Evolutionary Algorithms for Nurse Scheduling Problem

Ahmad Jan
Graduate School of Engineering,
Hokkaido University,
Sapporo, 060, Japan.
jan@complex.eng.hokudai.ac.jp

Masahito Yamamoto
Graduate School of Engineering,
Hokkaido University,
Sapporo, 060, Japan.
masahito@complex.eng.hokudai.ac.jp

Azuma Ohuchi
Graduate School of Engineering,
Hokkaido University,
Sapporo, 060, Japan.
ohuchi@complex.eng.hokudai.ac.jp

Abstract- Nurse scheduling problem (NSPs) represents a difficult class of Multi-objective optimization problems consisting of a number of interfering objectives between the hospitals and individual nurses. The objective of this research is to investigate difficulties that occur during solution of NSP using Evolutionary Algorithms, in particular Genetic Algorithms (GA). As the solution method a population-less Cooperative Genetic Algorithms (CGA) is taken into consideration. Because contrary to competitive GAs, we have to simultaneously deal with the optimization of the fitness of the individual nurses and also optimization of the entire schedule as the final solution to the problem in hand. To confirm the search ability of CGA, first a simplified version of NSP is examined. Later we will report a more complex and useful version of the problem. We will also compare CGA with another multi-agent evolutionary algorithm using pheromone style communication of real ants. Finally, we will report the results of computer simulations acquired throughout the experiments.

1 Introduction

Generally, we can divide NSP approaches to conventional and Evolutionary Computation (EC) approaches, respectively. Two typical conventional methods are goal programming model[4], and mathematical programming [3]. Concerning EC approaches for solution of NSP, there have been only a few papers published so far. Yamamura et al proposed a Cooperative Genetic Algorithm (CGA) [2] and recently Yamamoto et al[5] proposed a collective multi-agent approach using pheromone style communication of real ants for solution of NSP. In this paper, the approach proposed by Yamamura et al [2] is taken into consideration. CGA initializes its search with an initial feasible schedule and continues searching in feasible search region only. This approach successfully generates feasible schedules, however due to presence of strong constraints, and restricting the algorithms to search in the feasible search region only, some optimization efforts are necessary to be carried out. In CGA new candidate solutions are created through application of an extended two-point crossover. Newly created candidate solutions are evaluated and updated using Pareto ranking scheme. Only non-dominated solutions are allowed to survive for further competition and are introduced to the Pareto rank1. From Pareto rank 1 of all non-dominated candidate solutions, only one solution is randomly selected and replaced with the previous schedule. The first phase of the search continues until the pre-defined objectives are met or generation reaches its maximum. As our future work, we are also interested to develop a second phase of the CGA to support decision maker interaction. In the second phase, we consider that the acquired schedule should be presented for the decision maker so that she can modify it as she wishes and as long as she is not satisfied with the acquired schedule.

Table 1: shifts and their corresponding assigned symbols

Shift		Symbol
day-shift	(8:00 - 16:00)	d
night-shift	(16:00 - 24:00)	n
late-night shift	(00:00 - 08:00)	l
off-days	as nurses' preference	h

The organization of this paper is as following. In section 2, we will briefly introduce NSP and problem definition. In section 3 hard and soft constraints. In section 4 fitness calculation including fitness factors of individual nurses. In section 5 description of the CGA for NSP and a brief description of a multi-agent approach for the same problem. In section 6 results of the computer simulations. In section 7 optimization results. Finally, in section 8 we will discuss and conclude this work.

2 Nurse Scheduling Problem (NSP)

NSP as a scheduling task consists of assignment of shifts and holidays to nurses for each day on the time horizon, taking into consideration a variety of conflicting interests or objectives between the hospitals and individual nurses. Given a number of nurses with specific skills and working agreements, a contract may consist of general constraints as there are restrictions on the number of nurses for each shift; the maximum number of shifts in a week, a month, etc. Moreover, a number of personal wishes or desires representing nurses' preferences are al-

