generated data are presented. We compared the solutions given by CPLEX and the genetic algorithm. Above 7(weapon)×15(target) size problems, CPLEX could not solve the problem even if we take enough time to solve the problem since the required memory size increases dramatically as the number of nodes expands. On the other hand, genetic algorithm approach solves all experimental problems very quickly and gives good solution quality.

* Office of the Analysis & Evaluation, ROK Army Hq.

1. Introduction

We consider a fire sequencing problem for the weapon-target system. We can classify the weapon-target problem arising in the military field into two regions. One is targeting problem and the other is fire sequencing problem. Targeting problem is considered to allocate weapon systems against targets so as to achieve the desired objective while satisfying various tactical/operational constraints. On the other hand, fire sequencing problem is concerned with the scheduling of the firing operation when the results of targeting problem is given. The objective of fire sequencing problem is to minimize the timespan to finish all the required firing.

The two problems may be considered together as one model. However, combined model will be too complex to solve. Moreover, there may occur a situation that a plan officer can obtain weapon-target allocation using other mathematical models and/or subjective judgment, in which case it will be more desirable to consider the problem separately. In the Republic of Korea Army doctrine, the optimal assignment and the firing sequencing problem are not enough developed. Recently, these problems are considered as the important issues according to development of C4I system and the training based on both field and simulation. In this paper, we consider only fire sequencing problem.

While many researchers have considered targeting problem for several decades, fire sequencing problem has rarely been studied [18]. Flood [1, in Ash] formulated a targeting problem as a problem of minimizing enemy threat with nonlinear objective function and linear constraints. Danzig [20, in Manne] modified Flood's model with reliability concept. Manne [20] transformed Flood's model into a transportation problem approximately and solved it. Ash [1] solved Flood's model that consists of small kill probability by Lagrangian multiplier method.

Miercort and Soland [21] proposed optimal allocation of missiles against area and point defenses by heuristic methods and Grotte [14] considered a targeting model that minimizes collateral damage. Bracken and McCormick [5] considered a targeting problem with nonlinear objective function and general assignment constraints and they also considered a mathematical model for inflicting specified damage with linear objective function. They approximated the problems to the linear programming models, and solve the problems and rounded the fractional solutions to nearest integers.

Orin [23] considered optimal allocation of weapons against layered defenses by minimum cost network model. Sherali et al. [26] suggested 5 kinds of linear functions that provide lower bounds on the nonlinear objective function for the probabilistic partial set covering problem. The objective function of the probabilistic partial set covering problem has a similar structure with a targeting problem of minimizing the total enemy threat. Kwon et al. [17] solved a targeting problem composed with various tactical constraints by Lagrangian relaxation. Other researchers and analysts have made efforts to solve their specific targeting problems by heuristic methods [9,12].

In the fire sequencing problem region, although many scheduling problems have been researched, most of the results can not be used to solve the fire sequencing problem because important assumptions of the fire sequencing problem are different from ordinary scheduling problems. Fire sequencing problem is very important when we minimize the time span under known results of targeting problem. It can be used as a sub-module of C4I (Command, Control, Communications, Computer and Intelligence) system and it plays a key role of the distribution of fire. As the computerized system becomes powerful, fire sequence problem will be more significant subject in military field. Kwon et al. [18] defined the concept of fire sequencing problem and presented a heuristic method based on the node packing problem. Lee et al. [19] showed the computational complexity of the problem is strongly NP-complete. However, in their early work, they assumed that the density of weapon-target allocation matrix is very sparse and the heuristic method they developed is very effective on such a problem. In their paper, they suggested a further research under a dense weapon-target allocation matrix should be done. So, it is necessary to develop a general method to solve the fire sequencing problem in various environments.

The constraints of fire sequencing problem are as follows. Considering from the weapon system side, a weapon system can fire only one target at a time. Next, if a weapon starts firing against a target, it should terminate firing after consuming all planned rounds. On the other hand, considering from the target side, the fire starting time on a target from all weapon systems assigned to the target should be the same. However, the fire terminating time of each weapon on a target may be different.

In the manufacturing systems, jobs are generally processed by only one machine at a time and optimize various measurements [24]. However, for the fire sequencing problem, a target may be fired on by more than one weapon system simultaneously. In the previous researches, the simultaneous resource scheduling bears resemblance with the fire sequencing problem that a machining operation may require the multiple resources simultaneously[10]. In the early days, people in the computer and communication systems paid attention to simultaneous resource scheduling. Recent researches about simultaneous resource scheduling are concerned with the manufacturing systems. Dobson and Karmarkar [10] suggested the Lagrangian relaxation and the surrogate relaxation for simultaneous resource scheduling for providing a lower bound. However, it has somewhat different faces with the fire sequencing problem. For the detailed contents of comparison between the fire sequencing problem and the simultaneous resource scheduling problem, see Kwon et al. [18].

