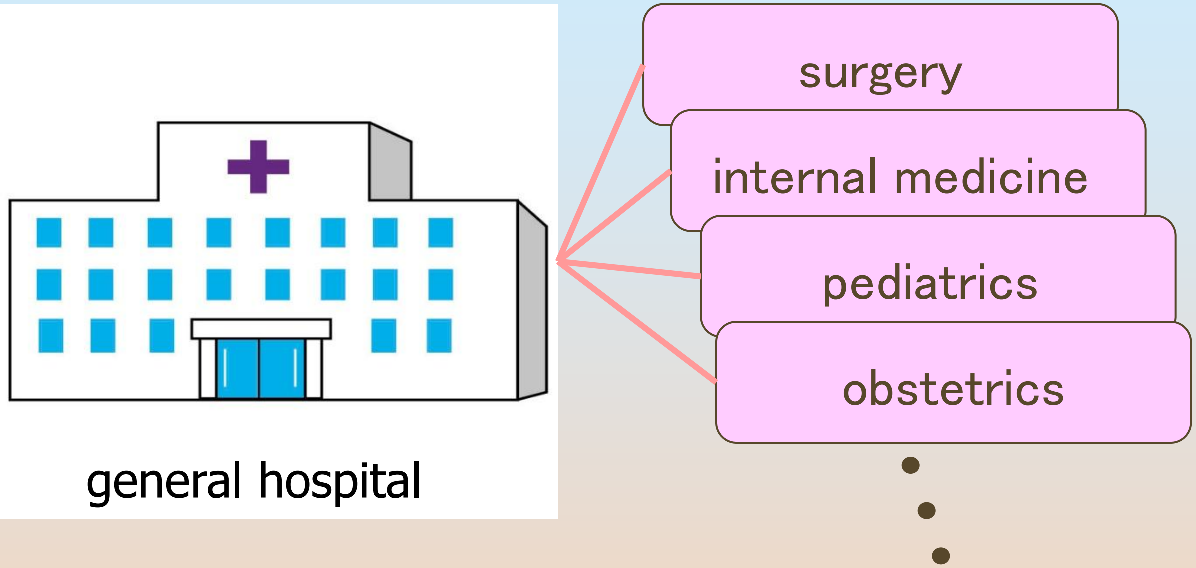
Penalty Weight Adjustment in Cooperative GA for Nurse Scheduling

1. Introduction

1. Introduction



- General hospital consists of several sections.
- About 15–30 nurses belong to each section.
- A section director or a manager arranges a shift schedule of the nurses every month.

1. Introduction

many requirements

- requirements on the hope holiday.
- duty load in equality.
- the number of the night shift in equality.
- affinity between the nurses in the night shift.
- etc. ・・・・

Veteran director requires one or two weeks.

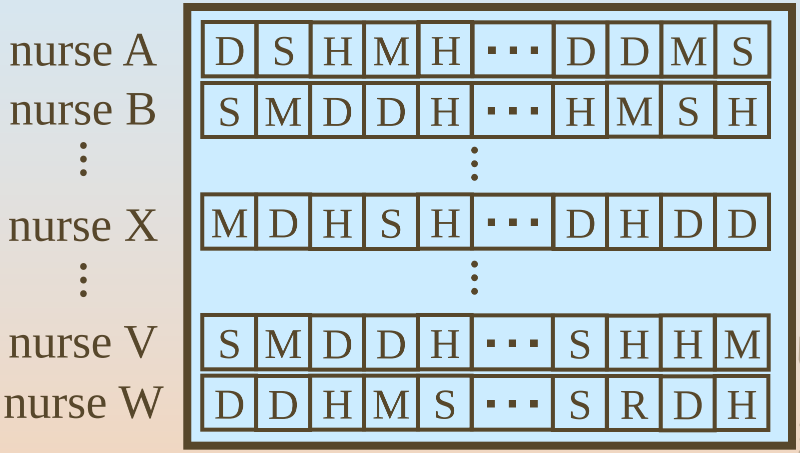
Automatic Nurse Scheduling

- The nurse scheduling is very complex task, because the director consider <u>many requirements</u> for the scheduling.
- In our investigation, even veteran director spends **one or two weeks** for the nurse scheduling.
- This means a great loss of work force and time.
- Therefore, computer software for the nurse scheduling is strongly required recently.

1. Introduction

- We know there are actually several commercial software to generate nurse schedule. However, they are not used, because the optimized result is **dissatisfactory**.

- We discuss the cooperative GA (CGA) to generate & optimize the nurse schedule.



- The conventional CGA searches solutions **only by using crossover** operator, because it is considered as the only one operator keeping **consistency** of the population.

- A **mutation** changing small parts of the population brings very important change to the population.

1. Introduction

- We have proposed an **effective mutation operator** activated depending on the optimization speed [19].

  ⇒ This requires some parameters to define itself.

- We have proposed a mutation operator activated periodically, **Periodic Mutation** [20, 21].

  ⇒ It brings almost same result as the earlier one.

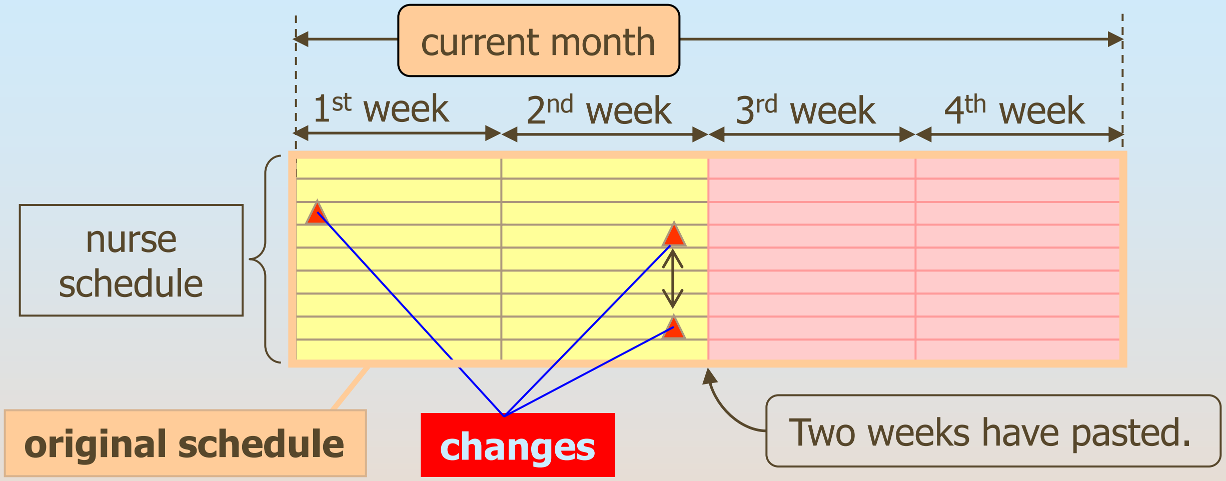
  Only one parameter is required.

1. Introduction

- In the actual cases, there are the cases that nurses originally assigned to a rest are forced to attendance by means of emergency.
- There are also the cases that a nurse whom duty has been assigned originally takes a rest due to a disease.

- The shift schedule must be changed in the actual.

