Proceedings of the 9th WSEAS International Conference on Applied Mathematics, Istanbul, Turkey, May 27-29, 2006 (pp205-213)

An Improved Genetic Algorithm for the Dynamic Cargo Crew Pairing Problem

SHAW CHING CHANG
Information Engineering Institute
Institute for Information Industry
No. 18F, No.333, Sec.2, DunHua South Road, Taipei
TAIWAN, R.O.C.
http://www.geocities.com/kenny_chang22

Abstract: - Dynamic factors of crew pairing problem can make it more realistic. A stable algorithm without parameter adjustment is important for the dynamic crew-pairing problem as well as the schedule operator. The available seats for deadhead trips become the main dynamic factor of the cargo crew-pairing problem. Since it is the one of the factors hard to be controlled by the traditional crew- pairing problem. An improved genetic algorithm for solving this dynamic cargo crew-pairing problem has been developed in this paper. The test data is the real scenario of an international airline in Taiwan. The result shows that the algorithm is more advantageous than the existing technology, either in the cost or in the performance of generating the solution.

Key-Words: - Dynamic Crew Pairing Problem, Genetic Algorithm

1. Introduction

From the basic flight sector, duty, pairing to the roster of the crew, this is actually a four-layered hierarchy of the formation of the crew schedule. The crew-pairing problem is the first process of the crew-scheduling problem. It generates the duties from flight sectors and the pairings from duties. Thus a pairing is the combination of several duties. The following process is the crew rostering problem, which assigns the pairings to crews based on the result of crew pairing problem. This paper considers only the crew-pairing problem and assumes the duties are predetermined as well as the number of required crews for the duties. The schedule operator of an international airline also puts his main effort on the pairing generating process since most of the duties contain only one flight sector.

The dynamic factors of cargo crew pairing problem are the number of pilots in the duty, the number of unoccupied seats in a freighter and the number of flights. First, crew-pairing problem traditionally ignores the differential number of crews between duties for the reason of simplifying the problem. It was assumed that a duty only consists of a single flight sector before. Usually the maximum number of crews in a duty is four, which is designed for a long flight. The minimum number is two, which is for a short flight. Three pilots are allocated for a medium flight. Each pilot in a duty corresponds to a rank of crew. The assignment of different ranks of crews containing in a duty is for the reason of safety operation. So the crew-paring problem becomes to the crew-paring problem for every rank of crew solving rank by rank. To simplify the problem, this paper takes this factor as a given condition. Second, the freighter is mainly used for the cargo carriage. There are no seats in the cabin. Thus the seats in the cockpit limit from 4 to 8 depending on the type of aircraft. For example, the number of seats in MD-11F is four. If a duty requires three pilots on that plane, then there is only one seat left for a deadhead trip. When this aircraft hands over to the other three pilots, this will cause the original three pilots to take different aircraft to travel as a passenger and result to operate with three different pairings since they must be moved to the other airports to continue their duties. Thus the main difference between the cargo flights and the passenger flights is the shortage of the seats for a deadhead trip in a freighter with limited seats. In this paper, this is the key to make this problem as a dynamic problem. Once the pairings of one crew type has been determined, the cost of the pairings of the other crew type will be affected by the utilization of the seats in the previous crew-pairing stage. Third, the demand of goods varies from time to time. The schedule of cargo flight therefore has its fundamental uncertainty and the variety of the number of flights between every month. In practical condition, this factor can be treated as a given data

for determining the crew pairings month by month. The above features form the traditional stationary crew-pairing problem as a dynamic cargo crew-pairing problem.

Traditional stationary crew pairing problem is usually formulated as a set-partitioning problem proposed by Barnhart [3]. It is expressed as follows,

$$\begin{aligned} &\min \sum_{p\in P} c_p y_p \\ &st.1 \sum_{p} y_p = 1 \quad i \in F \\ &y_p \in \{0,1\} \quad p \in P \end{aligned} \qquad \text{.....(1)}$$

where y_p equals to 1 if pairing p is in the solution, and 0 otherwise. F is the set of flight segments and P is the set of pairings. A column p has a 1 in row i if the flight i is flown by pairing p, and c_p is the cost of pairing p. This model guarantees each flight sector is covered once and that the total cost is minimized. There are many approaches to solve this problem, including Branch and Bound method (see Ernst [12] and Anbil [2], for example), column generation (see Anbil [1]; Crainic and Rousseau [9], Desrosiers and L¨ubbecke [10], Lavoie [18], for example), hybrid method of combining Branch and Bound and column generation (see Barnhart [4], Desaulniers [11] and Freling [14], for example), network flow models (see Yan and Tu [24], Guo [16] and Mellouli [20][21], for example), simulated annealing algorithm (see Emden-Weinert and Proksch[13]), tabu search (see Cavique [8], for example) and genetic algorithm (see Beasley and Chu [5], Levine [19], Ozdemir and Mohan [22], Chang [7], for example)