We present the integer programming formulation of the fire sequencing problem and analyze the problem in section 2. We take a genetic algorithm approach for the fire sequencing problem in section 3. We report computational results with randomly generated data in section 4. Finally, we give concluding remarks in section 5.

2. Problem Description

To formulate the fire sequencing problem, let us denote the following notations.

W : Set of weapons,

T : Set of targets,

t_j : Fire starting time against target j, for all $j \in T$

α_{ij} : Planned firing duration from weapon i to target j,

$W(j,k)$: Set of weapons which fire target j and k.

Fire Sequencing Problem (FSP)

$\min \; t_f$

$$\text{s.t} \quad t_f \geq t_j + \max_{j \in W}(\alpha_{ij}) \text{ for all } j \in T \qquad (1)$$

$$t_j - t_k \leq -\max_{j \in W(j,k)}(\alpha_{ij}) + M(1 - y_{jk}) \text{ for } j < k \quad \text{for all } j, k \in T \qquad (2)$$

$$t_k - t_j \leq -\max_{j \in W(j,k)}(\alpha_{ij}) + M(y_{jk}) \text{ for } j < k \quad \text{for all } j, k \in T \qquad (3)$$

$$t_i \geq 0 \quad \text{integer} \quad \text{for all } i \in T$$

$$y_{ik} \in \{0, 1\} \text{ for all } j, k \in T, j < k$$

where, M is a large number

The binary variable y_{jk} is equal to 1 if and only if target j precedes target k to be attached, for all $j, k \in T, j < k$. FSP has $|T| + 1$ general integer variables and $|T|(|T| - 1)/2$ binary variables and $|T|^2$ constraints. Constraints (1) mean the dummy target's starting time, t_f , is the last and we wish to minimize it. Constraints (2) and (3) are the disjunctive constraints which represent the starting time relationship. The same starting time of some weapons is reflected by right hand side value. If target j and k are available on same time, $-max_{i \in W(j,k)}(\alpha_{ij})$ and $-max_{i \in W(j,k)}(\alpha_{ik})$ will be 0 and $t_j = t_k$ can be accomplished.

We can regard fire sequencing problem as a job sequencing problem. The machines correspond to the weapons and the jobs to the targets. While simultaneous resource scheduling problem requires that the operation termination time of a job on multiple machines must be the same [10], fire sequencing problem may have different termination time. Therefore, simultaneous resource scheduling problem can be regarded as a special case of fire sequencing problem [19].

One of the fire sequencing problem properties is the history dependent attribute. To minimize the timespan, we have to consider the proper target sequence. However the optimal target sequence of a target is influenced by all the targets allocated ahead. It is different from the one machine scheduling with sequence dependent set-up time problem [24]. One machine scheduling with sequence dependent set-up time problem can be solved by traveling salesman problem [24]. However, since fire sequencing problem should consider all history ahead of a target, traveling salesman problem can not be applied for fire sequencing problem.

The computational complexity of FSP is strongly *NP-complete* and remains *strongly NP-complete*[13] even if the number of weapons is two[19]. Now we compare the complexity of FSP with the complexities of other well-known scheduling models. First, consider the parallel machine scheduling problem with the objective of minimizing makespan without preemptions [24]. The problem is known to be NP-complete even if there are only two machines. However, the problem can be solved in pseudo-polynomial time if the number of machines is fixed. It is well-known that the scheduling problem with the objective of minimizing makespan in flow shops with unlimited intermediate storage can be solved in polynomial time when there are only two machines [24], though the problem with three machines is strongly NP-complete. Lee et al. showed that FSP is already strongly NP-complete even if only two weapons (machines) are available. Moreover, even when the processing times (α_{ij}) are all the same, the problem remains to be strongly NP-complete, but in this case, the other two problems (parallel machines, flow shop) can be solved trivially. These results show that FSP is a very difficult scheduling problem.

About simultaneous resource scheduling problem, the objective of it is to minimize the total weighted flow time of n tasks on m resources. Each task j requires a subset S_j of the m resources and consumes time t_j. The additional restriction is that the task j must capture all of the resources S_j simultaneously. This problem is known to be NP-complete [10]. Differences between the simultaneous resource scheduling problem and FSP lie in 1) the objective function and 2) a task j does not release any of the resources S_j until it is completely processed in the simultaneous resource scheduling problem.