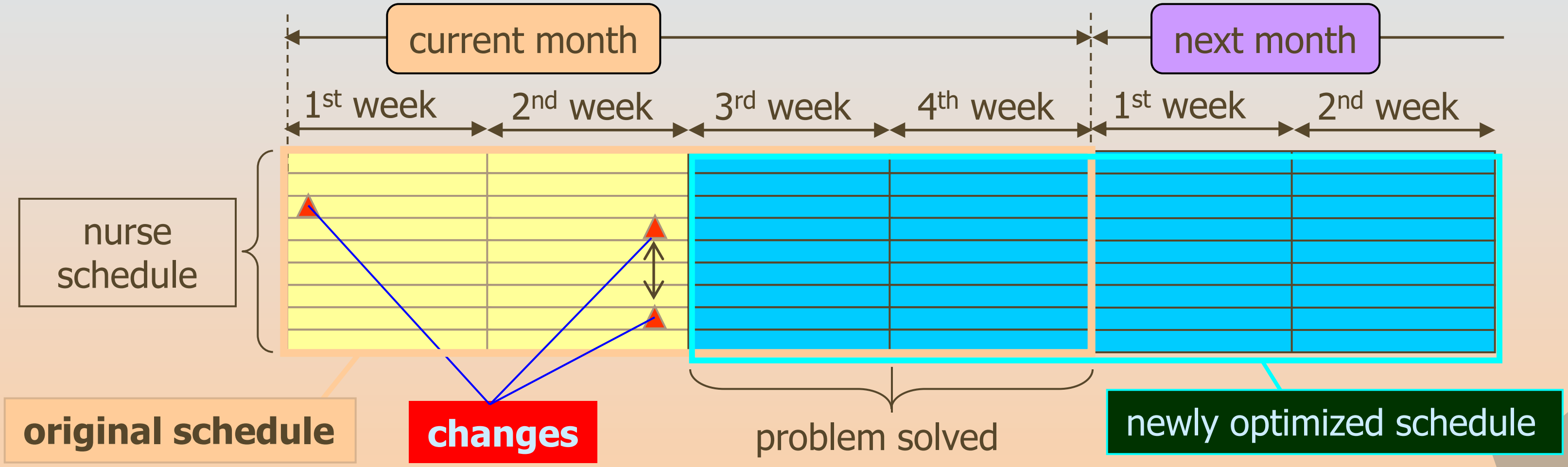
1. Introduction



- By means of the **changes**, several **inconvenience** occurs, for example, imbalance of the number of the holidays/attendance.
- Such an **inconvenience** causes the fall of the nursing level of the whole nurse organization.
- The changed schedule should be re-optimized to **break off** the **inconvenience** as much as possible.

1. Introduction

- On the other hand, there is a requirement that the shift schedule already given does not want to change if possible.
- We define a penalty function to improve this confusion. It calculates the difference between the original and the optimized schedules.



- In this research, we treat the re-optimization of the coming four weeks (one month) including the remaining weeks of the current month and the several weeks of the next month.

1. Introduction

- The re-optimization of such the changed schedule is difficult even by using the effective mutation operator.

⇒ We have proposed an **effective parallel mutation operator** for the re-optimization [23,24].

However, ･･･

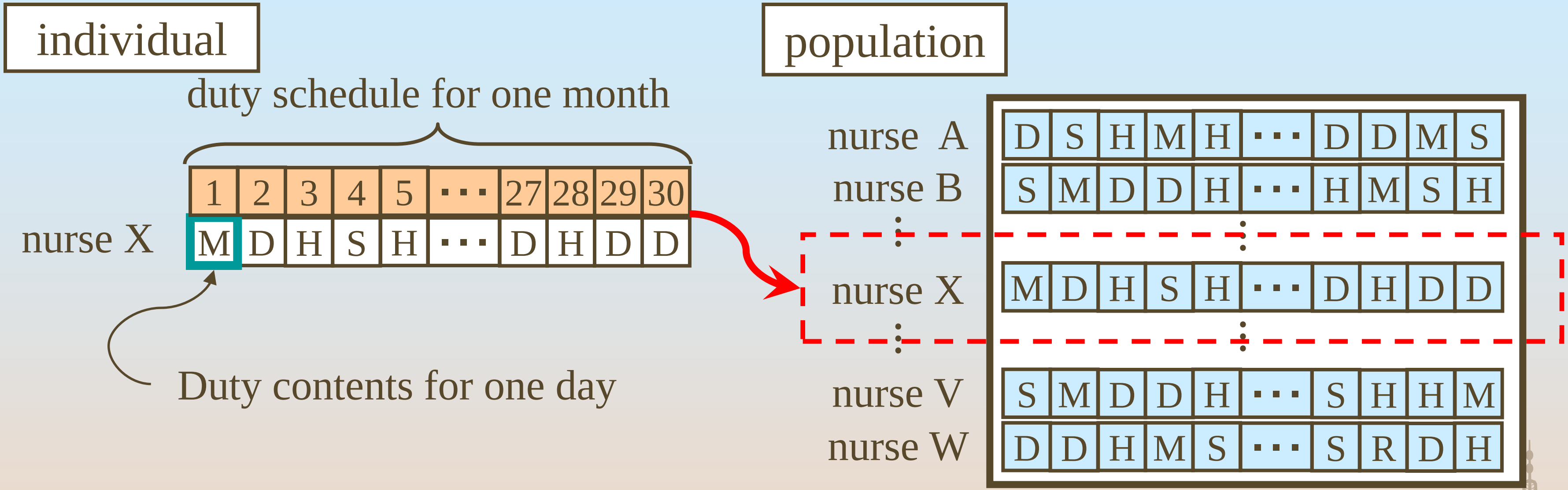
- The re-optimization is very hard task even by the parallel mutation, because the **solution space** of the nurse scheduling problem is **very complex**. There are many local minimum area in the solution space.

in this presentation, ･･･

- We propose a technique to escape from such the local minimum area by using **Penalty weight Adjustment**.

2. Evaluation of Nurse Scheduling

2. Nurse Scheduling



- An individual consists of the sequence of the shift symbols.
- The individual shows the one-month schedule of the nurse X.
- Gathering all the individuals, the population is constructed.
- In CGA, the population does not contain two or more individuals giving the same nurse's schedule.

2. Nurse Scheduling

- The following two requirements have to be satisfied by means of the genotype coding and the genetic operations.

(**Strong Constraints**)

  - meeting, training and requested holiday must be accepted.
  - the number of nurses at each shift interval must be secured.

- The following requirements are evaluated by penalty functions.

