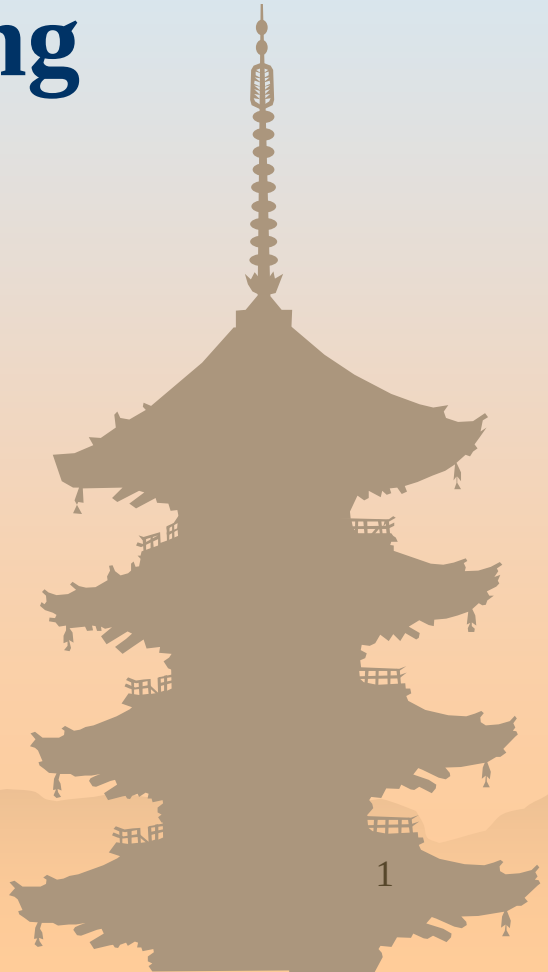
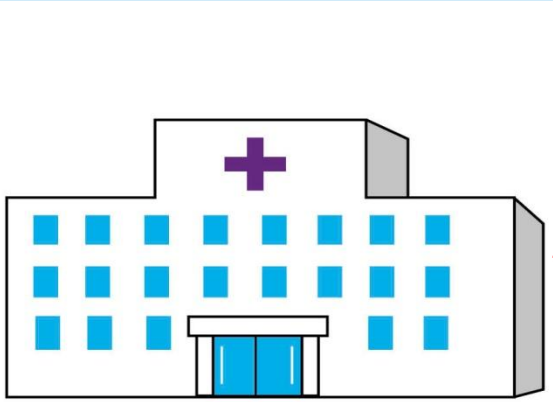
Parallel Processing of Cooperative Genetic Algorithm for Nurse Scheduling



1. Introduction



general hospital

surgery

internal medicine

pediatrics

obstetrics

- A general hospital consists of several sections.
- 15-30 nurses belong to each section.
- A director of the section arrange a shift schedule of the nurses every month.

1. Introduction

many requirements

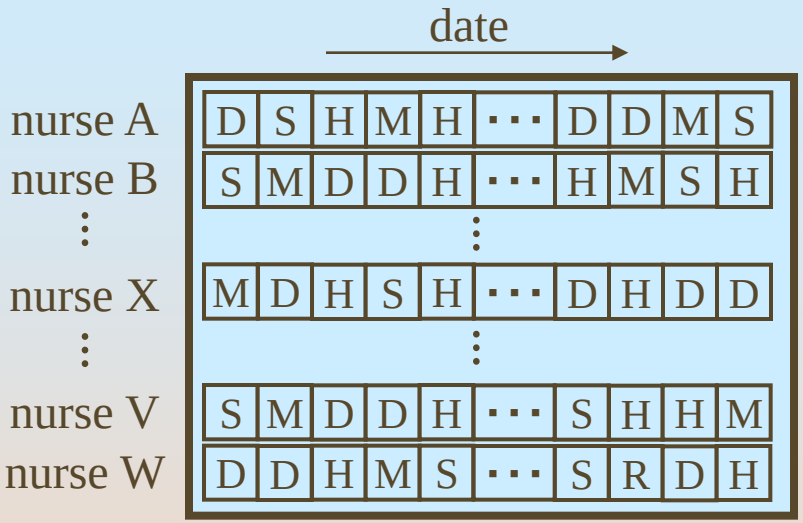
- requirements on the hope holiday.
- duty load in equality.
- the number of the night shift in equality.
- affinity between the nurses in the night shift.
- etc.

Veteran director requires one or two weeks.