Table 2: some typical examples of restriction conditions for real world NSPs

Classification of Constraints	Description
human related limitations	1) chief nurse is exempted from night-shifts 2) those in birth leave or sick leave are exempted, too. 3) do not assign night shifts for unskilled nurses only
DAILY restrictions:	4) each nurse can work only one shift in a day 5) number of day shift nurses must be $\geq$ required number of nurses for day shift 6) number of night shift nurses must be = required number of night shift nurses 7) number of late-night shift nurses must be = required number of late-night shit nurses
restriction on combination of the late-night shifts	8) consideration of the professional level of the nurses 9) do not combine those nurses, who are not in good relations with each other
restriction on individual nurses' monthly schedule.	10) number of legal holidays = assigned number of holidays 11) combination of the day shift and night shift is illegal 12) combination of the day shift and late-night shift is illegal
restrictions on the previous and next month's schedule	13) the interval between night-shift patterns must be at least $\geq$ one week. 14) request for the holiday when the coming next day is due night shift, is illegal 15) request for the holiday when it is due night-shift is illegal.
Nurses' Preferences	16) the right for assignment of the desired day off. 17) the right for selection of the preferred partners.
⋮ etc...	⋮ etc...

lowed. For instance, a demand for the desired day off, demand for doing certain shift on a certain day with a certain nurse, etc. Conventionally, every nurse works on three shifts, day shift, night shift, and late night shift and has some holidays. Throughout this paper, these shifts and holidays are denoted by symbols shown in table 1. For an image of the complexity of NSP, in table 2 some typical real world examples of the restriction conditions for NSP are summarized.

2.1 Problem definition

To describe NSP mathematically, let N, M be the number of nurses and days; w one of the three shifts or day off required to be scheduled, respectively. Then, NSP represents a problem to decide a $M \times N$ matrix, so that each X_{ijw} element of the matrix express that nurse i works her w th shift on day j.

where

N : Max number of nurses to be scheduled

M : Max number of days to be scheduled

$$X_{ijw} = \begin{cases} 1 & \text{(if Nurse } i \text{ works } w \text{ shift on day } j) \\ 0 & \text{(otherwise)} \end{cases}$$

Generally, the objective of solution of NSP is to find a schedule fulfilling a set of constraints representing the objective of the hospitals. Moreover, it is desired to satisfy as many wishes of the individual nurses as possible. In this paper, the maximum numbers of nurses and days to be scheduled is set to 15 nurses and 30 days, receptively.

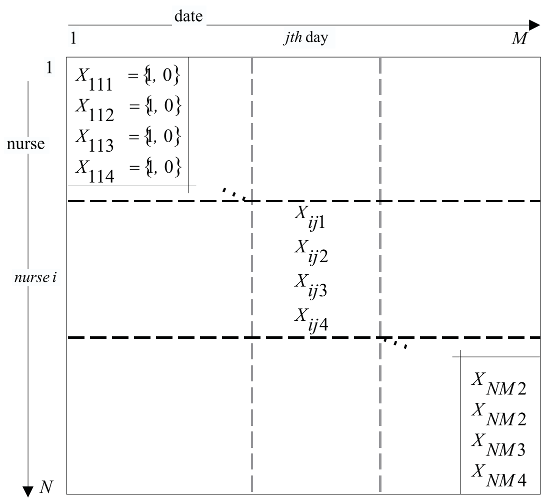


Figure 1: mathematical description of NSP

3 Hard and Soft Constraints

Existing constraints are generally divided into two major categories; hard and soft constraints. The hard constraints must always be satisfied. Violation of the hard constraints means that the acquired schedule is no longer

w	shift
1	n (night)
2	l (late-night)
3	d (day)
4	h (day off)

feasible. Soft constraints are desired to be satisfied as much as possible, however their violation does not lead to generation of infeasible schedules.

3.1 Assignment of shifts and restriction conditions

In this paper, from table 2, the following items are considered as the hard constraints (see also table 2 DAILY restrictions , items 4, 5, 6, 7):

$$\sum_{k=1}^{w} X_{ijw} = 1 \quad (\forall i,j) \tag{1}$$

$$\sum_{i=1}^{N} X_{ij1} \geq Rd_j = 9 \quad (\forall j) \tag{2}$$

$$\sum_{i=1}^{N} X_{ij2} = Rn_j = 2 \quad (\forall j) \tag{3}$$

$$\sum_{i=1}^{N} X_{ij3} = Rl_j = 2 \quad (\forall j) \tag{4}$$

Where:

Rd ... required number of nurses for day shift,
Rn ... required number of nurses for night shift,
Rl ... required number of nurses for late-night shift,
k ... number of allowed shifts per day for nurse i

As the nurses preferences, they are allowed to choose their day offs as they wish.