  (**Weak Constraints**)

  - duty load depending on the duty pattern of consecutive 3 days. (F_1)
  - 4 or more night shifts should not be assigned. (F_2)
  - prohibited duty patterns. (F_3)
  - fairness of the holidays and the night shifts assignment. (F_4, F_5)
  - more than or equal to 6 consecutive duty days. (F_6)
  - nursing levels must be kept at each shift interval. (F_7, F_8, F_9)
  - unfavorable combinations in the night shift. (F_{10})
  - two or more new faces should not assigned to the midnight shift. (F_{11})
  - one or more expert or more skilled nurses must be assigned on day time. (F_{12})
  - difference between original schedule and optimized schedule. (F_{13})

2. Nurse Scheduling

- Finally, the total penalty function E is defined by summarizing these **penalty functions** with **weights**.

$$E = \sum_{i=1}^{M} \sum_{k=1}^{6} h_k F_{ki} + \sum_{j=1}^{D} \sum_{k=7}^{12} h_k F_{kj} + h_{13} F_{13} \qquad (1)$$

3. Basic Algorithm of CGA

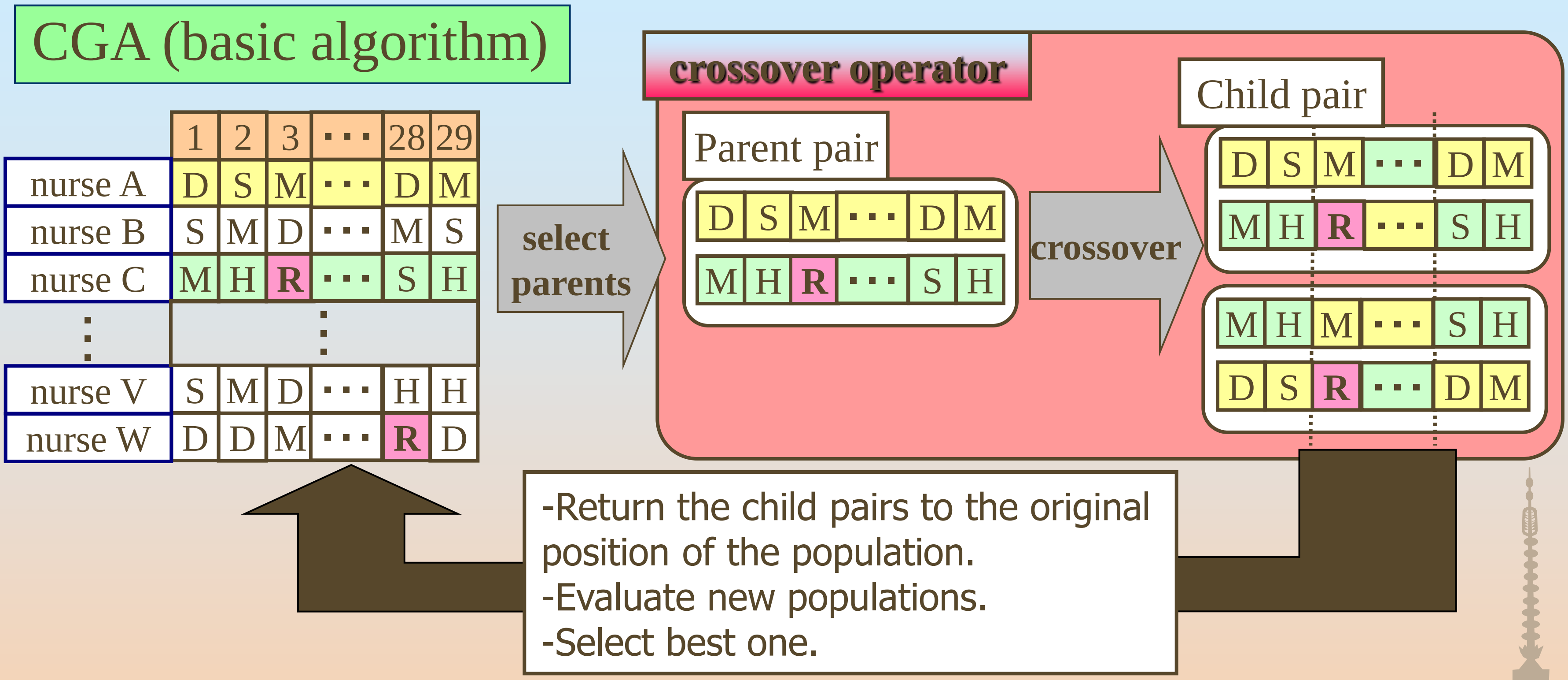
3. Basic Algorithm of CGA

CGA (initialization)

D: day shift, S: semi-night shift, M: midnight shift
R: requested holiday, H: holiday
m: meeting, T: training

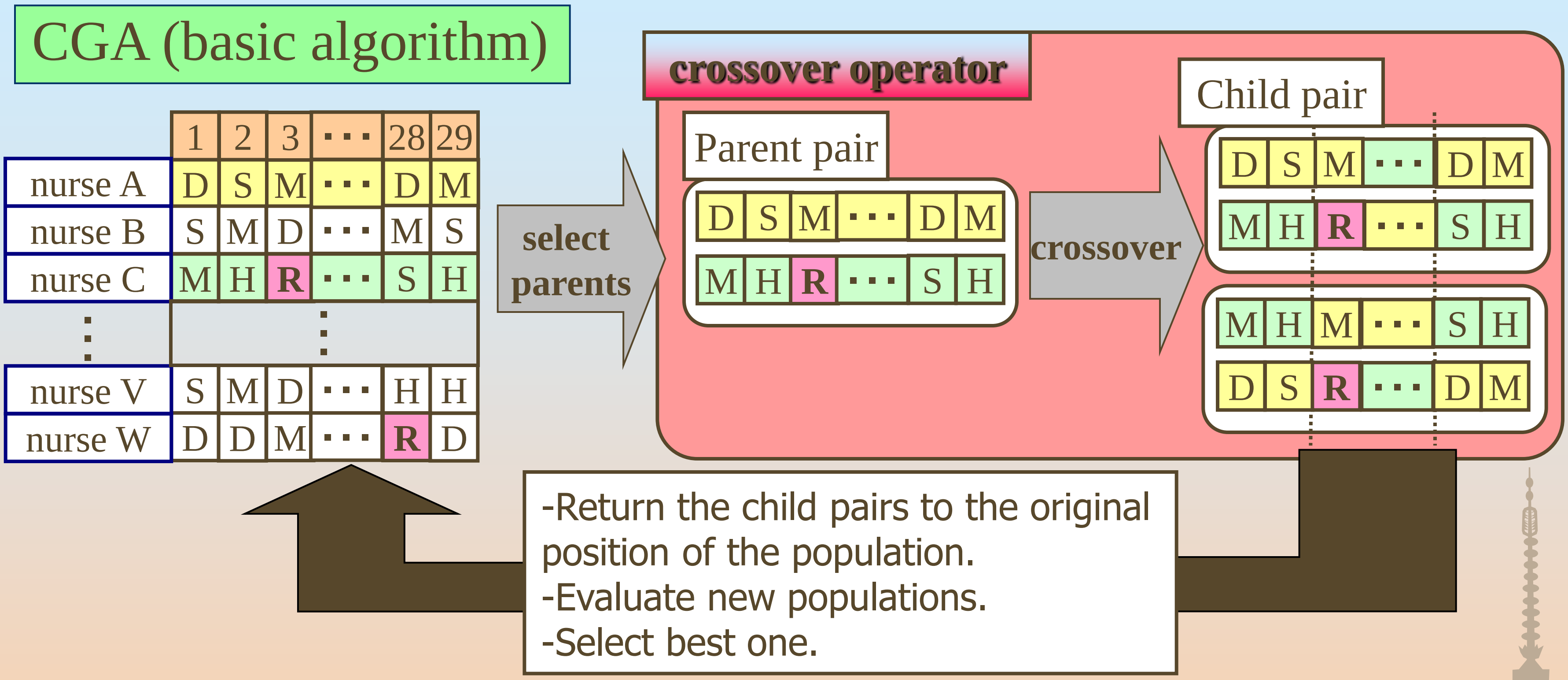
- CGA put the shift symbols randomly with keeping the number of nurses at the day time shift, the semi-night shift and the midnight shift as 6, 3 and 3 respectively.