Based on the eq. (1), the dynamic cargo crew-pairing problem in general can be revised as follows,

$$\begin{aligned} &\min \sum_{p\in P} c_p^t y_p^t \quad t \in T \\ &st.1 \ \sum_{p} y_p^t \geq 1 \quad i \in F^t \\ &st.2 \ \sum_{t}\sum_{p} y_p^t \leq F_i^C \quad i \in F^t \\ &y_p^t \in \{0,1\} \quad p \in P^t \end{aligned} \qquad \text{.....(2)}$$

where this equation introduces the phase factor, t, which stands for the four types of crews and the scheduling period of this problem. For example, if there are three months of flight to be scheduled, then t will be expressed by a time series set from one to twelve. The first four numbers stand for the scheduling sequences of the four types of crew in the first month. The following four numbers, i.e., from 5 to 8, stand for the next month and the last four numbers are for the last month. However, to keep this problem a moderate size to solve, the t in this paper stands for 4 numbers at a time. Constraint st.1 allows the assignment of deadhead trips. A new constraint, st.2, is to ensure that in all phase the summation of assigned crew in flight i will not exceed the seat capacity of flight i, F_i^c. The constraint st.2 will enlarge the problem size of eq. (1). Also in the real world, there are still many other regulations to be added in eq. (2) as stated in the following section. It will be hard to use the traditional approach to proof the existence of the global optimum for the dynamic problem time after time and solve that problem in an acceptable time period as Branke [6] stated in his paper that

"If the optimization problem is dynamic, the goal is no longer to find the extremum, but to track their progression through the space as closely as possible."

The genetic algorithm, GA, based heuristic algorithm, can provide flexibility in handling the dynamic factors and the variations of the model such as constraints in cumulative flight time, mandatory rest periods, or limits in the amount of work allocated to a particular base by modifying the evaluation function. Most traditional methods may have trouble accommodating the addition of new constraint as easily. GA works directly with integer solutions. There is no need to solve the LP relaxation. It can also be run as a model combining the simulation and optimisation to take the dynamic factors into consideration. Its main flaw is the time of convergence. Two manoeuvres are also developed to improve the performance of the conventional genetic algorithm. The test case in this paper considers for a real world application of an international airline in Taiwan.

2. Problem Formulation

To enhance the performance of problem solving, this paper separates the routes into few uncorrelated areas, such as Pan-American area and Pan-European area. The pan-American area, AMS, includes the flights from pacific Asia to American. The pan-European area, EUR, includes the flights from pacific Asia, mid east of Asia and to Europe. The seat capacity of the cockpit of the freighter allows only for four pilots at a time. The test data used in this paper was the real scenario in August of 2004. In case of that some deadhead trips are cheaper or

faster than taken the cargo flights. The alternative flights for deadhead trip include not only the cargo flights but also the passenger flights. The other information about the flights and the flight for the deadhead trips of the test data is summarized in the following Table 1.

Table 1. Characteristics of test cases

Test data of month	August	
Items\Area	AMS	EUR
Number of Flights	300	476
Number of Airports	7	18
Average Frequency	43	26
Number of Hotels	34	
Number of alternative flights for deadhead trip	7022	

2.1. The problem objectives

The followings are 5 types of objectives considered in this paper.

1. The amount of the required deadhead trips in the pairing generating process, pnc_counts. The assumption is that at most two deadhead trips can be applied to connect either from the base to the airport or from the airport to the base. At most one deadhead trip can be applied to connect between two non-base airports.
2. The total flight time of all deadhead trips in the solution, pnc_time. This constraint can make the choice of deadhead trips prefer to a shorter flight.
3. The hours of the generated pairings, Perdiem, which uses the unit of hour to express the duration of all pairings.
4. The days of the generated pairings, Manday, which uses the unit of day to express the duration of all pairings.
5. The maximum length of a pairing, PMD. It checks if the pairing exceeds the predetermined threshold.