4 Fitness Calculation

To evaluate each nurse, a fitness function denoted by F_i is calculated for each nurse. All nurses are assigned the same fitness function. F_i consists of three factors that are described in the following subsections.

4.1 Fitness of the night shift pattern, in respect to it's order and length: F_i^p

To evaluate the number of the consecutive night shift patterns (as a vector value), four valid patterns and their corresponding penalty values are assigned. These patterns and their corresponding penalty values are shown in table 3. Selected length and order of F_i^p is observed, and corresponding fitness values are assigned. The more is the gap from the basic night patterns, the more penalty is charged.

Table 3: Pre-assigned fitness values of working-patterns

Working Pattern	Assigned Penalty F_i^p
l l n n	0
l l n	-40
l n n	-40
l n	-60
others	-120

4.2 Number of consecutive night shifts, in respect to its length: F_i^c

The aim of this evaluation factor is to observe the number of consecutive night shift patterns, regardless of their order. F_i^c is formalized according to the following equation:

$$F_i^c = \sum_k 100 \times max[c_k - 4] \tag{5}$$

Where, counting from the beginning of the month, c_k represents the length of the k th night shift pattern. In words, while the number of the consecutive night shifts are less or equal to four consecutive shifts, no penalty is assigned, otherwise corresponding penalty values are assigned.

4.3 The interval between night shifts: F_i^d

$$F_i^d = -\sum_j (d_j - 11)^2 \tag{6}$$

Where, d_j represents either of the j th number of the consecutive day shifts or day offs, counting from the beginning of the month. In words, if the interval between night shifts (i.e., either of the consecutive day shifts or day off) is exactly 11 days, it is considered as an ideal interval, and thus no penalty is charged. Otherwise, a penalty according to F_i^d is assigned.

4.4 Overall fitness of individual nurse: F_i

Overall fitness of individual nurses, F_i is calculated according to equation 7.

$$F_i = \alpha F_i^p + \beta F_i^c + \gamma F_i^d \tag{7}$$

α, β, γ are parameters, representing specific criterion of the hospitals, chief nurses, etc. After all, the objective function for the entire schedule is formalized as:

Maximize: $avg = \dfrac{1}{N}\sum_{i=1}^{N} F_i$

Minimize: $dev = \sqrt{\dfrac{1}{N}\sum_{i=1}^{N} F_i^2 - \dfrac{1}{N^2}(\sum_{i=1}^{N} F_i)^2}$

Subject to: DAILY restrictions (see subsection 3.1)

It is desired to attain maximization of the average of the fitness of all nurses, and minimization of the variance of all nurses. A schedule with $avg = 0$ and $dev = 0$ represents the desired absolute solution to the problem. Moreover, for individual nurses $F_i = 0$ represents the best fitness value. Since it is very difficult to predict whether there ever exists a solution to a schedule with the values of $avg = 0$ and $dev = 0$, thus it is desired to attain the values of avg and dev as close to zero as possible.

4.5 Multi-agent approach for NSP

In order to compare search performance of CGA with another evolutionary approach for NSPs, a brief overview of a multi-agent approach for solution of the same model of NSP is described. Yamamoto et al [5] proposed an algorithms for solution of NSP using pheromone style communication of real ants. Search procedure of this approach begins with an infeasible initial schedule. Each nurse tries to improve her own schedule using two kinds of operators, namely 'swap' and 'slide' operators. The pheromone information is used to resolve conflicts among nurses. Unfortunately, this approach does not always guarantee generation of feasible schedules, which is a serious drawback in real NSPs.

