ISSN 1023-2842
Sun Yat-Sen Management Review
2002 International Issue pp.135-157

A Genetic-Based Heuristic Approach Toward Maintenance Scheduling Of Oil Tanks

Sheng-Tun Li
Li-Yen Shue
Pei-Chun Lin
National Kaoshiung First University

ABSTRACT

The oil market in Taiwan is becoming very competitive nowadays than before, which is due to her recent entry into WTO and the liberalized Petroleum Management Law that allows international petroleum vendors to sell oil products in Taiwan market. For a competitor to become successful, the control of gas stations, pipelines, and storage tanks plays a key role, however, due to the space limitation and the residents' increasing awareness of environmental protection issues on the island, the construction of new storage tanks is being recognized as an issue next to impossible. Hence, any new vendors, undoubtedly, would have to rent tanks from existing oil companies for operation. Thus, tank owner must schedule tanks efficiently to satisfy the maintenance requirements, and, at the same time, meet the operation needs for itself and leaseholders. In this study, we examine the tank maintenance scheduling needs of an existing company, and investigate the performances of an evolutionary computing approach, which is based on genetic algorithm and simulated annealing. Experimental results with Integer Programming were used to highlight the problem with this conventional approach. We also conducted an extensive comparison with GA and GASA to confirm the superior performance of this hybrid approach in tank utilization and computation times.

Keywords: scheduling optimization, genetic-simulated annealing algorithm, genetic algorithm, oil-tank maintenance scheduling

Introduction

In coupling with her recent entry into WTO, Taiwan government has recently revised national Petroleum Management Law, which has removed the monopoly element and has completely liberated the oil market in Taiwan. Any international petroleum vendor can sell their oil products in Taiwan nowadays; this liberalization has also brought in an intensive competition to the local oil market. The revised Petroleum Management Law requires that vendors of refining and importing must maintain a minimum safety reserve of 60 days or 50,000 kiloliters. For a vendor to remain in a competitive position, it is necessary to be able to control the distribution channel of products, which consists of storage tanks, pipelines, and gas stations. Therefore, for any new competitor, especially the international ones, it is essential for them to have storage tanks facility. However, due to the problem of very limited space available for constructing new tanks and the residents' increasing awareness of environmental protection issues on the island, the construction of new storage tanks is such a thorny issue that few people even like to mention about it. The only other alternative for any new comer is to rent tanks from existing oil companies. Thus, the owner of existing tanks must be able to schedule tanks effectively to meet its own needs as well as the needs of new comers.

Due to safety concern, tank owners must follow the American Petroleum Institute (API) standard 650 to conduct physical maintenance of tank facilities, and, during the maintenance period, a tank will be completely out of operation. It currently requires the storage tanks must be inspected every two years, and depending upon the corrosion degree inside the tanks, a so-called "open inspection" procedure will be held every five to ten years, and each tank may take 60 to 240 days of outage for each open inspection; depending on the capacity and construction type of tanks. Thus, the development of a good maintenance schedule of storage tanks can substantially help leaseholders increase the availability of tanks and revenue for the owner.

In developing tank maintenance schedules, one existing company relies on both the tacit knowledge of senior engineers and the package of linear programming (LP). Initially, LP is applied to produce a basic schedule, which provides only a reference schedule, and will be revised by the experience of senior engineers whenever they feel their knowledge can improve the schedule. This interactive approach can sometimes be very ineffective, but management was not too much concerned with this problem in the past, when the monopoly law was in effect and had excluded all other competitors. The present competition environment has forced the management to pay attention to the efficiency problem, and seek better methods to improve tank facility utilization that will benefit both the owners as well as leaseholders. In literatures, due to the integer nature of items in maintenance scheduling, Integer Programming had been a major approach (Dopazo & Merrill, 1975; Kralj & Rajakovic, 1994; Mukerji & Parker, 1999) in the past. However, its usefulness was usually hindered by the curse of dimensionality and is poor in handling the nonlinear objective and constraint functions that characterize the scheduling problem. These methods give an optimal solution for reasonably sized problems. However, in the case of large scale problems, the difficulty with dimensionality could limit the applications of this mathematical optimization technique.

Another well-known heuristic optimization paradigm that has been applied to scheduling problems is genetic algorithm (GA) (Goldberg, 1989; Storer, Wu & Park, 1993; Gen & Cheng, 1997; Jain & Meeran, 1998; Man, Tang & Kwong, 1999; Deris, et. al., 1999; Negnevitsky & Kelareva, 1999; Lapa, Pereira & Mol, 2000; Wang & Handschin, 2000), which is based on the "natural selection" of the evolutionary mechanism that tends to move toward optimizing the behavior of a system. It, in general, iteratively constructs a population of individuals, evaluating their fitness, and generating a new population through genetic operations until a predetermined number of iterations (generations) is reached or the fitness value converges. This method seems to offer hopes for overcoming the difficulty IP has experienced. However, GA, as is known, can suffer from the non-convergence or early-convergence (Rudolph, 1994; Ting, Li & Lee, 2002) for some problems. As a result, GASA (Sun, Dayhoff & Weigand, 1994) was recently proposed as a hybrid optimization approach by combining the features of both GA and SA (simulated annealing), it could keep the features of GA, and, at the same time, avoid the non-convergence or early-convergence deficiency. This method may present a better alternative to address maintenance scheduling problems than GA.

The objective of this research is to investigate the performances of GASA in scheduling oil tanks to meet maintenance requirements of one petroleum company and maximize its minimum net reserve for leasing purpose. The performances of conventional approaches of Integer Programming (IP) will be investigated first and used for comparison purpose. In Section 2, we formulate the maintenance-scheduling problem of storage tanks in an IP model and present its results. Section 3 provides a detailed description of GASA. Its scheduling results and that of GA are given in Section 4. Finally, conclusions are given in Section 5.