3. Basic Algorithm of CGA



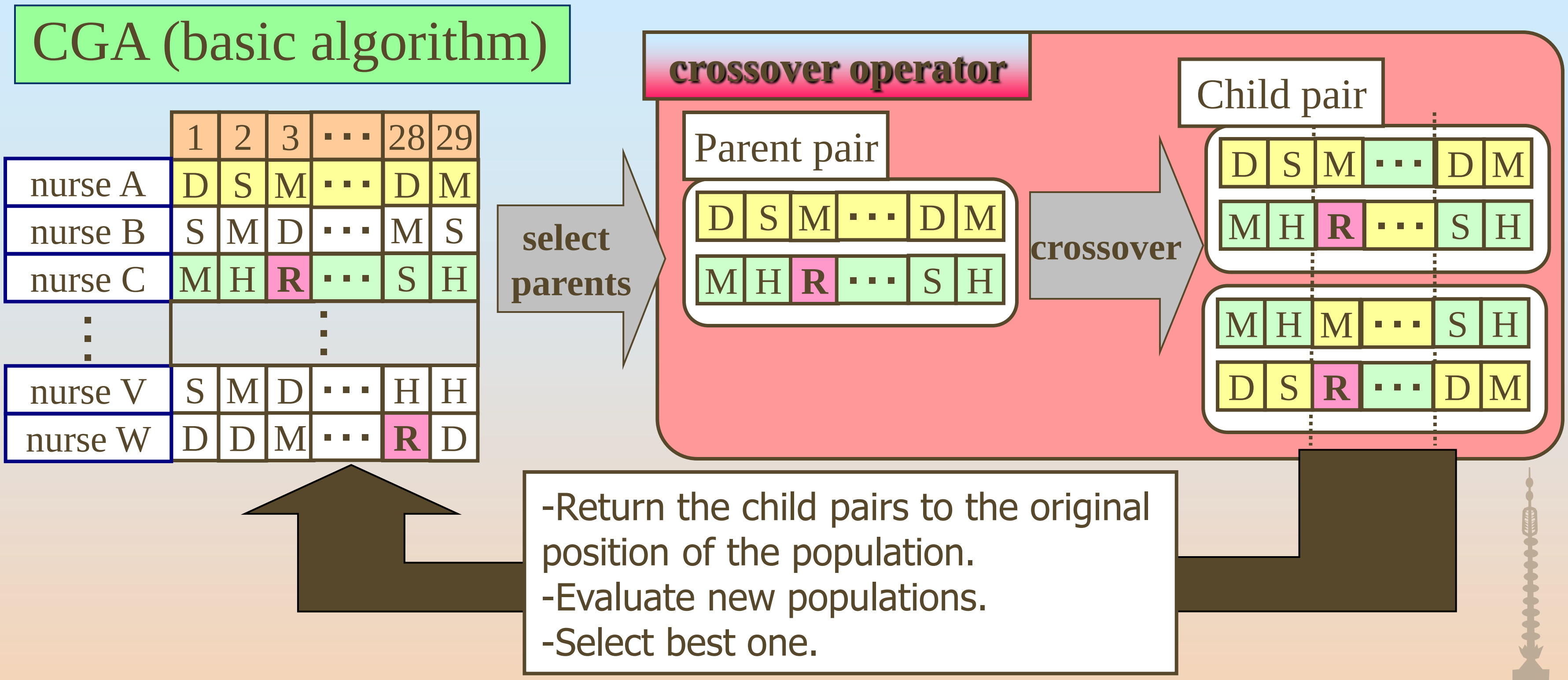
- CGA searches good solution by basically using the crossover operator.

3. Basic Algorithm of CGA



- The crossover operator selects two individuals, where one is selected by roulette selection manner and another is randomly selected.
- By the two-points crossover, two child pairs are regenerated.

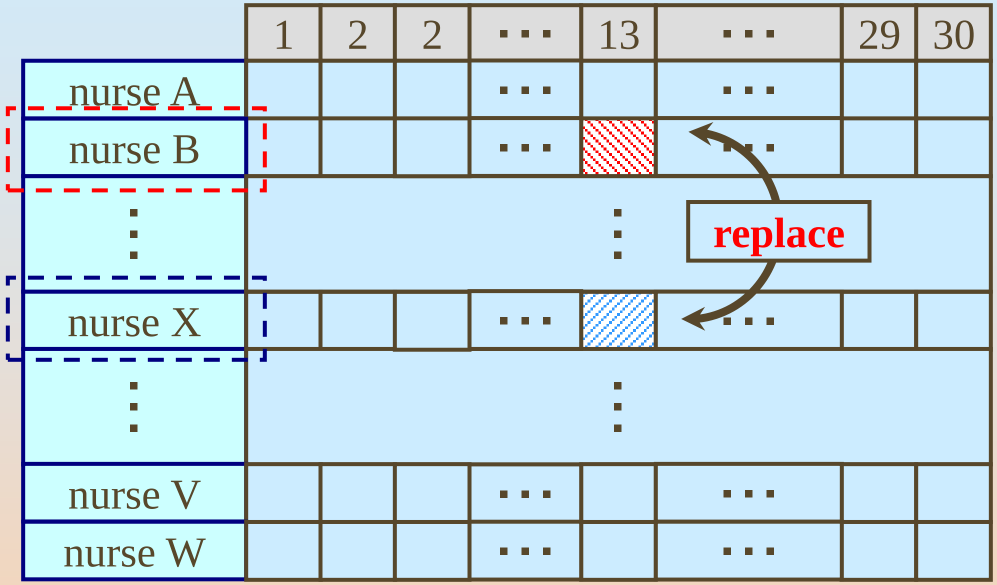
3. Basic Algorithm of CGA



- Setting these child pairs back to the original positions of their parent pair, the population is evaluated by the penalty, E.
- This procedure is applied to 100 parent pairs in 1 generation cycle.

3. Basic Algorithm of CGA

CGA (mutation operator)