5 Algorithmic Flow of CGA

Figure 2 shows the pseudo code of CGA for the experiments reported in this paper. To begin with, let us briefly describe search procedure of CGA . After initialization of the initial feasible schedule and 15 days of the history of the previous month, (see figure 3 top and figure 4 top), all nurses are evaluated according to their fitness F_i, including the 15 days of the history of the previous month. To create a new schedule, from the current schedule two nurses are selected, the first nurse with the worst F_i and the second nurse is randomly selected. Moreover, two crossing points are selected so that all legal children could be generated. As crossover progresses new solutions (new_ch) are generated and updated if and only:

if ((new_ch.avg≥current.avg) && (new_ch.dev≤current.dev))

Any candidate solution that is no longer non-dominated is simply discarded from the list of candidate solutions If and only if the new_ch is non-dominated, it can survive and is introduced to the rank 1 of the Pareto non-dominated solution From rank 1 of all non-dominated candidate solutions, only one solution is randomly selected. The selected solution becomes the current schedule. The first phase of the search continues until for all nurses $F_i \neq 0$ or generation≤ Max_generation. We also consider a second phase of CGA for decision maker interaction. In the second phase of the search, the acquired schedule is presented to the decision maker. If she is satisfied, CGA terminates. While she is not satisfied, the system asks her for entering a benchmark day[j-MAXD] representing a point in the schedule where no more changes are needed. For the remaining days (day[j+1] to day[MAXD]) she is asked to enter nurse[n].shift[j] that she wishes to replace with nurse[m].shift[j]. After replacement of the above mentioned shifts, CGA continues to search for n runs and search is restricted from day[j+1] to day[MAXD]. The acquired schedule is presented for the decision maker and this process continues until the decision maker is not satisfied. Currently we work on the second phase of CGA and we will report the acquired results in our future works.

Table 4: parameter settings for the experiments

time horizon	30 days
history of the previous month	15 days
max number of nurses	15 nurses
max generation	10000
α=β=γ	1.0

6 Computer Simulations

6.1 Confirmation of the search ability of CGA

To confirm the search ability of CGA (without application of mutation and escape, described in subsection 7.3), some preliminary experiments were carried out. Table 4 shows parameter settings for the computer simulations being reported. During these experiments day off was not taken into consideration, i.e, the initial schedule shown in figure 3(top) was initialized.

Figure 3 (bottom) shows the best observed schedule obtained during the preliminary experiments. This experiment revealed the existence of the absolute solution for simplified version of the problem. However, the solution was observed only once out of each 12 independent runs of the CGA. Later our experiments showed that after some optimization efforts it is possible to acquire the best solution for the simplified version of the problem in each run of the CGA, i.e, after application of the escape operator, that is described in subsection 7.3.

6.2 Consideration of the day off

In this experiment, the schedule shown in figure 4 (top) with addition of the day offs was initialized. However, the absolute solution was not observed. Unfortunately it is very difficult to predict whether the absolute solution for this schedule ever exists. To investigate whether it is possible to improve the search performance of CGA, some experiments were carried out that are described in the following subsections.