Automatic Nurse Scheduling

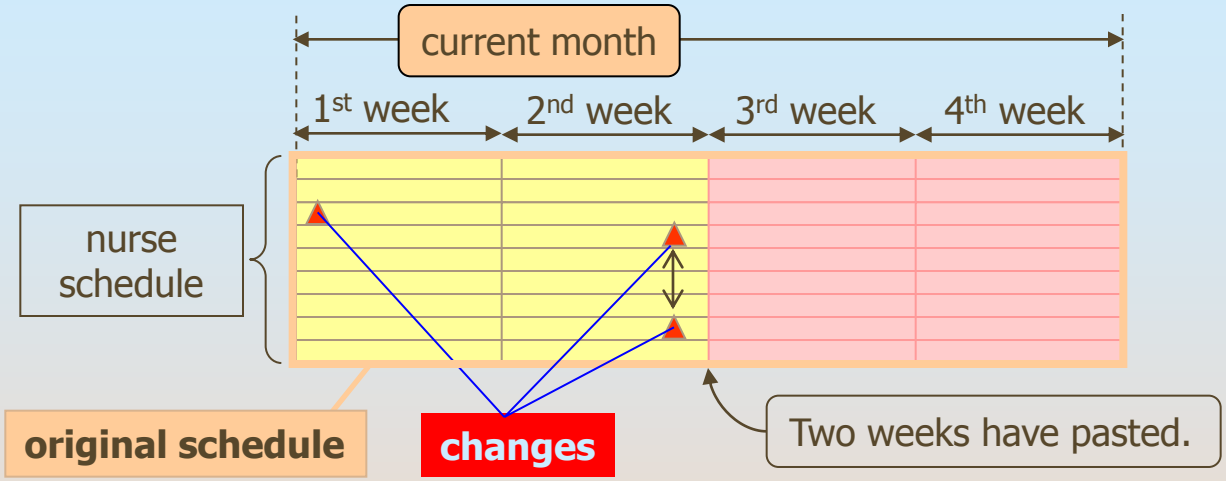
- The nurse scheduling is very complex task, because the director consider many requirements for the scheduling.
- In our investigation, even veteran director needs one or two weeks for the nurse scheduling.
- Therefore, computer software for the nurse scheduling is strongly required.

1. Introduction



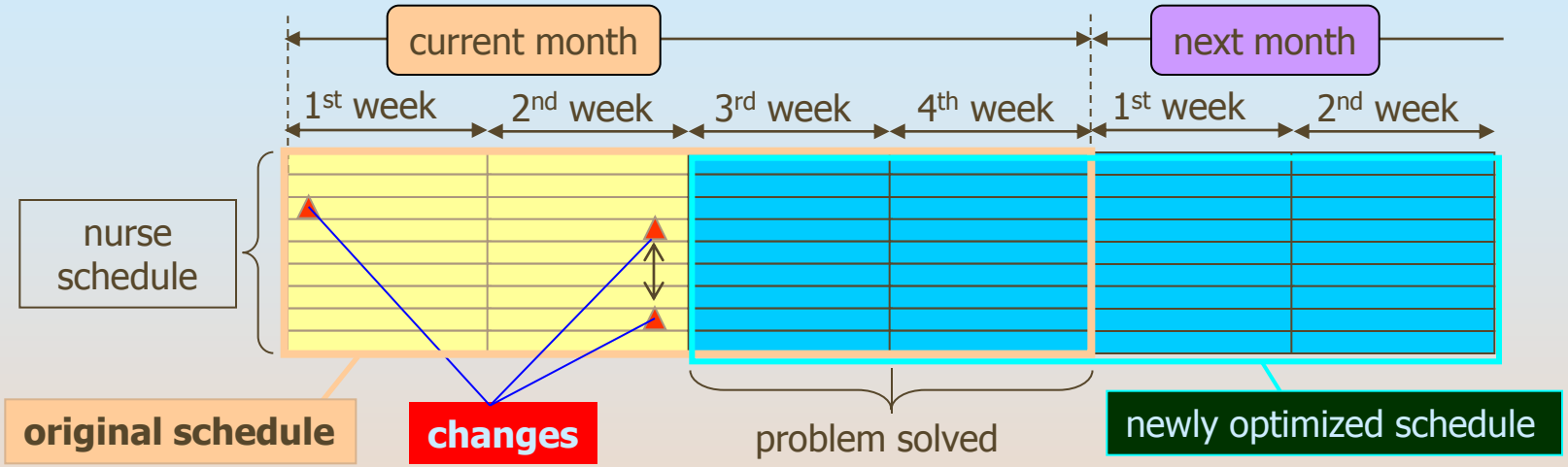
- We discuss the cooperative GA (CGA) to generate & optimize the nurse schedule.
- The conventional CGA searches solutions by using only crossover operator, because it is considered as the only one operator keeping consistency of the population.
- A mutation changing small parts of the population is very important in the GA optimization.
- We have proposed an effective mutation operator for the CGA.