Integer Programming Formulation And Results

From the Integer Programming point of view, this problem falls into the category of multi-period planning, where the consequences of decisions made in earlier periods can affect the decisions of future ones. The core of its formulation is the consideration of interactions between earlier periods and future ones. However, this seemingly difficult problem can be expressed in Integer Programming optimization model as through they are actually decoupled.

In this particular case, the objective is to develop a tank maintenance plan for the coming year to maximize the minimum weekly net reserve during the period; hence it is sensible to have the period correspond to weeks of the year. Thus, the period of the planning horizon in a year is 52-week. The inter-period interactions are usually accounted for in models by the introduction of the "in-inspection" decision variable, and these variables "link" adjacent periods. The decision that is to be made at a period j is whether to start inspection on tank i, with each tank i having an individual capacity c_i and requiring m_i weeks for maintenance. Total demand for week j is D_j and the total capacity for the petroleum company is C. With the total number of tanks T, we formulate the model for this problem in the following.

In determining when each tank should start inspection, we define

$$s_{ij} = \begin{cases} 1 & \text{if tank } i \text{ starts inspection in week } j \\ 0 & \text{otherwise} \end{cases}$$

where $i = 1..T$, and $j = 1..52$.

This leads us to define a status variable for each tank:

$$a_{ij} = \begin{cases} 1 & \text{if tank } i \text{ is in inspection in week } j \\ 0 & \text{otherwise} \end{cases}$$

Since the objective is to maximize the minimum of 52 weekly net reserves, we define the weekly net reserve as

$$R_j = C - \sum_{i=1}^{T} a_{ij} c_i - D_j$$

The objective function thus becomes

$$\text{MAX}\{\text{MIN } R_j = C - \sum_{i=1}^{T} a_{ij} c_i - D_j\}$$

The constraints of this problem are caused by the needs of tank outage for open inspection on a regular base, and the petroleum company decides that all inspections must be completed in the same year. This can be translated into the following constraints:

Constraint 1. All tanks must start inspection during this year.

Constraint 2. All tanks must finish inspection during this year.

Constraint 1 is expressed as

$$\sum_{j=1}^{52} s_{ij} = 1, \forall i$$

Constraint 2 is expressed as

$$\sum_{j=1}^{52} a_{ij} = m_i, \forall i$$

Thus, $s_{ij} = 1$ should also imply $\sum_{k=j}^{j+m_i-1} a_{ik} = m_i$ ($j + m_i - 1 \leq 52$), and this relation is expressed in the following constraints $\forall i, j$:

$$s_{ij} \leq M y_{ij}$$

$$m_i - \sum_{k=j}^{j+m_i-1} a_{ik} \leq M(1 - y_{ij}),$$

where $y_{ij} = 0 \text{ or } 1$ is to ensure that the constraint $s_{ij} \le 0$ must be satisfied, if $m_i - \sum_{k=j}^{j+m_i-1} a_{ik} > 0$ is true; otherwise $s_{ij} \le 0$ may not have to be satisfied. M is a large positive number, so that $m_i - \sum_{k=j}^{j+m_i-1} a_{ik} \le M$ and $s_{ij} \le M$ hold for all variables that satisfy other constrains in the problem.

The complete IP model that consists of constraints and the objective function looks like:

$$MAX\left(\text{MIN } R_j = C - \sum_{i=1}^{T} a_{ij} c_i - D_j \right)$$

$$\begin{aligned} s.t. \quad & \sum_{j=1}^{52} s_{ij} = 1 \\ & \sum_{j=1}^{52} a_{ij} = m_i \\ & s_{ij} \le M y_{ij} \\ & m_i - \sum_{k=j}^{j+m_i-1} a_{ik} \le M\left(1 - y_{ij}\right) \\ & s_{ij}; a_{ij}; y_{ij} = 0 \text{ or } 1 \quad \forall\ i, j \end{aligned}$$

At this moment, the company has more than one thousand tanks in operation, and each tank has its own capacity and requires a different maintenance period. On the basis of predicted maximum loads for the next 52 weeks (see Table 1), and the total capacity of 12,000,000 kiloliters, we apply the above IP model to find the maintenance schedule for 5 tanks, 10 tanks, 20 tanks, and 50 tanks. For confidentiality, we only show the data for the ten-tank case in Table 2. The model was run using GAMS/CPLEX (GAMS/CPLEX, 2002) in PC/Windows 2000 Professional with P4/2.2GHz CPU and 768 megabytes RAM, and has found the optimal solution for the 4 situations that is to allocate maintenance time to tanks and maintain a net reserve of 9,875,000 kiloliters.

Table 1 The maximum loads in a year

Week	1	2	3	4	5	6
Maximum loads in kiloliters	1664516	1941935	1941935	1941935	2020391	2125000
Week	7	8	9	…	51	52
Maximum loads in kiloliters	2125000	2125000	2036866	…	1896774	1896774

Table 2 The capacity and maintenance weeks in the 10-tank case

Tank No.	1	2	3	4	5	6	7	8	9	10
Capacity (kiloliters)	1000	2400	1000	5000	12000	9867	14000	15000	15000	25000
Maintenance (weeks)	3	3	5	5	5	8	8	8	10	10

Table 3 illustrates the execution time needed by the IP model, which indicates clearly that it has grown exponentially as the number of tank increases, and the three hours of computation time for 50 tanks is becoming unacceptable for management. Despite the fact that IP does find optimal solutions, the steep increase of computation time presents a grave concern to the management, because IP appears not capable of dealing with more realistic number of tanks within a reasonable time frame. In addition, most key data are more of random nature than certainty; hence more computer runs may be needed for adjusting tank maintenance schedule during the course as more accurate information becomes available. As a result, the IP approach may not be able to satisfy the needs of management that requires both good results and efficient computation time.