3. Genetic Algorithm for FSP

3.1 Introduction

Genetic algorithm(GA) is biologically inspired search method borrows mechanisms of inheritance to find solutions [16]. GA is a general purpose search method that can be used to provide heuristic solutions to hard combinatorial optimization problems. GA searches a problem space with a population of structure and select structures for continued search based on their performance. Each structure decodes to form a point in the problem space in the context of optimization problems[7].

GA has achieved successfully in field of many industrial engineering and management science, especially job-shop scheduling, keyboard configuration design, optimization and pipeline systems, traveling salesman problem, and multi-vehicle routing problem. Therefore, GA is a reliable approach to obtain solutions in hard combinatorial optimization problems[3, 6, 22, 27].

GA repeats a typical procedure to get a solution in a generation. However, it has a flexible structure that can change the procedure and adopts various strategies according to the characteristic of the problem. So, researchers have developed various strategies and methods to get good solutions in a short time. The solution of one generation evolves by the crossover and mutation operations as the generation proceeds. To make the implementation of GA on a specific problem, first of all, we have to represent the problem solutions (genetic encoding) that can be manipulated (through some sort of crossover or mutation) to yield other candidate solutions to the problem. Second, acting on an initial population, these transformations create the next generation of candidate solutions. Third, calculation of the objective function for each candidate solution supplies a measure of fitness which affects its likelihood of leaving surviving offspring in the next generation. Selection pressure is the tendency toward the survival of the fittest; high selection pressure means low probability of the survival of the less fit [25]. For the detailed concept of GA, see David[8].

```
Begin
 t←0
P(t) Initialization ( Generation of initial population )
Evaluation of P(t)
While( not satisfy the condition of termination ) do
  Begin
   t←t +1
  Selection P(t) from P(t−1)
  Genetic operations (Crossover and mutation)
  Evaluation of P(t)
  End
End
```

Figure 1. The overall procedure of the genetic algorithm

3.2 Representation of Solution Structure

In FSP, a solution structure can be represented in the sequence of impact for each target based on the location order. For example, the structure of (4 1 3 2) means target 1 is impacted by all assigned weapons for the fourth time and target 2 is impacted for the first time, and so on. In case of simultaneous impact on the different targets, we adjust the sequence of impact when we consider the fitness function evaluation. So, the structure of solution we set up is a type of cyber sequences not an actual one, that is, the solution structure (1 3 2 4) may mean (1 1 2 2) or (1 2 2 3) in the implementation. The cyber sequence restricts the sequence of firing strictly, it means that the later number of target should be impacted after the completion of impact for the earlier number of target. But, in the real situation, if we have available time to fire at the same time against the different targets, we fire them at the same time to finish the firing as soon as possible. The more detailed procedure of adjustment will be explained in section 3.4.

3.3 Generation of initial population and population size

The proper size of initial population is important to perform genetic algorithm to solve the problem successfully arising in many industrial engineering fields. We set the population size to be 100 and generate the initial population with random number generator. Every locus in the solution structure is represented differently from the other locus. As mentioned section 3.2, the simultaneous locus that can be occurred in the actual firing sequence problem may be represented in the evaluation procedure.

The length of solution structure is equal to the number of targets since the solution structure is represented the sequence of impacts.

3.4 Fitness function and the selection of parents

In the research of the genetic algorithm in FSP, the evaluation of the fitness function and the selection of parents are crucial procedures. First, we converted the current solution structure to the one that has actual representation of string. We decide the sequence of firing and the exact time of impact considering the matrix of weapon-target allocation results. It is a very simple procedure that decides the firing time in a given firing sequence. When we decide the actual sequence of impact based on solution structure, we have to consider both of the weapon-target allocation results and the solution structure.

$$\begin{array}{rl} \text{Target} & \begin{array}{ccccc}1&2&3&4&5\end{array} \\ \begin{array}{r}1\\ \text{Weapon}2\\ 3\end{array} & \begin{pmatrix}2&1&1&1&0\\0&0&1&0&3\\3&1&0&2&0\end{pmatrix}\end{array} \qquad\qquad (3\,1\,2\,4\,5)$$

[Firing time duration matrix] [solution instance]