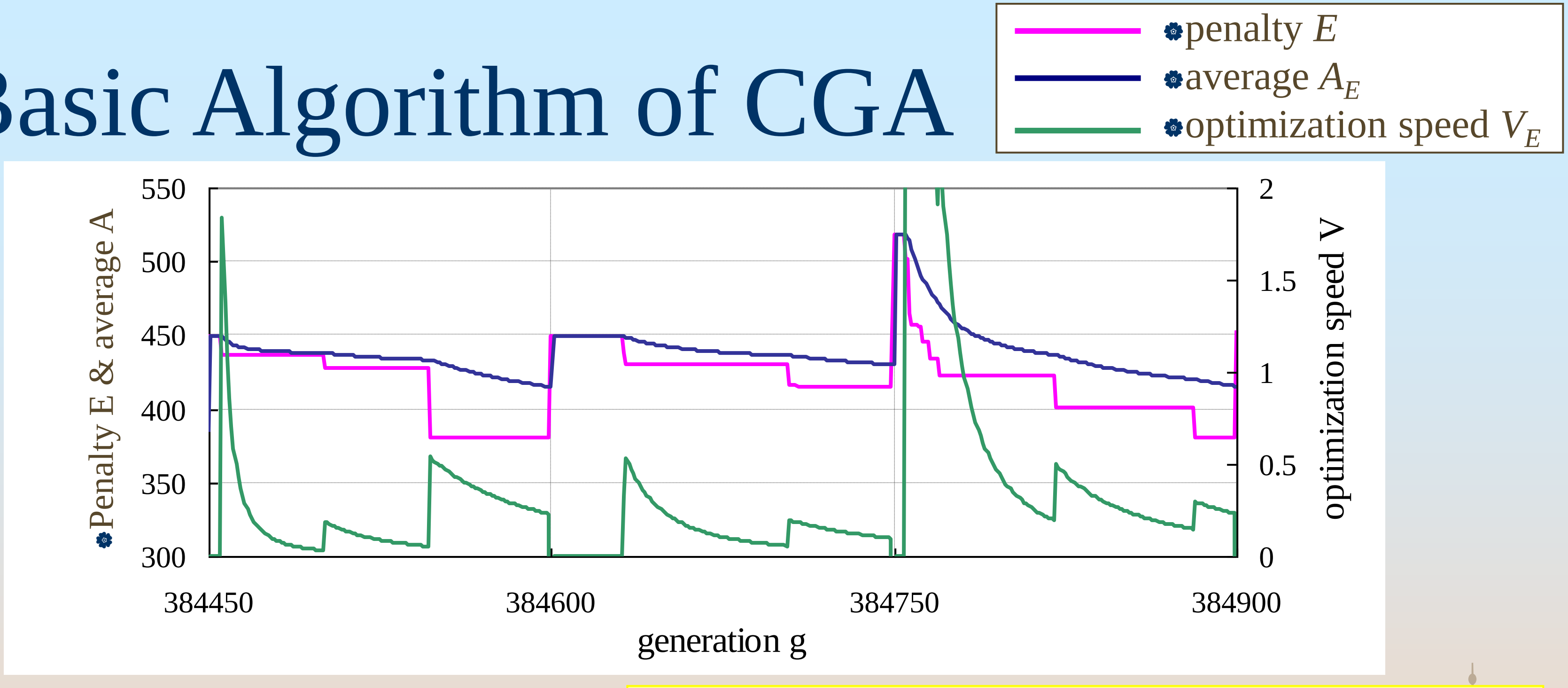


① Randomly select one of shift dates.

② Randomly select two nurses. If one of them or both two are fixed shift, return to ①.

③ Replace these two shifts.

3. Basic Algorithm of CGA



- Average value A_E of the penalty E for N_g generation cycles after mutation:

$$N_g = g - g_{prim}, \qquad (2)$$

$$A_E(g) = \frac{1}{N_g} \sum_{i=0}^{N_g - 1} E(g-i). \qquad (3)$$

- Optimization Speed V_E :

$$V_E(g) = A_E(g-1) - A_E(g) \qquad (4)$$

- When the optimization speed V_E becomes less than a **speedo−threshold** ε_E, the mutation is activated.

$$V(g) < \varepsilon_E.$$

- The optimization sometimes does not advance for several generation cycles right after the mutation. Then, the mutation is prohibited for **guard interval G_G** generation cycles right after the mutation.

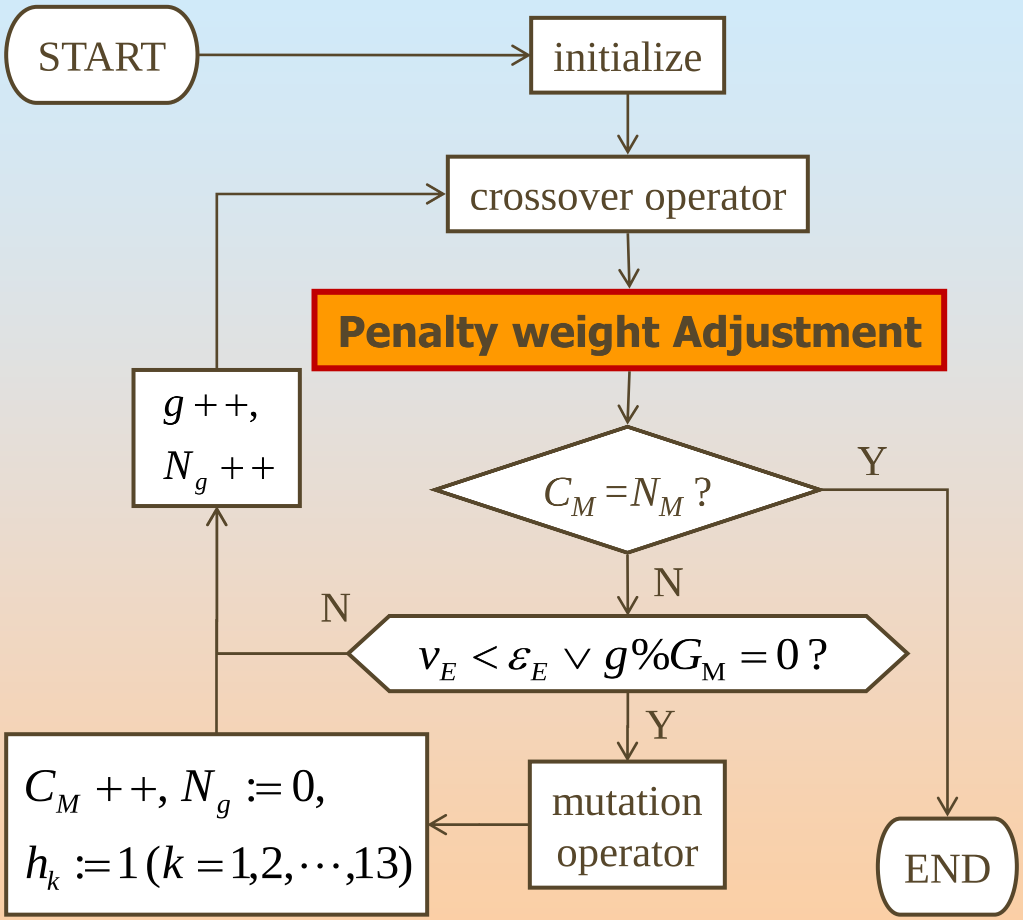
- When the mutation is executed N_M times, the optimization finishes.

4. Penalty Weight Adjustment

4. Penalty Weight Adjustment

- Re-optimization of the schedule is very hard task even by the parallel computing and then requires very long computing time.
- We consider that this problem is caused by the complexity of the solution space. (There are many local minima.)
- When the optimization is caught in a local minimum, some penalties stagnate decreasing as still greater value.
- If the shape of the **solution space** can be deformed, the searching point can **escape** from the local minimum.
- The shape of the solution space is defined by E.
- By **changing the penalty weights**, the shape of the solution space is deformed.