Table 3 The execution time for the IP approach

Number of tanks	5	10	20	50
Execution time (seconds)	0.33	0.66	7.58	10056.75

The Genetic-Based Heuristic Optimization Algorithm

GA has been successfully applied to solve a variety of complicated problems in search, optimization, and scheduling (Goldberg, 1989; Jain & Meeran, 1998), however, it is known that GA could fall into the non-convergence situations in some problems (Rudolph, 1994; Ting, Li & Lee, 2002). This problem is largely due to the genetic operations in crossover and mutation that bring in diversity to a new population. The fitness of GA improves rapidly in the beginning stages but slows down considerably in the later stages due to too much diversity. GASA was developed to overcome this deficiency, and could improve the convergence of GAs by using a feature of simulated annealing (SA) (Sun, Dayhoff & Weigand, 1994) that is another stochastic optimization technique. In the following, we will introduce the GASA approach in detail that should also cover GA and SA.

1. Genetic-annealing Approach (GASA)

GASA is a combination of genetic algorithms and simulated annealing approach. Genetic algorithms are stochastic and population-based search algorithms that determine the locations and values of a set of points in the domain space. The criterion for which new points are generated or old points are discarded is a function of the existing population. In genetic algorithms (Goldberg, 1989), individuals encode a set of decision variables by concatenating them in a bit string according to their fitness in a manner similar to the way Nature uses chromosomes. The initial population is generated randomly and the population size is kept constant throughout the process. The fitness f_i of individual i is evaluated based on an objective function. Only individuals with fitness larger than $\bar{f}$, $\bar{f}=\frac{1}{n}\sum_{i=1}^{n} f_i$, are reproduced for the next generation. The effect of this reproduction scheme is that only above average individuals are reproduced to replace poorly performing individuals. There are two kinds of genetic operations that are applied to reproduce individuals: crossover and mutation. First, individuals are paired at random for crossover. For example, the two individuals [0 0 | 0 1 0] and [1 0 | 1 0 1] are crossed-over at the location "|". After recombination, their offspring strings will be [0 0 1 0 1] and [1 0 0 1 0]. Next, the mutation operator is applied to form individuals by occasionally flipping random bits (genes) in the population to prevent the search from being trapped on a local optimum (Holland, 1975).

The other technique used in GASA is simulated annealing (Kirkpatrick, Gelatt & Vecchi, 1983) which is another well-known heuristic algorithm. It is based on the Boltzmann distribution in statistical mechanics. This method was developed for the problems with non-convex objective functions, and is based on the analogy of the physical process of annealing, which is a process for reducing the temperature of a material to a state with minimum energy. In applications, its basic feature is the possibility of exploring the solution space of the optimization problem by allowing non-improving moves. The approach of simulated annealing is performed as follows. Given an initial temperature *t*, the energy *f* of the initial solution (as the current solution) is evaluated. In the following, makes a small random move in the current solution and evaluates its energy. If the energy is improved (reduced), the move is retained as the current solution otherwise the move is accepted with probability determined by Boltzmann probability distribution $p = \exp(\frac{-\Delta f}{t})$. After this move, reduces the temperature *t* according to an annealing schedule and iterates the procedure of random-move, acceptance justification, and temperature reduction until an equilibrium is reached. A simple annealing schedule can be defined as $t' = \alpha t$, where t' is the new temperature and α is the decrease factor between 0 and 1.

The hybrid approach GASA combines features of both algorithms and is based on the concept of "division" by energy. This new approach was designed to rectify the problem with GA by reducing the diversity of the new population in the later stages of search, which is carried out by performing crossover and mutation operation only on chosen individuals. Based on the fitness of individuals, the Component 2 of SA is applied to classify the group of individuals according to their energy (objective values). High-energy individuals will conduct crossover and mutation operations, whereas low-energy individuals are considered more stable and therefore will not undergo these two operations. In such way, the diversity of search is controlled and the final global optimum of the objective function can be achieved in fewer generations.

The design of GASA is illustrated in Table 4; see (Sun, Dayhoff & Weigand,

1994) for details.

Table 4 The pseudo code of the GASA algorithm

```
Generate randomly initial individuals in population $P(0)$;
For $k = 1 : N$        /* $N$ is the number of temperature moves to attempt*/
  $t=1$;                                        /* initial temperature */
For $g = 0 : G_{max}$      /* $G_{max}$ is the maximal number of generation */
Evaluate fitness values of all individuals in $P(g)$
For each individual $i$ in $P(g)$
        Evaluate the energy change $(f_i - f^*)$ and set $p_i$;
        If $p_i >$ random [0,1]
          Keep individual $i$ in pool $P_c$;
        Else
          Put individual $i$ into $P_{nc}$;
        End_If
     End_For
     $P_{nc}$ = Select $(P_{nc})$;
     $P_{nc}$ = Crossover $(P_{nc})$;
$P(g) = P_c + P_{nc}$;
For each individual $i$ in $P(g)$
        Evaluate the energy change $(f_i - f^*)$ and set $p_i$;
        If $p_i >$ random [0,1]
          Keep individual $i$ in gene pool $P_m$;
        Else
          Put individual $i$ into $P_{nm}$;
        End_If
     End_For
     $P_{nm}$ = Mutate $(P_{nm})$;
$P(g) = P_m + P_{nm}$;
$g = g + 1$;
$t = \alpha * t$;
  End_For
$k = k + 1$;
End_For
```

In Table 4, an initial population *P(0)* is created randomly, then the population of the *g*-th generation *P(g)* is selected according to the fitness function *f*. After that, the Boltzmann-based probability $p_i = \exp\left(\frac{f_i - f^*}{t}\right)$ will determine the qualified individuals that are to be retained in the gene pool, and those unqualified ones will be crossed-over. f^* is the optimal fitness value in a population. Agaii· the procedure is repeated by the mutation operation. Following a decrease in temperature ($t = \alpha * t$), the new population will experience the same procedures until the stopping criterion is met, and α is the decreasing factor for temperature that is usually a real number between 0 and 1. The outermost loop is controlled by a parameter, *N*, the number of temperature moves to attempt, which is application-dependently determined (Sun, Dayhoff & Weigand, 1994). Upon completing the genetic operations, all the individuals are placed back into the whole gene pool for the next round of selection.