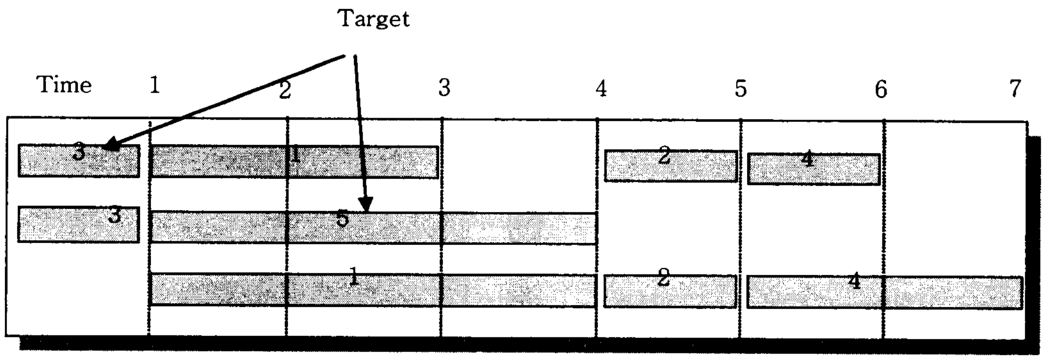

Figure 2. Sequence of firing given a targeting results and solution structure

Let's take one example. Assume that we have a weapon-target allocation result and solution structure respectively as Figure 2. The weapon-target allocation result indicates that the number of weapon and target are 3 and 5, respectively. The number of element in matrix means the firing time duration of from weapon i to target j.

As explained above, a target should be impacted simultaneously by the assigned weapons and a weapon cannot transfer the firing against another target before all planned rounds are consumed. So, the minimum termination time of firing is 7 unit duration. Even if the sequence of impact is represented differently in the potential solution structure, the targets 1 and 5 are impacted simultaneously in the evaluation procedure of fitness function. Furthermore, the impact time of target 5 is represented as the last sequence in the solution structure, its real time of impact from assigned weapons is ahead of target 2 and 4 in implementation. It is due to the philosophy of idle weapon system restriction.

To compute the firing time duration under given weapon-target allocation and solution structure, let's define following notations. As defined in chapter 2, let W, T and α_{ij} be the set of weapons and targets and the firing duration from weapon i to target j, respectively. The first thing we have to do is to rearrange and to reindex of the columns with respect to the solution structure. We reindex the targets temporary to evaluate the firing duration as the sequence order. The additional notations are shown as follows.

k : Reindexed sequencial target index, $k \in \{1, 2, 3, \ldots |T|\}$

$L(k)$: The maximum firing time duration up to target index k

$l(i,k)$: The maximum firing time duration to the current target index k with respect to weapon i

Then the algorithm that finds the firing time duration on a specific solution can be represented as follows.

Initialize) $k=1$

$L(k)=0$

l(i,k)=0 for all $i \in W, k \in T$

Step 1) Termination condition check

If $k=|T|+1$ Print *L(k-1)* and stop

If $k = 1$, go to step 3

Step 2) Find the maximum previous firing time duration

$$\xi = max\{l(i,k-1)|\alpha_{ik} > 0, \text{ for } all \;\; i \in W\}$$

$$l(i,k-1) = \xi, \text{ for } all \; |\alpha_{ik} > 0 \;\; i \in W$$

Step 3) Compute the l(i,k) and find the current maximum length

If k=0 $l(i,k) = \alpha_{ik}$

Else $l(i,k) = l(i,k-1) + \alpha_{ik}$

$L(k) = \max_{i \in W}\{l(i,k)\}$

Step 4) Update index k

$k = k + 1$

go to step 1.

An example for evaluating a firing time duration is presented below. Let's assume that we have a weapon-target allocation matrix and a solution instance as shown below.

Target — Solution instance

$$\text{Weapon} \quad \begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{pmatrix} 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 3 & 2 & 0 \\ 1 & 0 & 0 & 0 & 3 \end{pmatrix} \end{array} = [\alpha] = A \qquad (3\ 1\ 5\ 2\ 4)$$

The first thing we have to do is to change the columns according to the solution instance. Rearranged columns with respect to the solution instance is shown below.

$$
\begin{array}{lcc}
 & & \text{Target} \\
 & & 3\ 1\ 5\ 2\ 4 \\
\text{Weapon} & \begin{array}{c}1\\2\\3\end{array} & \begin{pmatrix} 2 & 0 & 0 & 2 & 0 \\ 3 & 0 & 0 & 0 & 2 \\ 0 & 1 & 3 & 0 & 0 \end{pmatrix} \\
\text{Sequential index} & k & 3\ 1\ 5\ 2\ 4
\end{array}
$$

As shown above, the columns of matrix A are moved their places according to the solution instance and the column indexes are designated as a sequence order. The algorithm above gives following l(i,j) matrix, L(k) and the configuration of firing time duration as follows.

$$
\begin{array}{lcc}
 & & \text{Target Index} \\
 & & 1\ 2\ 3\ 4\ 5 \\
\text{Weapon} & \begin{array}{c}1\\2\\3\end{array} & \begin{pmatrix} 2 & 2 & 2 & 4 & 4 \\ 3 & 3 & 3 & 3 & 5 \\ 0 & 1 & 4 & 4 & 4 \end{pmatrix} = [l(i,j)]
\end{array}
$$