Figure 2: Co-operative GA for NSP

```
1. main(){ /* main function */
      generation==0; load initial feasible schedule and 15 days of the history of previous month;
      sum=0; sqr_sum=0;
         for( i=0; i<MAXN; i++ ){
               for( j=0; j<MAXH; j++ ){ /* 15 days of the history of the previous month */
               evaluate(nurse[i]); /* according to F_i */
               sum+=nurse[i].fitness; sqr_sum+=nurse[i].fitness×nurse[i].fitness;
               avg=sum/MAXN; dev=sqrt( sqr_sum/MAXN-avg*avg );
               }
         }

   do {
         if(generation==mutation_frequency) mutation();
         else if (generation==escape_frequency) escape();
         else generation();
         if(generation%250==0) print_current_schedule(); /* print current schedule after each 250 generations */
   } while (∀ F_i ≠ 0 or generation ≤ Max_generation);
} /* end of the main function */

2. description of the utility functions: /* MAXN, MAXD: Max number of nurses and days to be scheduled */
      crossover(p1, p2, c1, c2, cp1, cp2){
         for( j=0; j<MAXD; j++ ){
            flag=TRUE;
            if (nurse[n1].shift[j]==h | | nurse[n2].shift[j]==h) flag=FALSE; /* replacement of h is illegal */
            if (flag)
               if ( j≥cp1 && j<cp2 ){c1[j]=p2[j]; c2[j]=p1[j];} else { c1[j]=p1[j]; c2[j]=p2[j];}
         }

      mutation(){
         let n1, and n2 be two different nurses.
         mp1=select(MAXD); mp1=select(MAXD); /* select mutation points */
         flag=TRUE;
         if (nurse[n1].shift[mp1]==h | | nurse[n2].shift[mp2]==h) flag=FALSE; /* replacement of h is illegal */
         if (flag) swap( nurse[n1].shift[mp1], nurse[n2].shift[mp2] );
         }

      escape(){
         let n1, and n2 be two different nurses.
         do{ep1=select(MAXD); ep2=select(MAXD); /* select escape points */
            while(ep1==ep2){ep2=select(MAXD); } flag=TRUE;
            for(i=ep1; i≤ep2; i++){if (nurse[n1].shift[i]==h | | nurse[n2].shift[i]==h) flag=FALSE;}
            } while(!flag);
         if (flag) {
               for(j=0; j<MAXD; j++){if (j≥ep1 && j<ep2) swap( nurse[n1].shift[j], nurse[n2].shift[j] ); }
         }

      pareto(){
               ∀(new_ch.c) update, new_ch.avg and new_ch.dev according to:
                  if((new_child.avg ≥ current.avg)&&(new_child.dev ≤ current.dev))
         }

      generation(){
         let mate1, and mate2 be two different nurses.
         sum-=nurse[mate1].fitness+nurse[mate2].fitness;
         sqr_sum-=nurse[mate1].fitness×nurse[mate1].fitness+nurse[mate2].fitness×nurse[mate2].fitness;
         for( cp1=0; cp1<MAXD; cp1++ ){
            for( cp2=cp1+1; cp2≤MAXD; cp2++ ){
               flag=TRUE;
               for( i=cp1; i<cp2; i++ ){
                  if ( nurse[mate1].shift[i]==h | | nurse[mate2].shift[i]==h ) flag=FALSE;}
               if ( flag ){ new_ch.c1=nurse[mate1]; new_ch.c2=nurse[mate2];
                     crossover( nurse[mate1].shift, nurse[mate2].shift,new_ch.c2.shift, new_ch.c2.shift, cp1, cp2 );
                     evaluate(new_ch.c1); evaluate(new_ch.c2); /* according to F_i */
                     new_ch.avg=(sum+new_ch.c1.fitness+new_ch.c2.fitness)/MAXN;
                     new_ch.dev=sqrt(sqr_sum+new_ch.c1.fitness×new_ch.c1.fitness+
                                    new_ch.c2.fitness×new_ch.c2.fitness/MAXN-new_ch.avg×new_ch.avg );
                     pareto();
                     generation=generation+1;
         }
```