2. Problem Representation

In applying the genetic algorithm, we use a 52 bits (genes) length string, one bit for a week, to represent the information of maintenance schedule of a storage tank, so that the weeks of outage for that particular tank can be so indicated. Because of the fact that the maintenance once started cannot be terminated half way, hence the weeks of outage are shown as successive bits. A specific tank, within the 52 weeks of scheduling period, can have a number of possible maintenance schedules that are represented by different variations of bits. Figure 1 shows the 43 possible schedules for a tank that requires 10 weeks of maintenance in 52 weeks, and a problem with *n* tanks will need 52 times *n* bits to represent just one possible combination string for the scheduling *n* tanks, with each tank being encoded as one of its possible variation strings. A complete encoding example of scheduling a 10-tank problem, given in Table 2, is provided in Table 5, where the genes of tank 1 indicate its maintenance being scheduled from week 4 thru 6, and week 2 thru 4 for the second tank and so on.

01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
· · · · · · · ·																																																		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Figure 1 The 43 possible variations for a tank with 10 weeks of maintenance

Table 5 Example of gene encoding for 10 tanks

Tank No.	1	2	···	9	10
Genes	00011100000···0	01110000000···0	···	11111111110···0	01111111111···0

On the basis of the chromosome representation, any disturbance in certain gene caused by crossover or mutation represents a reorganized maintenance schedule for the corresponding tank. The design of genetic operators has a profound influence upon the performance of a GA model. We applied the popular one-points crossover operation in this study, that is, the selected parents exchange the genes on the right-hand side of the cutting point. The mutation operator is designed to randomly change one of the genes. The altered genes must also obey the major constraint that the maintenance of a tank cannot be aborted half way. To ensure the validity of genetic operations, the cutting point is located at the boundary of two contiguous 52-bit. Figure 2 shows an example of crossover operation, which recombines two valid substrings from parents. Figure 3 demonstrates a mutation example in which the second tank was randomly replaced by its 50 possible variations. In such a way, the constraints on the number and sequence of maintenance periods for all tanks can be enforced.

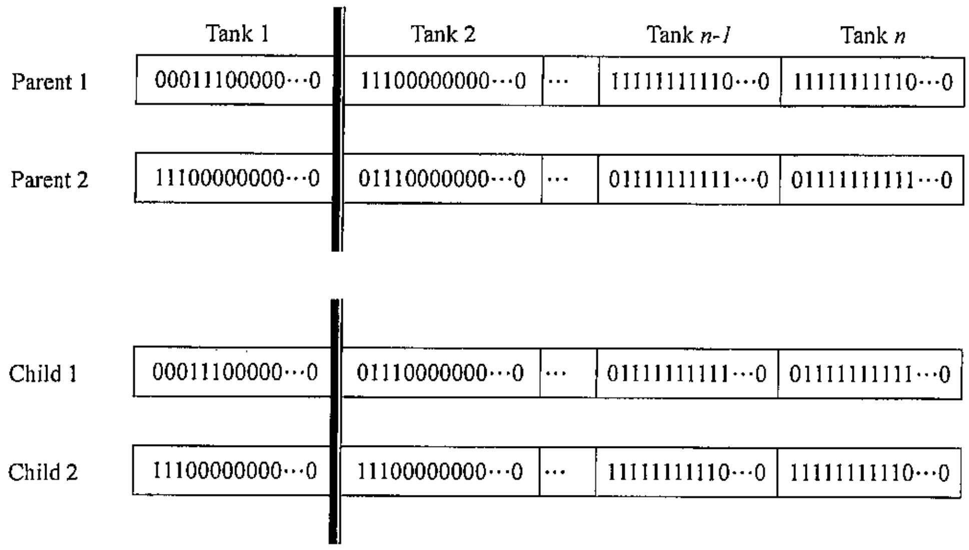


Figure 2 Example of crossover operation

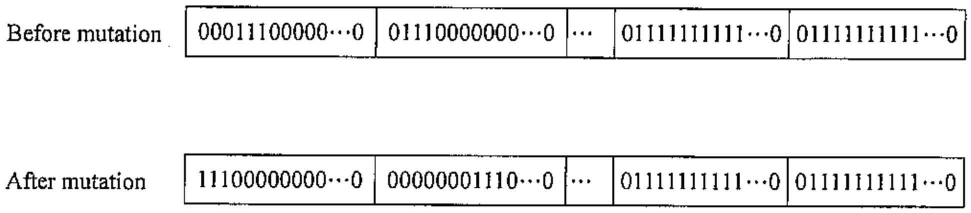


Figure 3 Example of mutation operation

The fitness function of GA can greatly influence its performances; thus, one must identify an effective function to evaluate the level of advantage that a chromosome possesses. To achieve this, we use the objective function to evaluate the merit of chromosome of this problem that is the minimum net reserve of 52 weeks as shown in IP model of Section 2.

Scheduling Results With Ga And Gasa

In order to investigate the performances of both GA and the GASA heuristic optimization algorithm, we conduct a case study with the same data as those for IP study above, and further investigate the feasibility of scheduling 100 tanks using both methods. The GA and GASA algorithms are implemented using Matlab 6.1 on PC/Windows 2000 Professional. Thus, there are five sets of storage tanks: 5, 10, 20, 50, and 100. The population is randomly generated every time with the crossover rate being set to 0.8 and mutation rate 0.01. In GASA, the temperature in each iteration is initialized to be 1 and its cooling factor α is set to be 0.85.