4. Penalty Weight Adjustment



Optimization flow with the Penalty weight Adjustment

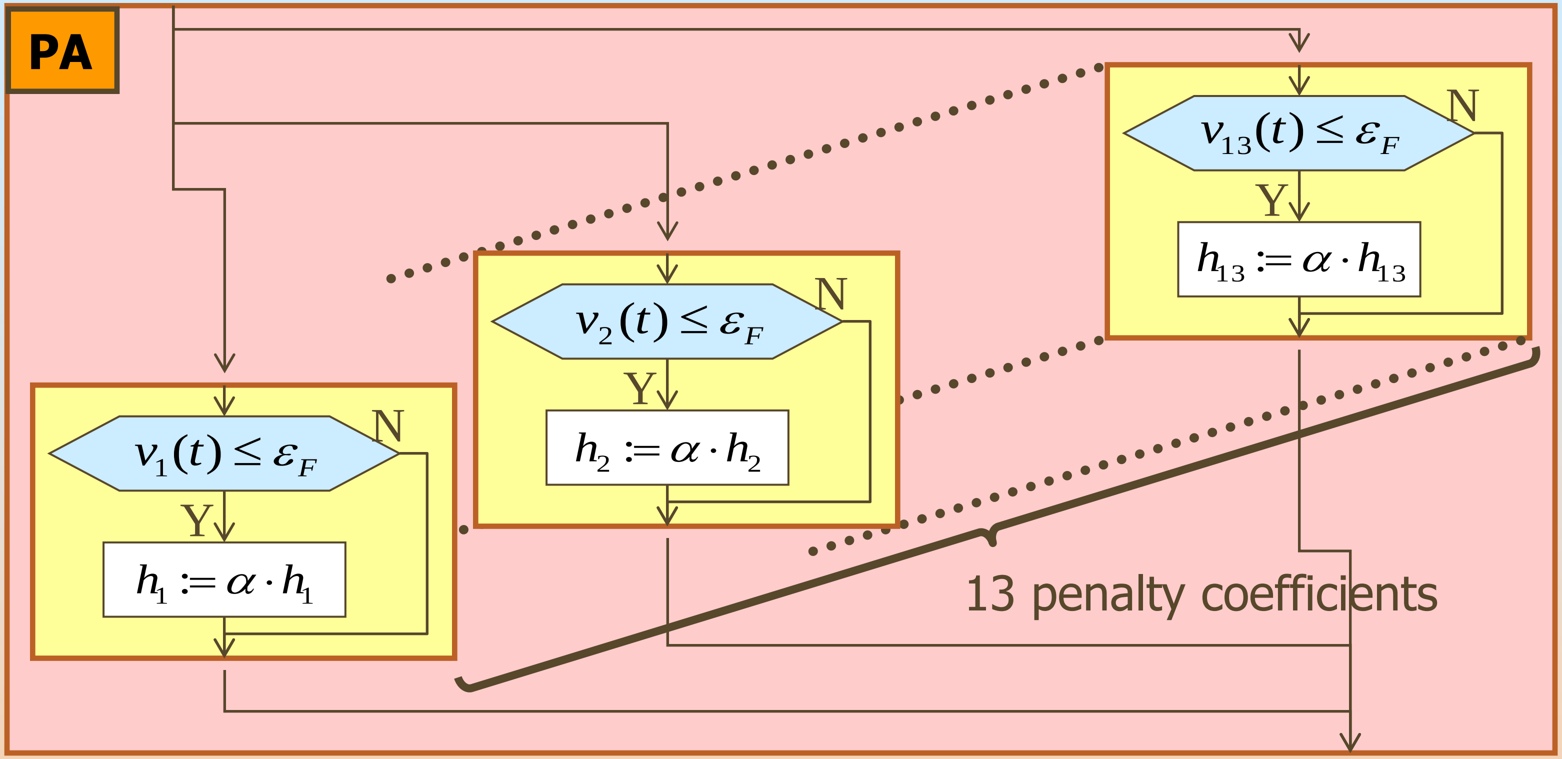
4. Penalty Weight Adjustment

- Initially, all penalty weights h_1---h_{13} are initialized to 1.
- **Decreasing speed**, v_k, of the k–th penalty is defined as

$$A_k(g) = \begin{cases} \dfrac{1}{N_g} \displaystyle\sum_{h=0}^{N_g-1} \sum_{i=1}^{M} F_{ki}(g-h) & (k \le 6) \\ \dfrac{1}{N_g} \displaystyle\sum_{h=0}^{N_g-1} \sum_{j=1}^{D} F_{kj}(g-h) & (k > 6) \end{cases} \quad (5)$$

$$v_k = A_k(g-1) - A_k(g). \quad (6)$$

4. Penalty Weight Adjustment



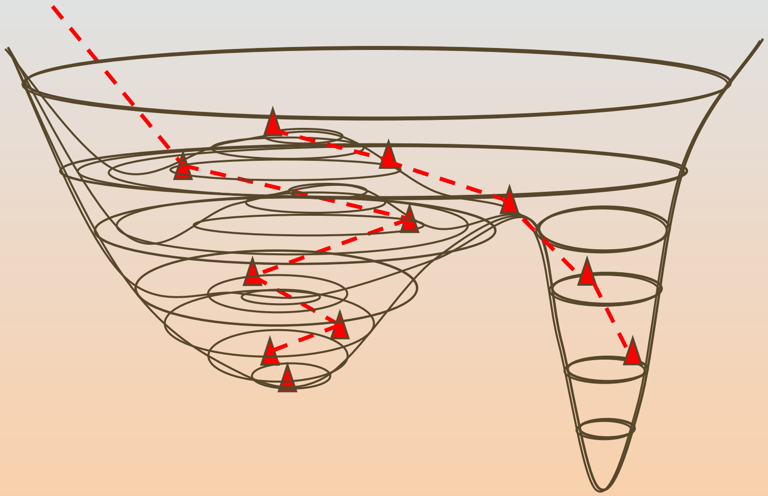
- When the **decreasing speed**, v_k, becomes less than or equal to a **speed threshold** ε_F, the penalty weight h_k is increased by multiplying with α . (ε_F=0.01, α =1.01)

- When the mutation is activated, all penalty weights are initialized to 1.

4. Penalty Weight Adjustment

★Effect of Penalty Weight Adjustment

Optimization stagnates… ⇒ caught in a wide local minimum region



Valley of the local minimum upheaves

Searching point escapes from the local minimum area.

5. Practical Experiment of Nurse Scheduling

5. Practical Experiment of NS

- The number of nurses : 23
- We suppose that two weeks have passed. In the past two weeks, two changes (one emergency attendance, one unplanned absence) exists.

- Guard interval G_{G} : 50.
- Speed threshold ε_E : 0.01.
- Mutation number N_M : 500. (PA1)
- Mutation period G_M : 2000 (GM)