The objective function defined here is a bit different with the one in the result comparison with the experienced operator. The objective function is used for finding the correct direction of the solution. The actual cost of the crew pairings will be calculated while comparing to the experienced operator. The objective function is defined as follows,

$$
\begin{aligned}
&c(x) = \sum_{t=1}^{4} \sum_{p=1}^{\alpha} PC_p^t(x), \\
&PC_p^t(x) = pnc\_counts_p^t(x) \times W\_pnc\_counts \\
&\quad + pnc\_time_p^t(x) \times W\_pnc\_time \\
&\quad + Perdiem_p^t(x) \times W\_Perdiem \\
&\quad + Manday_p^t(x) \times W\_Manday \\
&\quad + PMD_p^t(x) \times W\_PMD
\end{aligned}
\qquad \text{..... (3)}
$$

where the W_pnc_counts represents the weight of pnc_counts between all other objectives. The W_pnc_time represents the weight of pnc_time between all other objectives. The W_Perdiem represents the weight of Perdiem between all other objectives. The W_Manday represents the weight of Manday between all other objectives. The W_PMD represents the weight of PMD between all other objectives.

2.2. Constraints

There are 8 kinds of constraints considered in this paper. Seven of them are related to the crew pairing generation for each crew type and are defined as followings,

1. The number of misconnection between airports in a solution, pnc_err. It counts those misconnected routes, which cannot be found in the alternative flight lists. In our experience, this constraint can always be satisfied if it reaches a feasible solution. Or the data must be wrong. Because the aircrafts are always flown out from the base and back to the base eventually, so the problem of connection will be only on the frequency of that route of flight.
2. Constraint mrt is to check if the minimum rest time between duties of a pairing is enough. This regulation depends on the length of flight time and flight duty period. The regular rest time is derived from flight time. For example, if the flight time is between 10 and 14 hours, then the regular rest time should have 16 hours. The flight duty period is the period of time from check-in to checkout. If the deadhead trip is connected to the duty within 4 hours, then the calculation of duty time should consider the flight time of this deadhead trip. The minimum rest time will be the longest one between the regular rest time and the flight duty period plus few hours, i.e. max (regular rest time, flight duty period+2 hours). The 4 hours are the rule for judging the independency of the deadhead trip.
3. The pairing should start from the base and return

to the base, base2base. This is the definition of being a pairing.
4. Maximum weekly flight time, 7d32hrs. It checks if the flight time of a pairing within 7 consecutive days is more than 32 hours.
5. Minimum weekly rest time, 7d24hrs. It checks if there exists a period of rest time within 7 consecutive days being at least 24 hours. This constraint is valid only for the duration of pairing being over 7 days.
6. The pilot acclimation constraint, TimeDiff. It is to check if the next rest period within a pairing is enough whenever the time difference of a duty exceeds the regulation.
7. Constraint LongFT verifies that the rest period within a pairing is not enough prior to returning the base whenever the flight time of a duty is over 11 hours.

The last one is the summation of the assignment of the deadhead trips in all crew types cannot exceed the capacity of the seats in a flight. It will be handled in the function of deadhead_quota_calc(p) as shown in Algorithm 1. If any one of them has been violated, then it will be punished by its corresponding penalty. The penalty function for the violation of constraints is formulated as follows,

$$\begin{aligned} V_p^t(x) &= V\_pnc\_err_p^t(x)\times P\_pnc\_err \\ &+ V\_base2base_p^t(x)\times P\_base2base \\ &+ V\_mrt_p^t(x)\times P\_mrt \\ &+ V\_7d32hrs_p^t(x)\times P\_7d32hrs \\ &+ V\_7d24hrs_p^t(x)\times P\_7d24hrs \\ &+ V\_TimeDiff_p^t(x)\times P\_TimeDiff \\ &+ V\_LongFT_p^t(x)\times P\_LongFT \end{aligned} \quad \text{.....(4)}$$

where V_pnc_err is the number of violating the pnc_err constraint. P_pnc_err is its corresponding penalty. V_mrt is the number of violating the mrt constraint. P_mrt is its corresponding penalty. V_base2base is the number of violating the base2base constraint. P_base2base is its corresponding penalty. V_7d32hrs is the number of violating the 7d32hrs constraint. P_7d32hrs is its corresponding penalty. V_7d24hrs is the number of violating the 7d24hrs constraint. P_7d24hrs is its corresponding penalty. V_TimeDiff is the number of violating the TimeDiff constraint. P_TimeDiff is its corresponding penalty. V_LongFT is the number of violating the LongFT constraint. P_LongFT is its corresponding penalty.