For problems with 5, 10, and 20 tanks, both GA and GASA achieve the same fitness 9,875,000 kiloliters very efficiently as shown in Tables 6 and 7; this is the optimal solution found with IP above. Table 7 presents execution time of GA and GASA with different sizes of tank and populations. From this table, it is very clear that GA and GASA need much less CPU time than the IP approach. For the more complicated 50-tank case, the fitness of both methods are only slightly worse (0.3%) than that found with IP, they are very good results, and they require only a small fraction (1/30) of the CPU times of IP. Even for the most complicated 100-tank case, GA and GASA can still obtain a near optimal fitness (98.4% of the optimal fitness, found by IP) within a reasonable duration 669.57 and 676.00 seconds individually. These results demonstrated the superiority of GA-based approach in dealing with complicated tank scheduling problems in solution quality and execution time.

Table 6 The fitness comparison of GA and GASA

pop. size / tanks	20		50		100	
	GA	GASA	GA	GASA	GA	GASA
5	9875000	9875000	9875000	9875000	9875000	9875000
10	9875000	9875000	9875000	9875000	9875000	9875000
20	9875000	9875000	9875000	9875000	9875000	9875000
50	9812000	9831040	9828000	9835807	9845874	9848500
100	9686233	9695823	9692404	9704124	9714186	9715133

Table 7 The comparison of execution time in seconds of GA and GASA

tanks / pop. size	5		10		20		50		100	
	GA	GASA	GA	GASA	GA	GASA	GA	GASA	GA	GASA
20	7.65	8.12	14.40	14.68	27.42	28.19	69.31	68.21	133.63	136.30
50	18.86	19.45	35.58	36.38	27.79	27.95	168.83	170.96	338.67	341.80
100	37.54	37.80	70.57	71.48	137.88	139.04	336.37	340.57	669.57	676.00

We further investigate the performances of GA and GASA. For the smaller problems of 5, 10, and 20 tanks, the results show an interesting coincidence, which is the same fitness 9875000 kiloliters achieved by both GA and GASA no matter what population size is chosen. This is a very unusual outcome. We discovered that the model was able to identify week 6, 7, and 8 as having loads much higher than the rest in a year and avoid scheduling tank maintenance during the three weeks, as shown in Tables 1 and 2. Their net reserves in these weeks will be lower than the remaining 49 weeks even without maintenance schedule. Hence, for smaller number of tanks, the resulting fitness that is to maximize the minimum reserves will always be the same.

Figures 4 and 5 illustrate the maintenance scheduling with 10 and 20 tanks using GASA with population size 100. Both achieve optimal fitness value 9875000. The number of weeks for maintenance of a specific tank that is showing in Figure 4 is the same as that given in Table 2.

For more complicated problems such as 50 and 100 tanks, we conduct a comprehensive comparison between the two algorithms. Figures 6 and 7 (a)~(c) depict the convergence with different population sizes for GA and GASA in the 50- and 100-tank scheduling problems, respectively.