5. Practical Experiment of NS

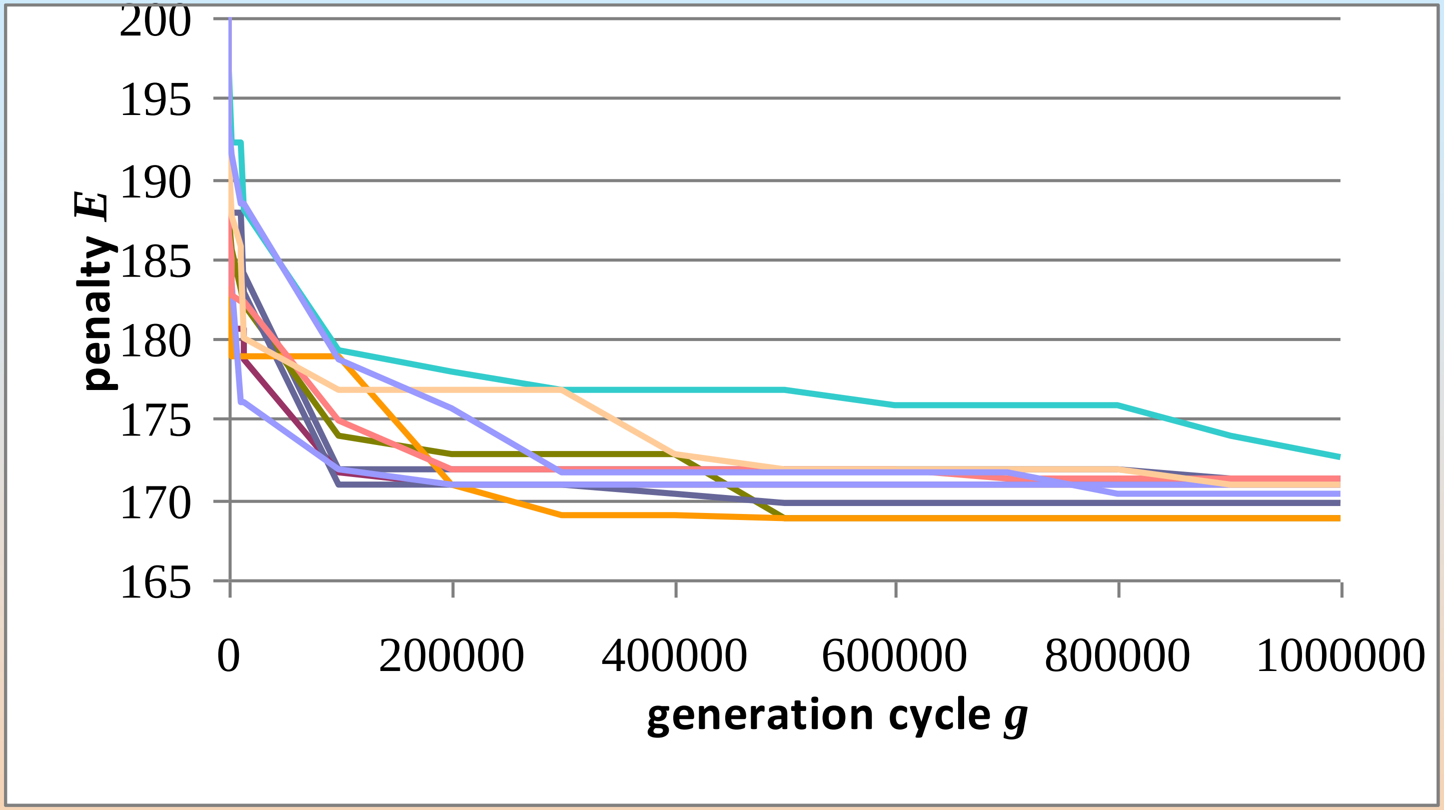


Fig.5: Ten optimization progresses when the periodic mutation is applied. The optimization is executed for 1,000,000 generation cycles. (**GM**)

- This result is almost same as the result given by CGA with the mutation depending on the optimization speed.

5. Practical Experiment of NS

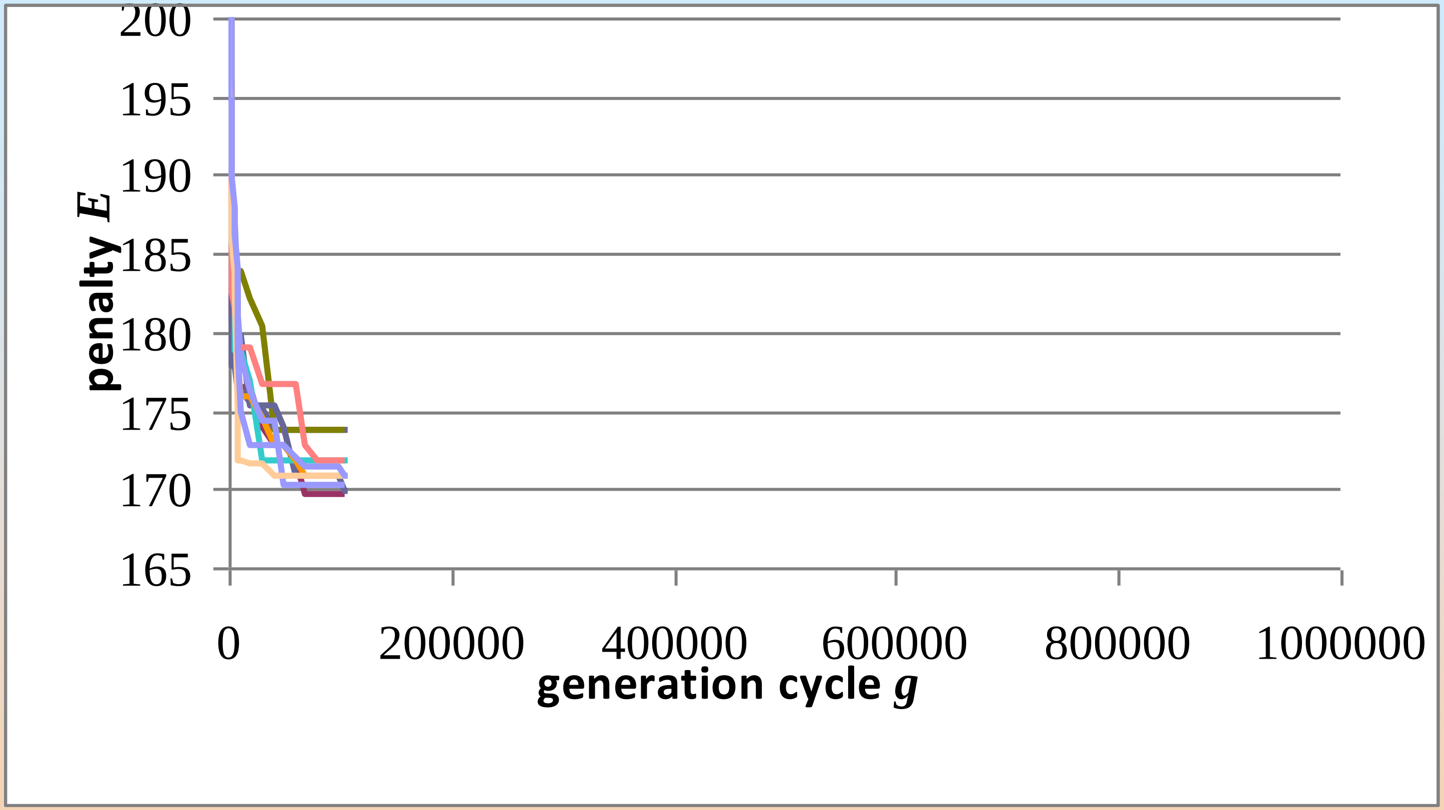


Fig.6: Ten optimization progresses when the mutation depending on the optimization speed with the **Penalty weight Adjustment** is applied. (**PA1**)

- The mutation period has been shorten by PA. Then the optimization has been accelerated.