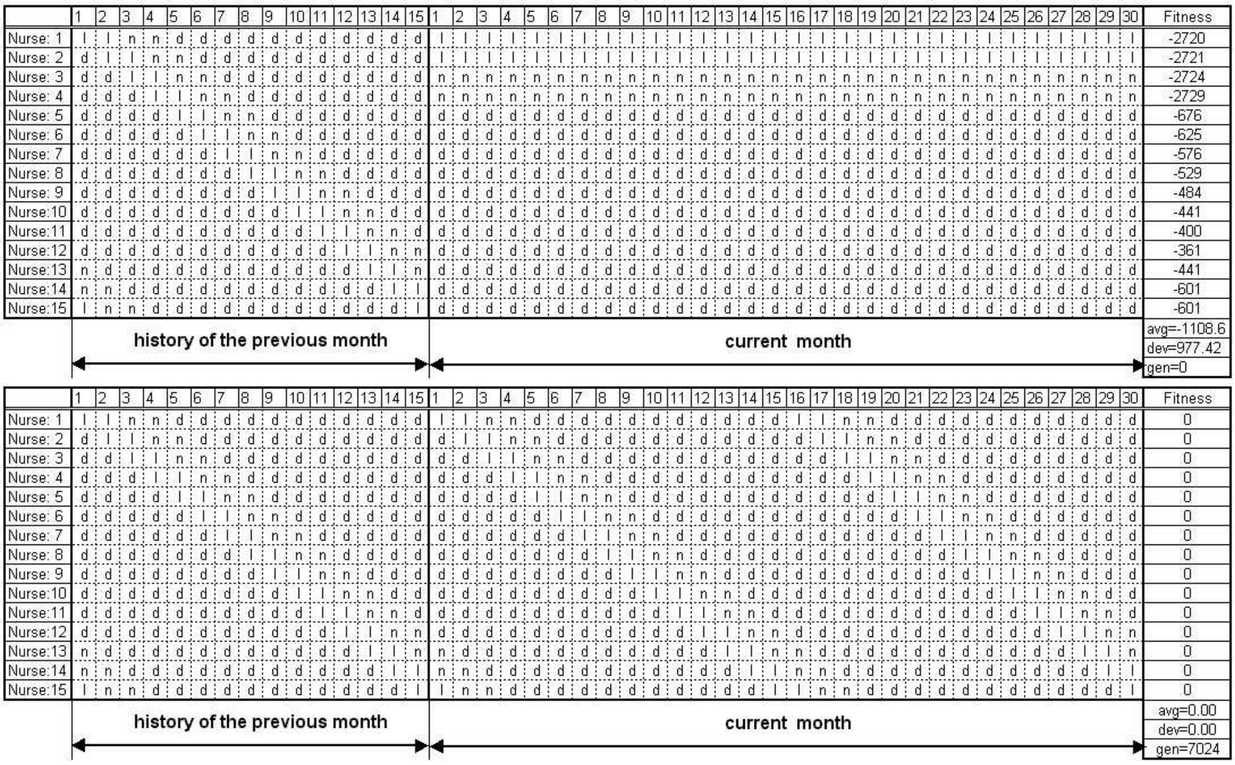


Figure 3: top: simplified version of the initial schedule, bottom: acquired final schedule

7 Optimization Efforts

7.1 Increasing the number of mates for crossover

In an attempt to generate more candidate solutions and explore in a wider portion of the solution space, the following experiments with GA operators were performed. Firstly, we increased the number of the selected mates for crossover, from 2 to 4 and to 6 mates, respectively. With more mates for crossover, considerable results were not acquired. In addition, comparing to the selection of two mates for crossover, with more mates computational cost of CGA gets higher as expected.

7.2 Diversification of the solution space

Since CGA is a population-less approach, consideration of other ranks rather than rank 1 and the use of some kind of niching techniques to keep CGA from converging to a single point on the front (as suggested by Goldberg [1]) is not applicable. As an alternative, to enable CGA for more exploration of the search space, in subsection 7.3 we will introduce a simple technique called "escape" operator.

7.3 Application of the mutation and escape operators

As the second attempt we applied mutation. During application of traditional mutation, two nurses i and j and two mutation points are randomly selected. Then, their corresponding shifts are interchanged. According to the results of simulations shown in table 6, and figure 6, traditional mutation fails to to improve search performance of CGA. as the probability of selection of day shifts are very high. As an escape strategy from the local minima and a hope for exploration of larger portion of the search space, we developed a very simple operator that is called escape operator. During application of the escape operator, two nurses i and j are randomly selected and a block-wise exchange of shifts between the selected nurses is carried out. Detailed explanation of the escape operator is given in figure 2. The best observed frequency of application of the mutation and escape operators are shown in table 5. It was observed that the best frequency of application of the escape operator is after each 10 generations.

Table 5: best observed frequency of mutation and escape

Max generation	10,000
Max run	100
Best observed mutation frequency	1950
Best observed escape frequency	10

Figure 4 (bottom) shows a typical view of the final acquired schedule after application of the escape operator after each 10 generations.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Fitness
Nurse: 1	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	l	l	h	h	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	h	l	l	l	l	l	l	-2041
Nurse: 2	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	l	l	l	l	l	l	l	h	h	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	-2322
Nurse: 3	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	n	n	n	n	l	l	n	l	l	n	n	n	h	h	n	n	n	n	n	n	n	n	n	l	n	n	n	n	n	n	-2325
Nurse: 4	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	n	n	n	n	n	n	n	h	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	-2449
Nurse: 5	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	n	n	d	n	n	d	d	d	n	n	d	d	d	h	h	d	d	d	d	n	d	d	d	d	d	d	-768
Nurse: 6	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	h	d	d	d	d	d	-365
Nurse: 7	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	d	h	d	d	d	d	d	-576
Nurse: 8	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	-529
Nurse: 9	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	h	d	d	d	d	d	d	d	-484
Nurse:10	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	d	d	d	d	d	h	d	d	d	d	d	d	d	d	d	d	-441
Nurse:11	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	h	h	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	-400
Nurse:12	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	-361
Nurse:13	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	h	d	d	d	d	d	d	d	d	d	d	d	-441
Nurse:14	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	d	d	d	d	d	d	d	d	h	h	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	-601
Nurse:15	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	d	d	h	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	-601
																																														avg=-980.27
																																														dev=796.55
																																														gen=0