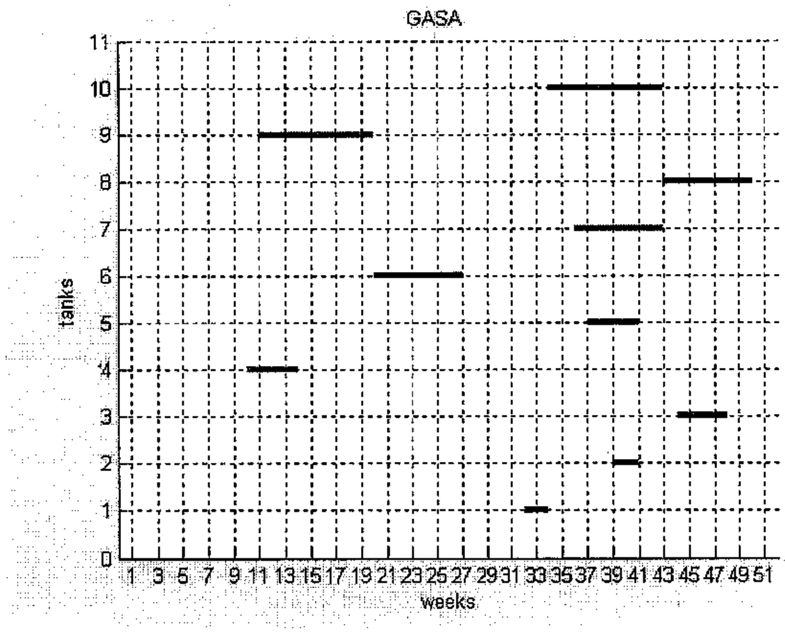


Figure 4　10-tank scheduling using GASA

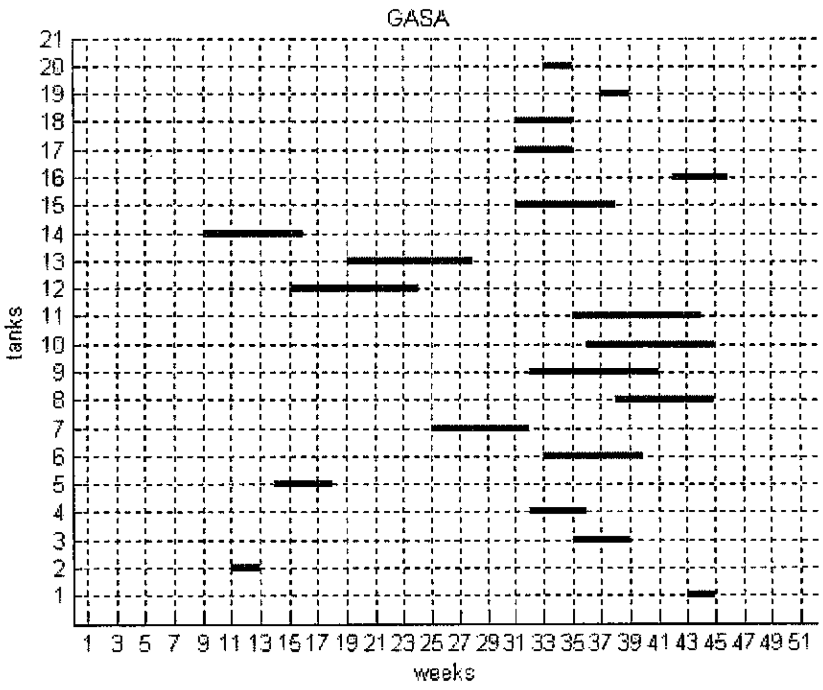


Figure 5　20-tank scheduling using GASA

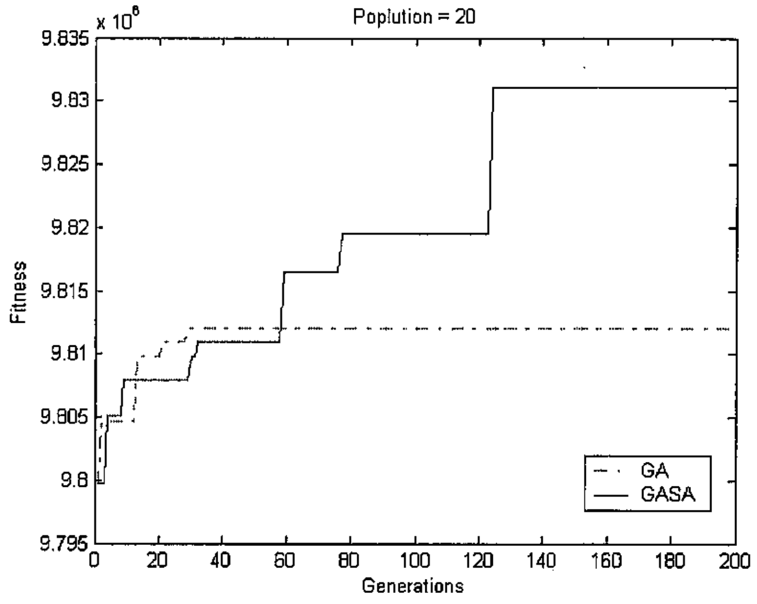


Figure 6 (a) The learning curves with population size 20 in the 50-tank scheduling problem

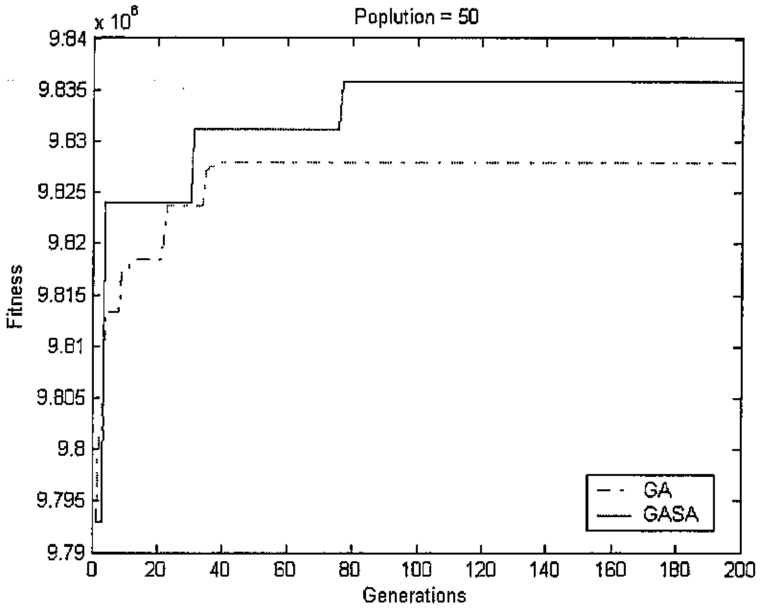


Figure 6 (b) The learning curves with population size 50 in the 50-tank scheduling problem

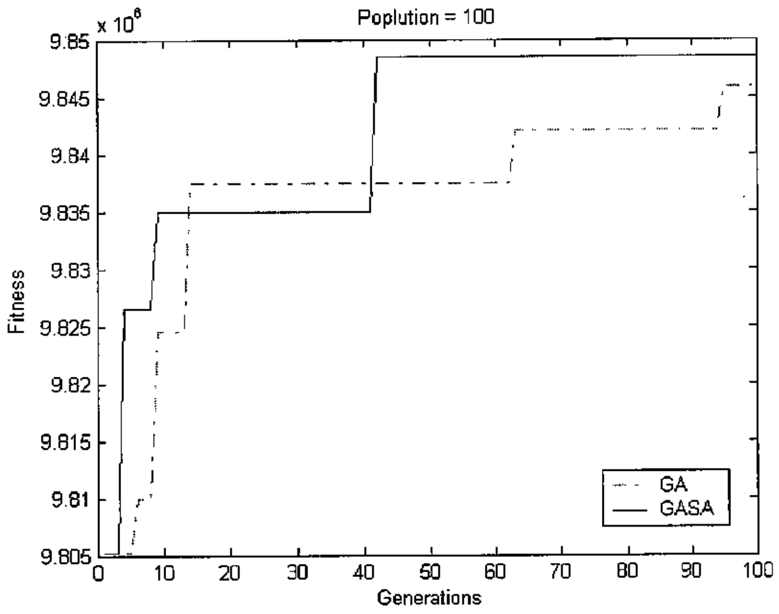


Figure 6 (c) The learning curves with population size100 in the 50-tank scheduling problem