3. Problem Solution

The GA is itself an evolutionary and interactive process. Through the optimal process generation by generation, the status of the seats available for a deadhead trip can be updated accordingly. To generate the other set of pairings for the other type of crews in a duty, the model can also consider the continuity of calculating the unoccupied seats in the same flight while generating the duty for different crew types. That is first, or t=1, to generate the pairing for the highest rank of pilot in all duties, i.e. the captain, and calculate the available seats for the next type of crew. The following step, or t=2, is for the other pilot of the two pilot duties, i.e. the first officer, and updating the number of available seats. Third, or t=3, is for the additional pilot of three pilot duties, i.e. the relief captain, and recounting the empty seats again. Finally, or t=4, is for the last pilot of the four pilot duties, i.e. the relief first officer. This indicates that the different rank of crews might have their different pairing.

The following Algorithm 1 shows the flow of the genetic algorithm designed in this paper.

Algorithm 1 Outline of Genetic Algorithm
for all $t \in T$ do
  $g \Leftarrow 0$
  $p \Leftarrow initPopulation()$
  deadhead_quota_calc(P)
  evaluate(P)
  repeat
    $\{p1, p2\} \Leftarrow Selection(P)$
    $p' \Leftarrow Crossover(p1, p2)$
    $Mutation(p')$
    if deadlock occurs then
      $CrossMutation(p', highMutationRate)$
    else
      $CrossMutation(p', lowMutationRate)$
    end if
    deadhead_quota_calc(p')
    evaluate(p')
    popReplacement(p',P)
    $g \Leftarrow g + 1$
  until terminating condition
end for

It starts with the initialisation of the population. After the initialisation, the quota of the deadhead trip for current each crew type is calculated before the main operation of genetic algorithm. Then evaluating the population, selecting the better parents to crossover and breaking the local optima

by mutation. If the good fitness value is not improved for some generations, the cross-mutation will be applied by a higher possibility. The quota of the deadhead trip for current generation is calculated and evaluates the population. Finally the new generated chromosomes replace the old chromosomes. The algorithm stops until the score of the fitness function is not improved for another 1500 generations. The following few sections describe more detail about the algorithm.

3.1. Matrix Encoding Scheme

In this paper a matrix representation is considered rather than an original string list representation from Holland [17], for the following three reasons. The first is the intuitive connection between a chromosome and a set of pairings. It can also maintain the solution space with the original problem. The third is that with the help of the powerful computing machine we can examine the legality of every chromosome very fast. This representation can also eliminate st.1 in eq. (1) by assigning all duties in the chromosome. For the deadhead trip consideration as st.1 in eq (2), if it is necessary it will be assigned as an attribute of the corresponding duty. This approach can reduce the search space of original problem.

	d_1	d_2	d_3	· · · ·	$d_{\beta-1}$	d_β
p_1	w_{11}	w_{12}	w_{13}	· · · ·	$w_{1\beta-1}$	$w_{1\beta}$
p_2	w_{21}	w_{22}	w_{23}	· · · ·	$w_{2\beta-1}$	$w_{2\beta}$
·				·		
·				·		
·				·		
p_α	$w_{\alpha 1}$	$w_{\alpha 2}$	$w_{\alpha 3}$	· · · ·	$w_{\alpha\beta-1}$	$w_{\alpha\beta}$

Fig. 1 Matrix Encoding Scheme

The encoding representation of a chromosome can be seen in Fig. 1. As depicted in this figure, the p_α represents the αth pairing, d_β represents the βth duty in the duty list of a pairing and $w_{\alpha\beta}$ represents the identification number of the βth duty of pairing p_α. The number of pairings, α, will be set as one and half times of the number of duties departed from the base. Fig. 2 illustrates all possible conditions in the chromosome. The number of duties in a pairing, β, is usually set to the largest number of duties that a pairing might contain. Every duty will be assigned a unique and sequentially ordered number for its identification. The flight-covering problem in eq. (1) and eq. (2) will be changed to duty covering problem thereafter.

3.2. Population Initialization

Initialization is a process to generate the initial population, which allows the application of variation operators. A certain number (population size) of individuals, exhibiting equal or similar genome structures, is created either randomly or heuristically. However, in crew scheduling problem if the duties are assigned randomly, the minimum rest time between duties will always be the most difficult constraint to satisfy. To generate a feasible solution, the estimated longest rest time between duties will be assigned in the population initialization, as the following Algorithm 2.