history of the previous month | current month

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Fitness
Nurse: 1	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	h	l	l	n	n	d	d	d	d	d	d	d	d	d	l	l	n	n	h	d	d	d	d	d	d	-40
Nurse: 2	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	h	h	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	-2
Nurse: 3	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	l	n	n	d	d	d	d	d	d	d	d	d	h	h	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	-44
Nurse: 4	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	h	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	-5
Nurse: 5	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	h	h	d	l	l	n	n	d	d	d	d	d	d	-13
Nurse: 6	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	l	l	n	n	d	h	h	d	d	d	d	d	-4
Nurse: 7	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	h	d	h	l	l	n	n	d	-40
Nurse: 8	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	l	l	n	n	d	d	d	h	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	-17
Nurse: 9	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	h	h	l	l	n	n	d	d	d	0
Nurse:10	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	h	d	d	d	d	d	h	d	d	l	l	n	n	d	d	d	d	-4
Nurse:11	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	h	h	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	d	l	l	-6
Nurse:12	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	l	l	n	n	d	d	d	h	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	-17
Nurse:13	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	h	h	d	d	d	d	d	l	l	n	n	d	d	-42
Nurse:14	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	d	d	d	d	d	d	d	h	h	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	l	n	n	-41
Nurse:15	l	n	n	d	d	d	d	d	d	d	d	d	d	d	l	n	n	h	d	d	d	d	d	d	d	d	d	d	l	l	n	n	d	d	d	d	d	d	d	d	d	d	l	l	n	-41
																																														avg=-21.07
																																														dev=17.22
																																														gen=6355

history of the previous month | current month

Figure 4: top: initial schedule with day off being added, bottom: acquired final schedule

As is shown in figure 5 applying escape operator CGA is given a greater chance to explore in a larger portion of the search space. As expected, some improvement is also acquired. Figure 5 shows distribution of the solution space prior to and after application of the of the escape operator.

Figure 6 shows comparison of the effect of application of the mutation after each 1950 generations and application of the escape operator after each 10 generations(the average of 100 runs of CGA).

7.4 Effect of the number of generations

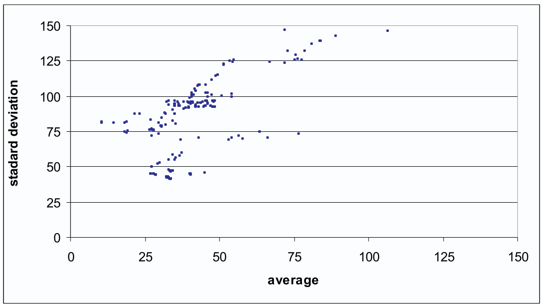
To confirm whether CGA can explore better results with more generations, the *Max_*generation was set to 100,000 generations. Best observed results of 100 independent runs of CGA are summarized in table 6. According to the acquired data, increasing the number of generations CGA can explore better results. All experiments were carried out on a PC with CPU Pentium 2 Processor 300 MHz. The total of the execution time for 10000 generations of CGA was 49 seconds.