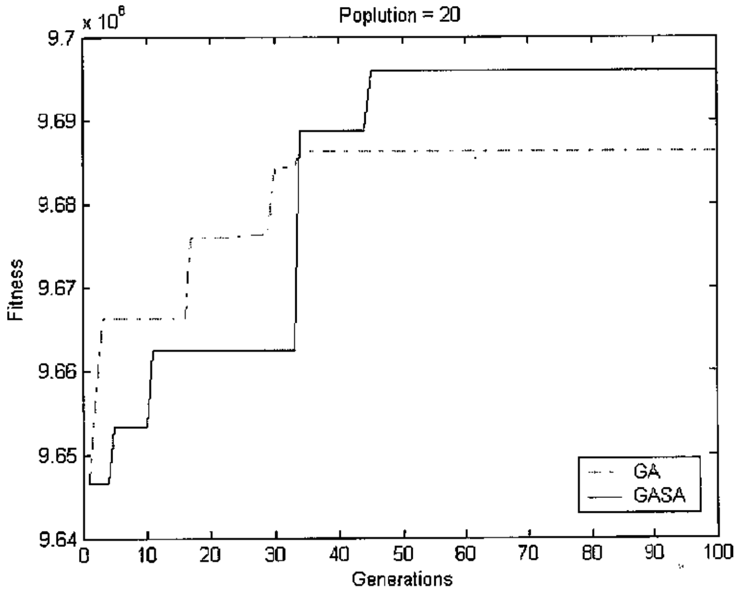


Figure 7 (a) The learning curves with population size 20 in the 100-tank scheduling problem

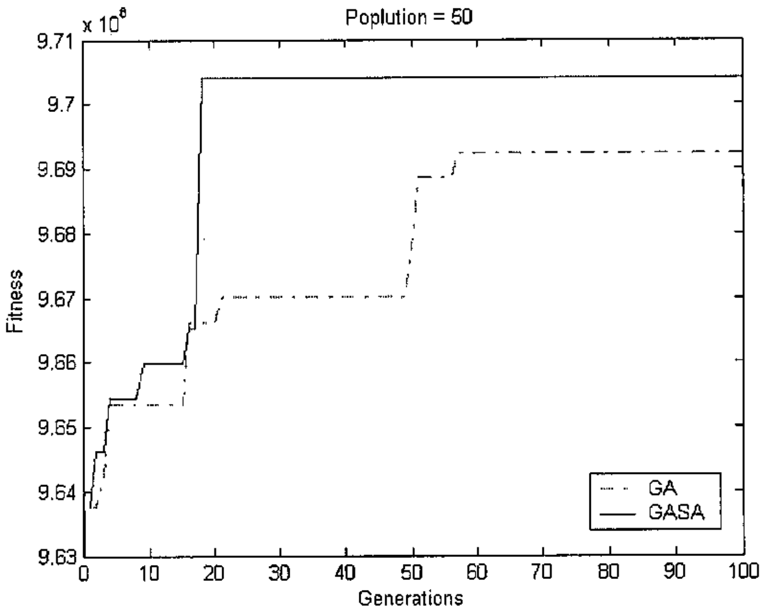


Figure 7 (b) The learning curves with population size 50 in the 100-tank scheduling problem

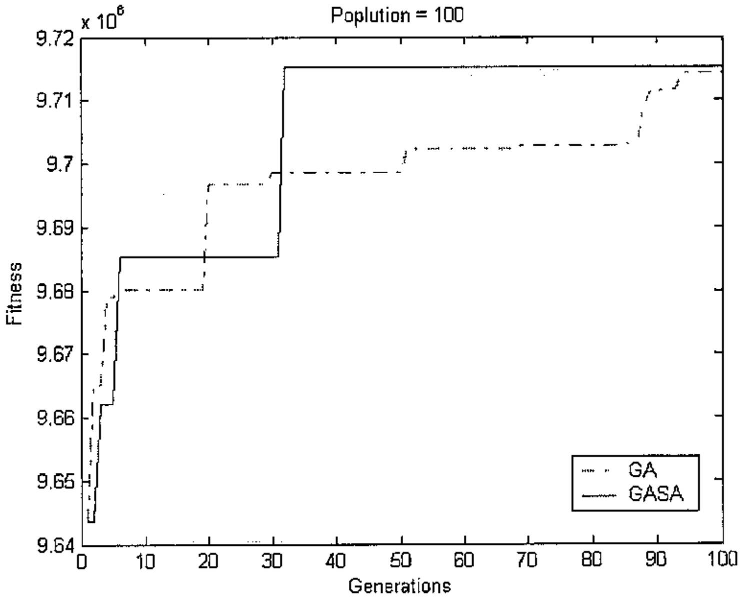
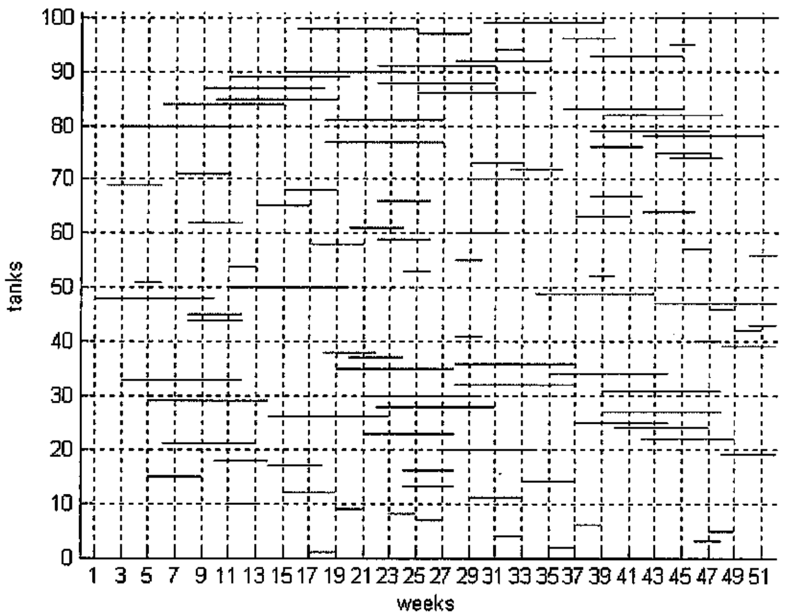


Figure 7 (c) The learning curves with population size 100 in the 100-tank scheduling problem

The above results show that larger population sizes provide better diversity and will result in better solutions for GA and GASA. They both obtain very good solutions, and GASA could clearly achieve better solution quality than GA. Another feature for GASA is the convergence speed; GASA can converge faster to reach the best schedule in terms of generations than GA, which is particularly obvious in the case of larger population sizes. This demonstrates the superiority of GASA in dealing with more complicated problems than GA. Figure 8 shows the scheduling result of 100-tank problem using GASA with population size 100. It can be seen that more concurrent maintenance can be scheduled and the need to schedule 6th thru 8th weeks as well.