Algorithm 2 Initialization Procedure
Require: affected duties set $D' \neq \phi$, individual s

```
for all d ∈ D' do
  b ⇐ 0
  if d(departure airport) is base then
    assign(d, b, 0)
    D' ⇐ D' − {d}
    b ⇐ b + 1
  end if
end for
for all d ∈ D' do
  if d is not assigned then
    for n=1 to ROW do
      for m=1 to COL do
        rt ⇐ largest_rest_time_of_regulation
        if d(dep. airport) is s[n][m-1](arr. airport) and
          d(dep. time) > s[n][m-1](arr. time+rt) then
          assign(d, n, m)
        else
          assign(d, n, 0)
        end if
        D' ⇐ D' − {d}
      end for
    end for
  end if
end for
```

The first part of the Algorithm 2 is to assign all duties departed with the base, such as w11, wd1 and we1 in the Fig. 2. They all departed from the base, TPE, in this example. The second part is to find if the other duties can be the successors of the duties

assigned in the chromosome. Otherwise it will be assigned as a first duty of a new pairing, such as wf1 and wg1 in Fig. 2. In Fig. 2, pairings from p_1 to p_{d-1} are called feasible pairings. Pairings from p_d to p_{h-1} are called unfeasible pairings. These pairings will be added the necessary deadhead trips to become feasible pairings. In addition, there are some pairings containing nothing, such as pairings from p_h to p_α.

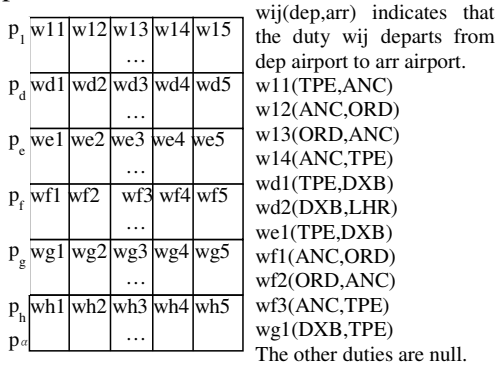

Fig. 2 Example of initial solution

3.3. Crossover Operators

A simple crossover operator, the column-based crossover, implements this function. Its basic idea is to construct new offspring in a way that columns are selected randomly from the chosen parents. This is illustrated in Fig. 3. It operates as the well-known one point crossover.

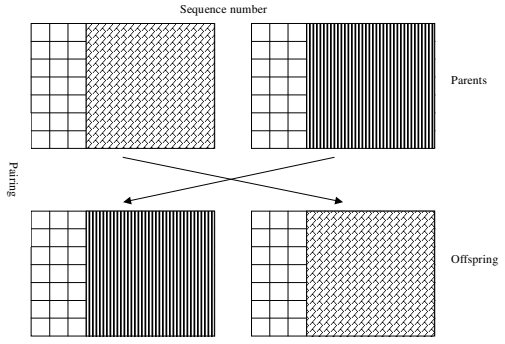

Fig. 3 One-point crossover operation

3.4. Mutation Operators

The intention of mutation operators is to avoid getting trapped in local optima through increasing the diversity of the population. As Branke [6] stated that this applies the idea of maintaining diversity throughout the run to tackle this dynamic problem. It usually modifies the resulting offspring slightly by changing only few values or varying a part of the individual. Here we introduce two mutation operators, which have been applied in our approach. Fig. 4 demonstrates the basic mutation operator. That is to select two cells randomly and swap their content.

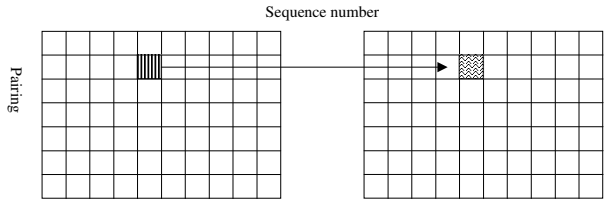

Fig. 4 Basic mutation operator

In order to increase the opportunity of forming different styles of pairings, the second mutation operator, named as cross-mutation, exchanges two chromosomes based on a randomly selected number of duty id. This idea is actually coming from the string based crossover operation. However, if this operation is being triggered very often, then it will have a huge disturbance for the current solution. So it should be triggered as a mutation operation. A random number is selected between the smallest duty ID and the biggest duty ID. Then we exchange those duties with their ID larger than the random number as the illustration in the Fig. 5. The selected random number is 5 in this case. So the duties of parents with their ID being larger than 5 will be replaced by each other as depicted in Fig. 5.