K	1	2	3	4	5
L(K)	3	3	4	4	5

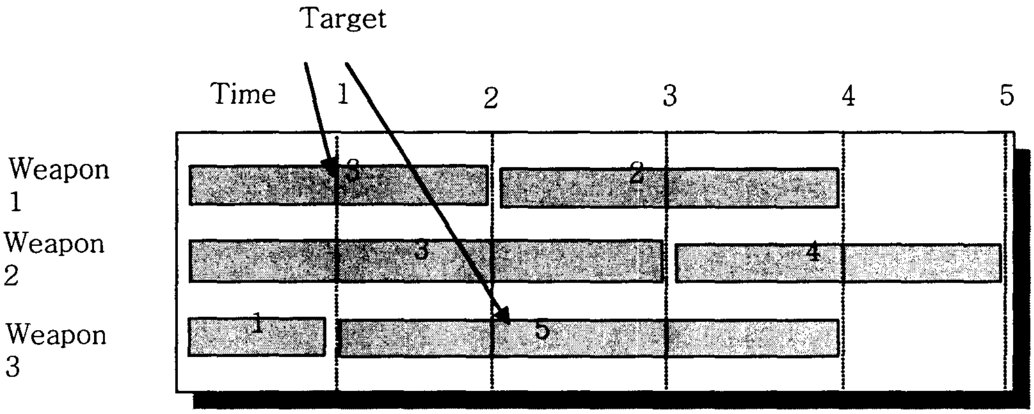

Figure 3. Fire sequence of an example problem

To select the parents in a population for the next generation, we adopt the geometric function based on ranking order, that is, $Prob(r) = q(1-q)^{r-1}$, $0 < q < 1$ where, r is a rank of fitness function and q is a parameter. Therefore, $Prob(r)$ represents the probability of selection that has rank r in the fitness function. As q becomes larger, the difference of selection probability also becomes larger. If the number of population(Np) is a large number, the summation of probability becomes 1, $\sum_{r=1}^{Nq} Prob(r) = \sum_{r=1}^{Nq} q(1-q)^{r-1} \approx 1.$, approximately. Here, we decide the rank of fitness by the firing time duration of each solution structure. In this paper, we set q to be 0.05.

3.5 Genetic operators

In this paper, we carried out two genetic operators, which are the crossover and mutation. The crossover operation is a phase that we generate the children objects from the parent. We adopted the order based crossover strategy that is developed by Davis(1985). To apply that, we initially designate arbitrary two points in the string of gene. The offspring(child) 1 is inherited the chromosomes located between two cutting points from parent 1, and the remaining chromosomes are taken from parent 2 in the restriction of not taking the same chromosomes in the parent 1. The offspring 2 is inherited the genes as the same way except the sequence of choice for parents.

We give the crossover ratio as 25 %. In a crossover procedure, we generate a random number in the interval of [0, 99]. If the number is less than 25, we carry out the crossover process. Figure 4 is an example of crossover process.

First, two cutting points for the parents are decided 3rd and 7th locations by the random numbers generator. The child 1 is inherited the genes (4 5 6 7) located between two cutting points from parent 1. Next, the remaining genes of child 1 are taken from the parent 2.

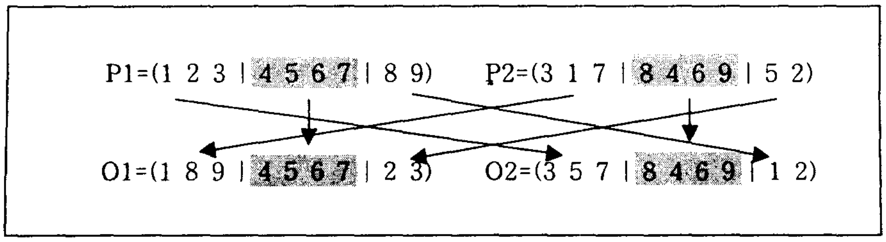

Figure 4. Crossover operation

We start to bring the genes of latter parts of child 1. The latter part of genes is (5 2). However the gene 5 already exists in the middle part of child 1. Therefore, we discard gene 5 and reselect the next candidate among the available set of gene. So, we accomplish the crossover process with O1=(1 8 9 | 4 5 6 7 | 2 3). The child 2 is got the same procedure of that of child 1.

Mutations are occurred in each generation as 5 % ratio. In a mutation procedure, we generate a random number in the interval of [0, 99]. If the number is less than 5, we carry out the mutation process. First, we take arbitrary two locations with random number generation in solution structure, and exchange the two genes.

3.6 Termination criterion

We set the termination criterion the number of generation to be 100. Although we can have many types of criterion, the number of generation criterion is a simple and easy method.