Table 6: comparison of the effect of the number of generations

Max generation	state	best generation	best avg.	best dev.	times found
10,000	CGA mutation escape	2112 1105 9774	-41.73 -43.4 -32.27	25.64 26.08 22.71	123 122 159
100,000	CGA mutation escape	44648 1106 54224	-32.40 -43.4 -13.47	24.91 26.08 20.00	138 122 184
max-step 200,000	multi-agent		-17.71	22.94	

8 Conclusion

The objective of this research was to investigate problems that occur during solution of NSP using Evolutionary Computation approaches, in particular, GAs. As a case study CGA was applied to NSP. During the preliminary experiments the best solution for the simplified version of the problem was found only once out of 12 independent runs of CGA. The presence of strong constraints and limiting CGA to search only in feasible solution space required some optimization efforts. In an attempt to improve the search performance of the CGA, the number of selected mates for crossover were changed from two to more mates. However, there was not a considerable improvement acquired, in addition computational cost of CGA was increasing. According to reported results, application of the conventional mutation



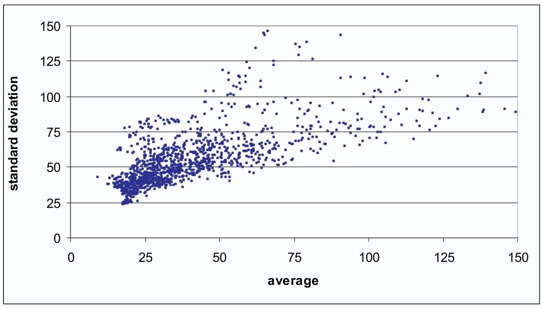


Figure 5: distribution of the solution space: prior to (top) and after (bottom) application of the escape operator

alone itself was not sufficient. Since the original CGA represents a population-less approach, consideration of other ranks rather than rank 1 and the use of some kind of niching techniques to keep CGA from converging to a single point on the front was not applicable to CGA. As an alternative, a so called escape operator was taken into consideration. Application of the escape operator enables CGA to explore in a larger portion of the solution space, moreover as expected some performance improvement is also acquired. Therefore for the moment we consider application of escape operator as the best choice for improvement of the search ability of CGA, because of it's simplicity and efficiency. Comparing to the multi-agent approach, CGA always satisfies hard constraints successfully i.e., required number of nurses for each shift is always satisfied. Moreover, after application of the escape operator and increasing the number of generations, CGA outperforms the multi-agent approach. In the current model of NSP from the table 2 the most important restriction conditions have been taken into consideration, i.e., the required number of nurses for each shift and as the nurses preferences they are given the right to select their day offs as they wish. CGA can find reasonable final schedules satisfying all predefined hard constraints. However, from the practical scheduling point of view, it is necessary to investigate the degree of satisfaction of the desicion maker regarding the acquired schedule. For this purpose we are considering a second phase of CGA

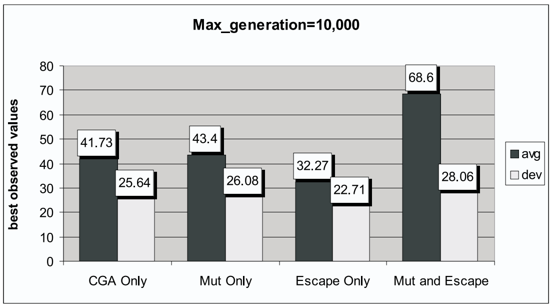


Figure 6: comparison of CGA and the effect of application of mutation and escape operators

for decision maker interaction. This is another topic that we would like to consider in our future studies of the problem. Besides, the F values (figure 4 bottom) are negative values. In the current model of NSP it is desired to optimize the fitness of individual nurses as much as possible. However, as they represent soft constraints, they can be violated. Regarding problem definition and design of the fitness function of the individual nurses, it is noted that the problem is not described realistically. For example, the length of consecutive day shifts is considered too long, the same fitness function is assigned for all nurses, etc. However, for evaluation purpose the definition of the problem is sufficient enough. In our future studies of practical scenario of NSP, we will reconsider the discussed difficulties and alter the present approach accordingly.

Bibliography

[1] Goldberg, D., Genetic Algorithm in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA 1989.

[2] Kitano, H. Genetic Algorithm vol 3, Sangyou Tosho, pp.89-126 in Japanese, May, 1995.

[3] D.Michael, Scheduling Nursing Personnel According To Nursing Preference: A Mathematical Programming Approach Duke University, Durham, North Carolina, 1977.

[4] I.Ozkarahan, (1987) Goal Programming Model Subsystem of Flexible Nurse Scheduling Support System Pennsylvania State University, Behrend College, School of business.

[5] M, Yamamoto et al, Collective Approach to Optimization Problems, Proceedings of ITC-CSCC '98, pp 1479-1482