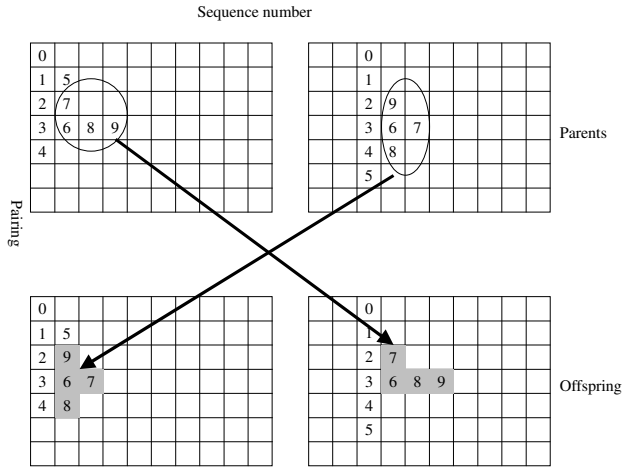

Fig. 5. Illustration of cross-mutation

This operation can make the number of pairing as small as enough by minimizing the deadhead trips. The operation rate of this method will be increased a bit if the score of the result has no improvement for 50 generations.

3.5. Evaluation

Normally, each individual's fitness value calculated

by the evaluation procedure is a number expressing the quality of the solution. Generally speaking, one common way to calculated fitness value of an individual is to combine its cost and the penalty. The cost of an individual can be easily translated from the objective function of the given problem. However, it does not take infeasibility into account and individuals in the population are likely infeasible solutions, hence an appropriate penalty has to be incorporated into the evaluation process. Accordingly, the fitness function f(x) can be of the form,

$$f(x) = c(x) + V(x) \quad \text{.....(5)}$$

where c(x) is the objective function and V(x) is a penalty function which is usually problem specific.

3.6. Selection Method

The selection of parents is the process that provides a change of reproduction to every individual in the population. As described earlier, some types of variation operators, such as crossovers, require two or more parents to produce new offspring. The quality of the resulting offspring may depend on its parents because most parts of the offspring are inherited from them. Because the variance among fitness values can be quite large, the risk that some individuals dominate the whole population after few generations must be taken into account. Hence, proportionate selection, roulette-wheel selection strategies, rather than strategies based on the absolute fitness value seem to be more applicable for this specific problem. The possibility that an individual is selected to do the reproduction is determined by the suggestion of Goldberg [15],

$$P_c = \frac{\left(F / f_c\right)}{\sum_{c=1}^{POP}\left(F / f_c\right)} \quad \text{.....(6)}$$

where POP is the total population number, f_c is the fitness value of the chromosome in population and F is defined as $F = \sum_{c=1}^{POP} f_c$.

3.7. Replacement Strategy

In light of the evolution process, new offspring produced by the GA operators replace individuals in the population. The average fitness of the population then can be improved over generations. The replacement schemes adapted in this paper is close to generational replacement, defined originally by Holland [17]. The entire population is replaced except the best one in the population after a new population is generated. It indicates that the best solution in current generation is always the best one since the algorithm started. This will help the business operators who are not familiar to the kernel of problem solving for monitoring the process easily.

3.8. The result

This case is running by a personal computer with the Pentium-IV 2GHz CPU speed. The performance statistics of this four-stage dynamic problem is shown in Table 2. Normally in an international airline, there will be some long duties required the fourth pilot. However, the more pilots are assigned in the aircraft, the more cost of the airline will spend. In this case, due to the unstable sources of the cargo service, there is no fourth pilot in this month of test data. So there is no such data in Table 2. In the other side, the running time of the algorithm depends on the size of the problem. However, the characteristics of the routes will also have a very important influence on it. The number of flights in EUP area is larger than those in AMS area. So the problem with EUP data should be more complex than AMS theoretically. But the fact is almost on the contrary. The main reason is that the frequency of routes is rather different between two areas. In average, from Table 1 the frequency of the routes in AMS area is about one and half times higher than those in EUP area. So the solution space of the problem with EUP data is approximately 66% of the solution space of the problem with AMS data. In this way the variation of the destination is not as important as frequency.