4. Computational Results

To test FSP, we randomly generated data. We generated 60 test problems for FSP in 6 kinds of weapon-target pairs, respectively. To reflect the reality, the data were randomly generated in the interval from 1 to 10. The weapon sizes were estimated by considering the number of field artillery battery in the battalion, regiment, division and corps level. In this paper, we set the number of target to be about 1.2-2 times the number of weapon systems so as to reflect the flexible combat situations.

All problems could be solved within a reasonable time on a Pentium PC(200MHz, 32 Mb Memory). To evaluate the results of genetic algorithm, we run a popular commercial code(CPLEX 4.0) that uses LP relaxation and branch-and-bound method.

The results are summarized in Table 1. It shows the average, maximum and minimum values of the CPU time for each problem size and the gap between the solution and a lower bound. We already know a lower bound of FSP, $max_{i \in W}(\Sigma_{j \in T} \alpha_{ij})$. This lower bound means that the minimum firing duration should be larger than the maximum summations of firing time duration in all target that place in a row. As showed in Table 1, the genetic approach can obtain solutions in a very short time. On the other hand, CPLEX did not solve the problem to the optimality in a reasonable time. Node limit in CPLEX is designated to save the time and compare the incumbent solution. Above 7×15 size problems, CPLEX could not solve the problem even if we take enough time to solve the problem since the required memory size increases dramatically as the number of nodes expands. So, we restricted the maximum number of node to be 30,000 in CPLEX.

CPLEX could solve only small size problem. In 5×7 size problem, both of the genetic algorithm and CPLEX give the optimal solutions. However the CPU time of CPLEX took about 3.7 times of genetic approach. As the problem size grows, the Integer Programming (IP) approach with CPLEX could not solve the problem since the number of node increases exponentially and could not fathom nodes because of bad lower bound. In 30 x 50 size problem, CPLEX took 813.9 seconds to the node limit 30,000(unsolvable to optimal solution), but the GA took only 13.0 seconds on average, and GA gives better solution.

Table 1. Computational results

Problem Size(Wepon × Target		CPU Time(Second)		GAP of Inclumbent Solution(%)		Cplex
		GA	Cplex *	GA ★	Cplex ★★	No..of Nodes
5 × 7	Ave	1.7	6.4	7.9 **	7.9**	4040
	Max	2.0	10.88	14.7**	14.7**	6965
	Min	1.0	6.44	0.0**	0.0**	466
7 × 15	Ave	2.1	86.6	24.0	35.4	*
	Max	3.0	92.8	32.4	40.5	*
	Min	2.0	82.9	17.4	24.0	*
10 × 20	Ave	3.0	127.5	29.6	41.6	*
	Max	4.0	143.1	33.1	46.6	*
	Min	3.0	122.4	24.7	36.1	*
15 × 30	Ave	5.1	291.9	39.6	46.4	*
	Max	6.0	286.2	47.3	52.2	*
	Min	5.0	308.3	27.7	38.7	*
20 × 40	Ave	8.0	557.7	48.1	55.2	*
	Max	9.0	603.4	53.2	59.3	*
	Min	7.0	520.2	41.8	49.2	*
30 × 50	Ave	13.0	813.9	51.1	57.3	*
	Max	14.0	823.3	53.6	64.1	*
	Min	12.0	803.8	49.0	52.3	*

The ratio of CPU time(GA/CPLEX) has a large amount of gap as the problem size grows. This means that IP method is not proper approach. Generally, the

* : Run Cplex with branch-and-bound node limit 30,000

** : Optimal Solution

★ : The mean ratio of lower bound to solution value of Genetic algorithm

$$= \frac{\textit{Solution value of genetic algorithm} - \textit{Lower bound}}{\textit{Solution value of genetic algorithm}} \times 100$$

★★ : (Z-Lowerbound)/ (Incumbent value of IP(Z)) x 100

Lower bound : $\max_{i \in W}(\sum_{j \in T} \alpha_{ij})$

LP(Linear Programming) optimal bound of FSP in CPLEX is ordinary worse than a lower bound, $\max_{i \in W}(\Sigma_{j \in T}\alpha_{ij})$. Figure 5 shows the mean time of CPU seconds of both approaches.