Fgiure 8 The scheduling of 100-tank problem using GASA with population size 100

Conclusions

This paper investigates the performances of the genetic-annealing approach, GASA, for planning oil tank maintenance schedule to achieve the best minimum reserve for a petroleum company in Taiwan. The main features of GASA is the application of adaptation and parallelism in GA, and the use of annealing technique to identify the inferior individuals and force them to undergo genetic operations. As a result, GASA becomes a multiple point search technique that examines a set of solutions in parallel, and the stochastic nature of the algorithm helps the search escape the local minima traps. With a set of real data, we conduct experiments with IP, GA and GASA for comparison. IP method finds optimal solutions with expensive computation time, and the later prevents it from being applied to large size problems. Both GA and GASA can achieve very good results, optimal results with small sizes of problems, with much better computation efficiency. GASA, with the help of annealing technique that filters out inferior individuals before applying genetic operations, outperforms GA in terms of solution quality and convergence efficiency. Undoubtedly, the proposed scheduling algorithm GASA can better utilize tank facilities and produce better revenue for existing companies, which are facing a steep competition environment after the introduction of the liberalization laws.

While the performances of GASA have been satisfactory for up to 100 tanks, it is not known if its performances will be acceptable for larger number of tanks, hence future work should continue to explore the feasibility of applying GASA to handle more tanks, perhaps up to 1000, so that petroleum companies can decide if it can be adapted for only localized scheduling or national scheduling. Another important issue that should be pursued in the future is the problem of planning horizon, which, in practice, should be continuous year after year, rather than limited to 52-week, as is done in this study. The present study may create the discontinuity problem between two consecutive schedules, which should not be present in actual planning. One can, of course, extend the horizon by increasing the number of weeks, the present method will then require a longer string, more bits, to represent each tank, and that may end up requiring substantial amount of computation time than is allowed in actual planning.

Acknoledgments

S.-T. Li and L-.Y. Shue were supported in part by NSC90-2416-H-327-012 and

NSC90-2416-H-327-011, respectively. The authors would like to thank R. L. Sun for his advise on the GASA algorithm and Pi-Yu Hsiao for her work on the implementation of GA and GASA experiments.

References

Deris, S. et. al. 1999. Ship maintenance scheduling by genetic algorithm and constraint-based reasoning. ***European Journal of Operational Research***, 112: 489-502.

Dopazo, T. F., & Merrill, H. M. 1975. Optimal generator maintenance scheduling using integer programming. ***IEEE Transactions on Power Applaratus Systems***, 94: 1537-1545.

Jain, S., & Meeran, S. 1998. ***A state-of-the-art review of job-shop scheduling techniques***. Technical report, Department of Applied Physics, Electronic and Mechanical Engineering, University of Dundee, Dundee, Scotland.

GAMS/CPLEX 2002, http://www.gams.com/, accessed at Dec. 5, 2002.

Gen, M., & Cheng, R. 1997. ***Genetic algorithms and engineering design***. A Wiley-Interscience Publication.

Goldberg, D. E. 1989. ***Genetic algorithm in search, optimization, and machine learning***. Addison-Wesley Publishing Company Inc.

Holland, J. H. 1975, ***Adaption in natural and artificial systems***. Michigan: University of Michigan Press.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. 1983. Optimization by simulation annealing. ***Science***, 220: 671-680.

Kralj, B., & Rajakovic, N. 1994. Multiobjective programming in power system optimization: new approach to generator maintenance scheduling. ***International Journal of Electrical Power & Energy Systems***, 16: 24-32.

Lapa, C., Pereira, C., & Mol, A. 2000. Maximization of a nuclear system availability through maintenance scheduling optimization using a genetic algorithm. ***Nuclear Engineering and Design***, 196: 219 – 231.

Man, K. F., Tang, K. S., & Kwong, S. 1999. ***Genetic algorithms***. London: Springer-Verlag.

Mukerji, R., & Parker, J. H. 1991. Power plant maintenance scheduling: optimization economics and reliability. ***IEEE Transactions on PWRS***, 6.

Negnevitsky, M. & Kelareva, G. 1999. Application of genetic algorithms for maintenance scheduling in power systems. ***Proceedings of IEEE International Conference in Neural Information Processing***: 447-452.

Rudolph, G. 1994. Convergence analysis of canonical genetic algorithms. ***IEEE Transactions on Neural Networks***, 5: 96-101.

Storer, R. H., Wu, S. D., & Park, I. 1993. Genetic algorithm in problem space for sequencing problems. In G. Fandel, T. Gulledge, & A. Jones (Eds.), ***Operations Research in Production Planning and Control: Proceedings of a Joint US/German Conference:*** 584-597. Springer Verlag, Berlin, Heidelberg.

Sun, R. L., Dayhoff, J. E., & Weigand, W. A. 1994. ***A population-based search through thermodynamic operation.*** Technical research report TR 94-78, Institute for Systems Research, University of Maryland at College Park.

Ting, C. K., Li, S. T., & Lee, C. N. 2002. On the harmonious mating strategy through tabu search. ***accepted to be published in Information Sciences: An International Journal.***

Wang, Y., & Handschin, E. 2000. A new genetic algorithm for preventive unit maintenance scheduling of power systems. ***International Journal of Electrical Power & Energy Systems***, 22: 343-348.