Table 2. The performance of the algorithm

	Month	August	
	Area	AMS	EUP
Generations	t=1	5892	6243
	t=2	11615	3787
	t=3	4699	3494
	t=4	--	--
Running Time (hh:mm:ss)	Total	2:29:30	1:47:31

The cost indexes in Table 3 considered in this comparison are briefly described as follows.

1. PerDiem, which is calculated as the multiplication of its hourly unit price, HUP, and the time duration from the time of checking in, TI to the time of checking out, TO. It is defined as,

$$\text{PerDiem} = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{HUP} \bullet (\text{TO}_{tp} - \text{TI}_{tp})] \quad \text{.....(7)}$$

2. Hotel cost, which is calculated as the multiplication of room rate, RoomRate, and number of nights, Night. The number of nights depends on the check in policy of every foreign contracted hotel. For example, if the pilot arrives before the time of checking in this hotel, then an extra night of a room is charged. It is defined as,

$$\text{H}otel = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{RoomRate} \bullet Night_{tp}] \quad \text{.....(8)}$$

3. Deadhead trip cost, which is the result of the multiplication of its number of usage in total crew pairings, NDHT, and its corresponding average sales price, CASP. It is defined as,

$$\text{DeadheadTrip} = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{NDHT}_{tp} \bullet C\text{ASP}] \quad \text{.....(9)}$$

4. Pay time cost, which is calculated as the multiplication of the accumulation of the flight time, FT, and half of the flight time of deadhead trips, FTD, in all pairings and its average hourly rate, AHR, of the corresponding crew type. It is defined as,

$$\text{PayTime} = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{AHR}_{t} \bullet (FT_{tp} + \frac{1}{2}\text{FTD}_{tp})] \quad \text{...(10)}$$

5. Flight time, which is calculated as the multiplication of the accumulation of the flight time in all pairings and its average hourly rate of the corresponding crew type. It is defined as,

$$\text{FT} = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{AHR}_{t} \bullet FT_{tp}] \quad \text{.....(11)}$$

6. Man day, which is calculated as the multiplication of the average daily rate, ADR, of the corresponding crew type and the day duration from the day of checking in, DI, to the day of checking out, DO, for all pairings. It is defined as,

$$\text{Manday} = \sum_{t=1}^{4}\sum_{p=1}^{\alpha}[\text{ADR}_{t} \bullet (\text{DO}_{tp} - \text{DI}_{tp} + 1)] \quad \text{.....(12)}$$

COM_Aug, the result of the algorithm in Table 3, is compared with an experienced schedule operator, as there is no existing software considering the dynamic factors. The MAN_Aug is the result of an experienced schedule operator using exactly the same data with the algorithm. He takes about 2 days to generate his result.

Table 3. Cost comparison

Item\source	MAN_Aug	COM_Aug
PerDiem	5,358,427	5,513,227
Hotel cost	10,814,969	11,886,083
Deadhead trip cost	335,401	222,202
Pay Time cost	42,746,856	41,762,440
Flight Time cost	40,903,648	39,296,888
Man Day cost	33,733,356	34,091,508
Total Cost	133,892,656	132,772,352

Table 3 and Table 2 show that the algorithm is more advantageous either in the schedule cost or the performance of generating the schedules. It can save the cost approximately by 1.1 million NTD monthly.

4. Conclusion

The crew-pairing problem in the real world is either dynamic in a changeable environment or complex in the negotiable regulation. The traditional crew-pairing problem can be said that it is the special case of the dynamic crew-pairing problem.

In this paper, the number of unoccupied seats in a freighter cockpit makes the stationary crew-pairing problem dynamic. It is hard to be solved by the traditional approach. Since it cannot be predetermined. A genetic algorithm for dealing with the dynamic cargo crew-pairing problem is developed. The special treatments in population initialisation and in mutation operator make this algorithm more efficient for finding an acceptable result. It has been tested for the data in the real world and compared their results with a senior scheduler. The result shows a tremendous advantage than the result by a scheduler. It can either save the cost approximately for 1.1 million NTD per month or the time in days. Thus, for solving this dynamic problem, this algorithm can also take every dynamic constraints and objective functions very well. Its architecture is very suitable for solving this kind of problem.

References:

[1] Anbil, R., Forrest, J. J. and Pulleyblank, W. R., Column generation and the airline crew pairing

problem, Documenta Mathematica, Extra Volume ICM, *In Proceedings of the International Congress of Mathematicians*, Berlin, 1998, pp. 677–686.