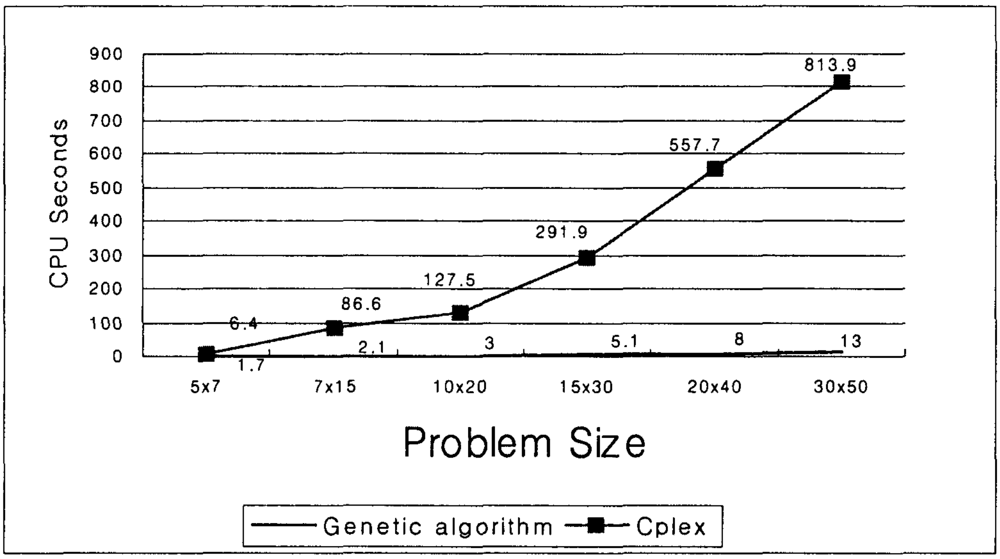

Figure 5. The mean time of CPU seconds

Figure 5 shows the CPU time(Seconds) between genetic approach and CPLEX. As mentioned above, we restricted the node limit in CPLEX, 30,000. The time of CPLEX is represented to the node 30,000 except 5 × 7 size problem. Nevertheless, IP approach is not effective not only CPU time but also the quality of solution.

In the side of solution quality, genetic algorithm approach gives better solutions than those of CPLEX. As mentioned above, we give the node limit in CPLEX to compare the solution. In the node limit 30,000, the gap of CPLEX defined above shows much than that of genetic algorithm. The genetic algorithm improves 3.3 % - 11.4 % in the gap defined in Table 1 according to the problem sizes. This result should be considered with the elapsed CPU time. So, we can conclude that GA is better solution approach than IP approach. Moreover, the genetic algorithm approach solves all problem sizes we considered within 14.0 CPU seconds. Figure 6 shows the mean ratio of GAP defined in Table 1.

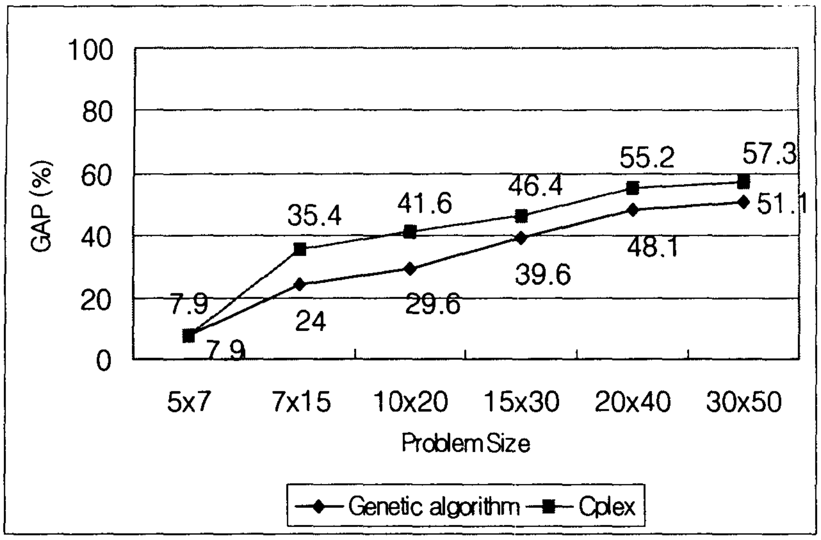

Figure 6. The mean GAP of incumbent solution

5. Conclusion

We introduced the fire sequence problem that seeks to minimize the timespan and proposed a solution method. To minimize the timespan is a criterion of deciding a crucial factor of tactics. When we plan the fire sequencing with results of targeting problem, we have to know the minimum time required for firing. If the timespan is not satisfied with the requirement of operation, we have to consider the trade-off between the timespan and overall cost of targeting problem. Thus, there are many requirements that can provide the guidance deciding the sequence of firing in military.

We formulated the fire sequencing problem by the integer programming. Preliminary tests showed that the integer programming approach performs poorly on this problem [18]. Therefore, we develop a genetic algorithm approach for the fire sequencing problem. The genetic algorithm solved the problem in a reasonable amount of time. Moreover, it found good incumbent solutions in all test problems in any type of matrix density of weapon-target allocation tables.