5. Practical Experiment of NS

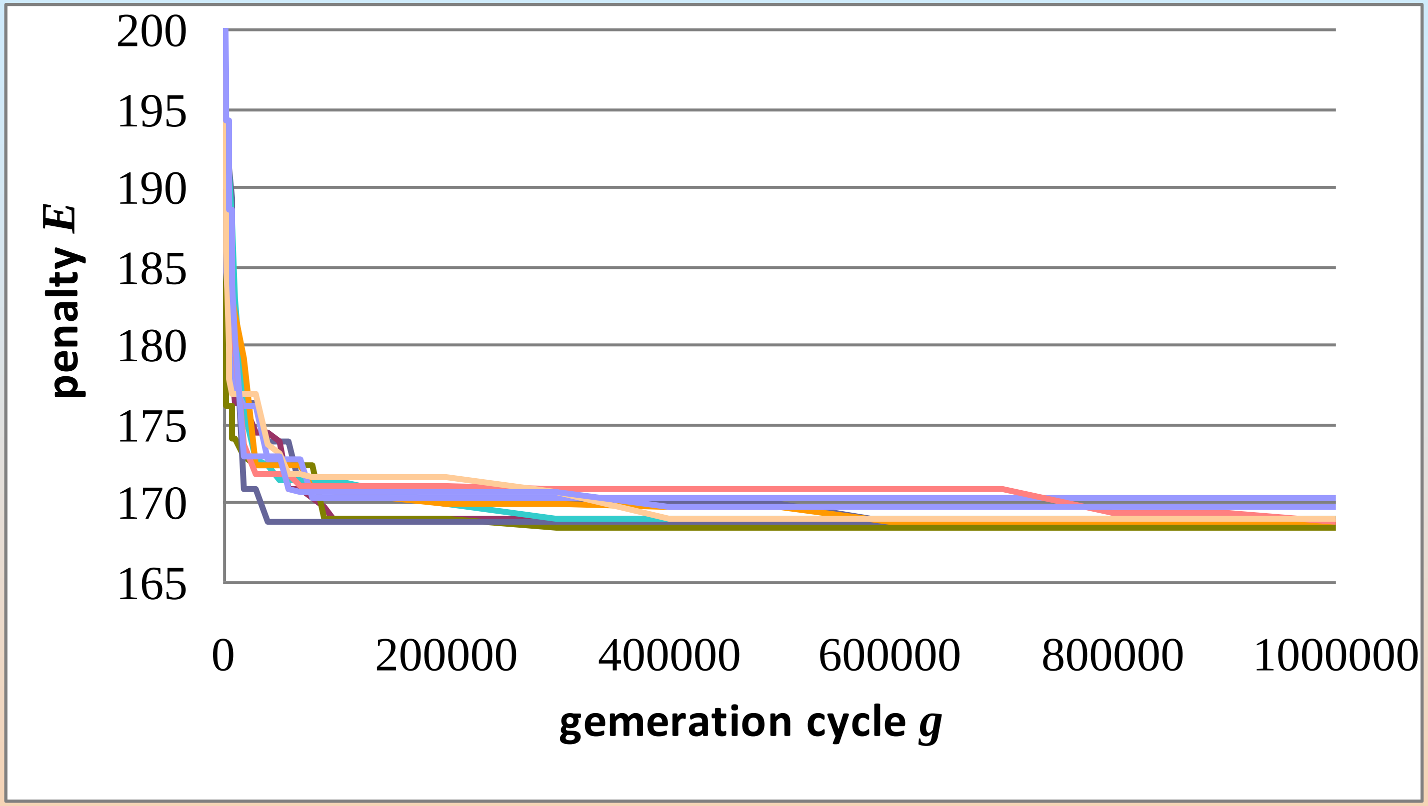


Fig.9: Ten optimization progresses when the mutation depending on the optimization speed with the **Penalty weight Adjustment** is applied. The optimization is executed for 1,000,000 generation cycles. (**PA2**)

5. Practical Experiment of NS

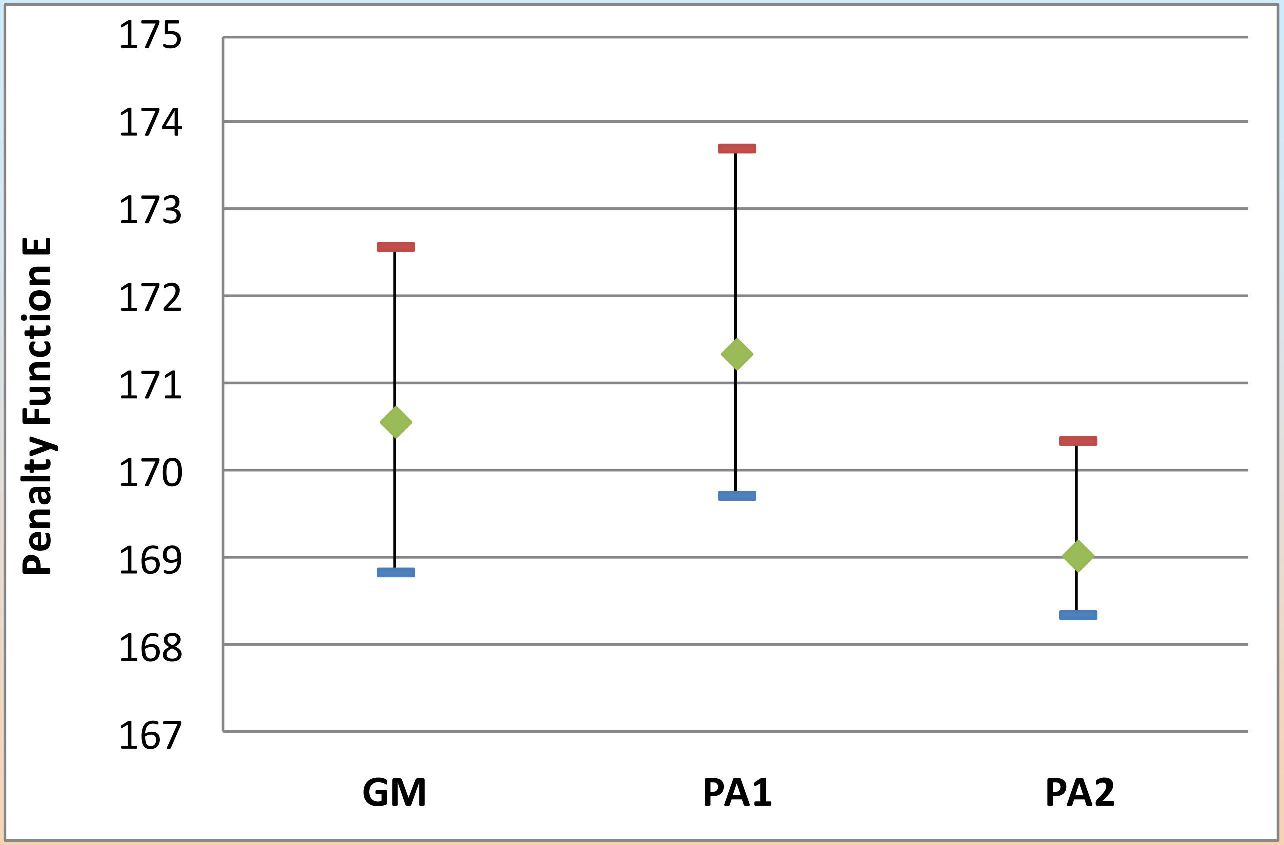


Fig.9: Comparison of the optimization results between the **periodic mutation (GM)** and the mutation depending on the optimization speed with the **Penalty weight Adjustment (PA1, PA2)**.

5. Conclusion

5. Conclusion

- In this research, we have treaded the nurse scheduling by using **CGA**.
- We have handled the case when the shift schedule has been changed in the middle of a month.
- The nurse scheduling including such changes become **difficult**, and the enormous generation cycles has been necessary to provide a good schedule.
- To improve the difficulty, we have proposed the mutation operator depending on the optimization speed and the **Penalty weight Adjustment**.
- CGA with **PA accelerates the optimization** of the nurse schedule, because it can search solution space effectively.

Thank you very much
for your kind attention!

Ask me **symplly**!
(Do not ask a complex question!)