[2] Anbil, R., Tanga, R. and Johnson, E. L., A global approach to crew pairing optimization, *IBM System Journal*, Vol.31, No.1, 1992, pp. 71–78.

[3] Barnhart, C., Cohn, A. M., Johnson, E. L., Klabjan, D., Nemhauser, G. L. and Vance, P. H., Airline crew scheduling. In Hall R.W., editor, *Handbook of Transportation Science,* Kluwer, 1999.

[4] Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P. and Vance, P. H., Branch-and-price: Column generation for solving huge integer programs, *Operations Research*, Vol.46, 1998, pp. 316–329.

[5] Beasley, J. E. and Chu, P. C., *A genetic algorithm for set covering problem*, the management school, Imperial College, 1994.

[6] Branke, J., Evolutionary approaches to dynamic optimization problems: A survey. In J. Branke and T. Back, editors, *Evolutionary Algorithms for Dynamic Optimization Problems*, 1999, pp. 134-137.

[7] Chang, S. C., A New Aircrew-Scheduling Model For Short Haul Routes. *Journal of Air Transport Management,* Vol.8, 2002, pp. 249-260.

[8] Cavique, L., Rego, C. and Themido, I., Subgraph ejection chains and tabu search for the crew scheduling problem, *Journal of the Operational Research Society*, Vol.50, 1999, pp. 608–616.

[9] Crainic, T. G. and Rousseau, J. M., The column generation principle and the airline crew scheduling problem, *INFOR*, Vol.25, No.2, 1987, pp. 136–151.

[10] Desrosiers, J. and L¨ubbecke, M. E., *A primer in column generation,* Technical report, Technische Universit¨at Berlin. At http://www.math.tu-berlin.de/coga/publications /techreports/2003/Report-048-2003.html, 2003.

[11] Desaulniers, G., Desrosiers, J., Ioachim, I., Solomon, M. and Soumis, F., A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, Kluwer Publishing Company, 1998, pp. 57–93.

[12] Ernst, A. T., Jiang, H., Krishnamoorthy, M. and Sier, D., Staff scheduling and rostering: A review of applications, methods and models, *European Journal of Operational Research*, Vol.153, No.1, 2004, pp. 3–27.

[13] Emden-Weinert, T. and Proksch, M., Best practice simulated annealing for the airline crew scheduling problem, *Journal of Heuristics*, Vol.5, No.4, 1999, pp. 419–436.

[14] Freling, R., Lentink, R. M. and Wagelmans, A. P. M., *A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm,* Econometric Institute Report EI 2001-29, Erasmus University Rotterdam, Econometric Institute, 2001.

[15] Goldberg, D., *Genetic Algorithms in Search: Optimization and Machine Learning*, Addison Wesley, 1998.

[16] Guo, Y., Mellouli, T., Suhl, L. and Thiel, M. P., *A partially integrated airline crew scheduling approach with time-dependent crew capacities and multiple home bases*, Technical Report WP0303, DS&OR Lab., University of Paderborn, Germany, 2003.

[17] Holland, J., *Adaptation In Natural And Artificial System*. University of Michigan Press, 1975.

[18] Lavoie, S., Minoux, M., and Odier, E., A new approach for crew pairing problems by column generation with an application to air transportation, *European Journal of Operational Research*, Vol.35, No.1, 1988, pp. 45–58.

[19] Levine, D., *Application of a Hybrid Genetic algorithm to Airline Crew scheduling*, Technical report, US Department of Energy, 1996.

[20] Mellouli, T., A network flow approach to crew scheduling based on an analogy to a train/aircraft maintenance routing problem. In S. Voß and J.R. Daduna, editors, *Computer-Aided Scheduling of Public Transport*, Springer, Berlin, 2001, pp. 91–120.

[21] Mellouli, T., *Scheduling and routing processes in public transport systems*, Habilitation Thesis, University of Paderborn, Germany, 2003.

[22] Ozdemir, H. T. and Mohan, C. K., Graga: a graph based genetic algorithm for airline crew scheduling, *In 11th IEEE International Conference on Tools with Artificial Intelligence*, 1999, pp. 27–28.

[23] Yan, S. and Tu, Y. P., A network model for airline cabin crew scheduling, *European Journal of Operational Research*, Vol.140, No.3, 2002, pp. 531–540