When we need to fire some targets on an exact time slot, those targets should be allocated first on required time slot. Moreover, when we need to evaluate the performance of fire sequencing problem with other criteria and/or constraints, we need to modify the presented fire sequencing problem. These studies will be the extensions of the fire sequencing problem provided in this paper.

References

[1] Ash, M., Flood's Assignment Model for Small Kill Level, Operations Research Vol. 7, pp.258-260, 1959.

[2] Baker, K., Introduction to Sequencing and Scheduling, Wiley, New York, 1974.

[3] Biegal, J., Draven, J., Genetic Algorithm and Job Shop Scheduling, Computers and Industrial Engineering Vol. 19, pp.81-91, 1990.

[4] Bracken, J., Layered Defense of Deceptively Based ICBMs, Naval Research Logistics Quarterly, Vol. 31, pp.653-670, 1984.

[5] Bracken, J., McCormick, G., Selected Application of Non-linear Programming, John Wiley & Sons, New York, 1980.

[6] Charles, J., A Genetic Algorithm for Service Level Based Vehicle Scheduling, European Journal of Operations Research, Vol. 93, pp.121-134, 1996.

[7] Chen, C., Vempati, V., Aljaber, N., An Application of Genetic Algorithm for Flow Shop Problems, European Journal of Operations Research, Vol. 80, pp. 389-396, 1995.

[8] David, G., Genetic Algorithm in Search, Optimization, and Machine Learning, Addison-Wesley, Reading MA, 1992.

[9] Day, H., Allocating Weapons to Target Complexes by means of Nonlinear Programming, Operations Research, Vol. 14, pp.992-1013, 1966.

[10] Dobson, G., Kamarkar, U., 1989. Simultaneous Resource Scheduling to Minimize Weighted Flow Times, Operations Research, 37, 592-600.

[11] Firstman, I., An Approximating Algorithm for an Optimum Aim-Points Problem, Naval Research Logistics Quarterly, Vol. 7, pp.151-167, 1960.

[12] Furman, G., Greenberg, J., Optimal Weapon Allocation with Overlapping Area Defenses, Operations Research, Vol. 21, pp.1291-1308, 1973.

[13] Garey, M., Johnson, D., Computers and Intractability, Bell Telephone

Laboratories Incorporated, New Jersey, 1979.

[14] Grotte, J. H., A Targeting Model that Minimizes Collateral Damage, Naval Research Logistics, Vol. 25, pp.315-322, 1978.

[15] Ho, J., Chang, Y., A New Heuristic for the n-jobs, m-machine flow-shop problem, European Journal of Operations Research, Vol. 52, pp.194-202, 1991.

[16] Kendall, E., Rhonda, K., Ramesh. S., O.R/MS Today, Vol. 8, 1992.

[17] Kwon, O., Kang, D., Lee K., Park, S., Park, K., Lagrangian Relaxation Approach to the Targeting Problem, Naval Research Logistics, Vol. 46, pp.640-653, 1999,

[18] Kwon, O., Lee, K., Park, S., Targeting and Scheduling Problem for Field Artillery, Computers and Industrial Engineering, Vol. 33, pp.693-696, 1997.

[19] Lee, K., Kwon O., Park, S., Park, K., Complexity of the Fire Sequencing Problem, International Journal of Management Science, Vol. 5, pp.55-59, 1999.

[20] Manne, A., A Target Assignment Problem, Operations Research, Vol. 5, pp.346-351, 1958.

[21] Miercort, F., Soland, R., Optimal Allocation of Missiles against Area and Point Defenses, Operations Research, Vol.19, pp.605-617, 1970.

[22] Moutaz, K., Zbigniew, M., Michael, W., The Use of Genetic Algorithm to Solve the Economic Lot Size Scheduling Problem, European Journal of Operations Research, Vol. 110, pp.509-524, 1998.

[23] Orin, D., Optimal Weapons Allocation against Layered Defenses, Naval Research Logistics, Vol. 34, pp.605-617, 1987.

[24] Pinedo, M., Scheduling, Theory, Algorithms and Systems, Prentice Hall, New Jersey, 1995.

[25] Sangit, C., Cecilia, C., Lucy, L., Genetic Algorithms and Traveling Salesman Problems, European Journal of Operations Research, Vol. 93, pp.490-510, 1996.

[26] Sherali, H., Kim, S., Parrish, E., Probabilistic Partial Set Covering Problem, Naval Research Logistics, Vol.38, pp.41-51, 1991.

[27] Taillard, E., Some Efficient Heuristic methods for the Flow Shop Sequencing Problem, European Journal of Operations Research, Vol. 47, pp.65-74, 1990.