1. Introduction



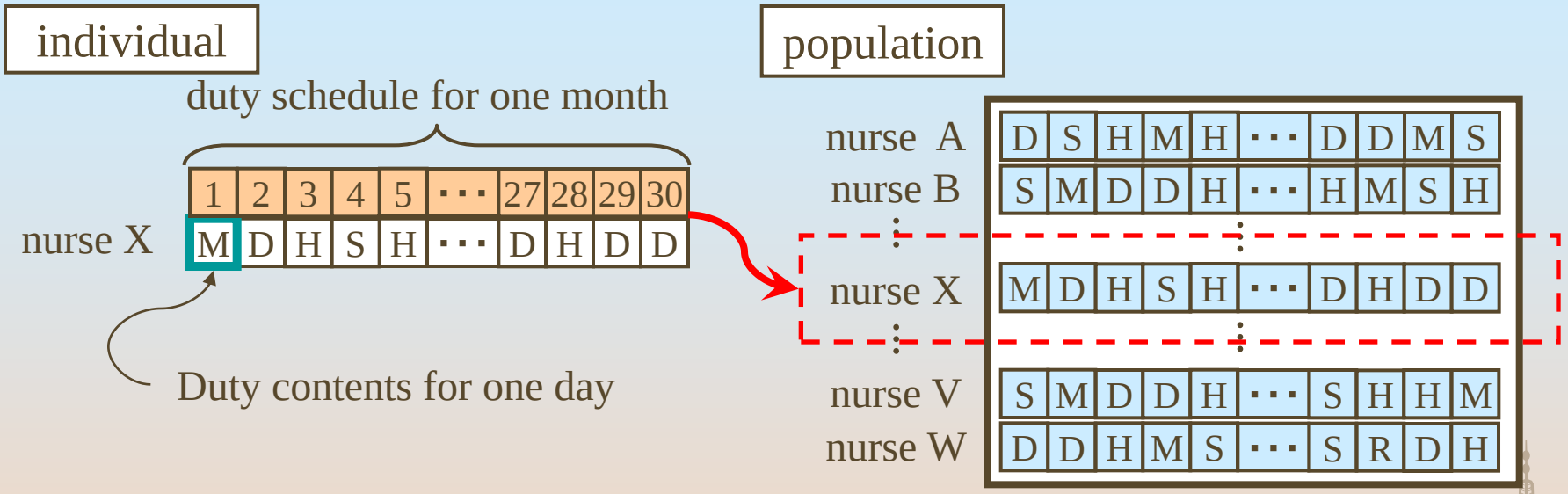
- We can consider a case when the schedule must be changed.
  - the unplanned absence of a certain nurse
  - the unplanned attendance of a certain nurse
  - the new assignment of a new face
  - the resignation of a certain nurse
  - the replacement of the shift of certain two nurses
- By means of the changes, several inconvenience occurs, for example, defections of duty days and holidays.

1. Introduction



- The changed schedule must be re-optimized to break off the inconvenience as much as possible.
- In this research, we treat the re-optimization of the changed nurse schedule.

2. Cooperative GA



- An individual consists of the sequence of the duty symbols.
- The individual shows the one-month schedule of the nurse X.
- Gathering all the individuals, the population is formed.
- In the CGA, there are not two or more individuals giving the same nurse's schedule.
- The CGA optimizes the population by using crossover and mutation operators.

2. Cooperative GA

- For the nurse scheduling, the manager consider many requirements.
- The following requirements are satisfied.
  - meeting, training and requested holiday should be accepted.
  - the number of nurses at each shift must be secured.
- The following requirements are evaluated by penalty functions.
  - duty load depending on the duty pattern. (F1)
  - 4 or more night shifts should not be assigned. (F2)
  - several prohibited duty patterns. (F3)
  - holidays and night shifts should be fairly assigned. (F4, F5)
  - more than or equal to 6 consecutive duty days. (F6)
  - nursing level must be kept. (F7, F8, F9)
  - several unfavorable combinations in the night shift. (F10)
  - two or more new faces should not assigned into the midnight shift. (F11)
  - one or more expert or more skilled nurses must be assigned into day shift. (F12)

2. Cooperative GA

- Finally, we summarize those penalties into one total penalty function.

$$E = \sum_{k=1}^{4} H_k{}^2$$

$$H_1 = \sum_{i=1}^{M} \left(h_{11} F_{1i} + h_{12} F_{2i} + h_{13} F_{3i} \right)$$

$$H_2 = \sum_{i=1}^{M} \left(h_{21} F_{4i} + h_{22} F_{5i} + h_{23} F_{6i} \right)$$

$$H_3 = \sum_{j=1}^{D} \left(h_{31} F_{7j} + h_{32} F_{8j} + h_{33} F_{9j} \right)$$

$$H_4 = \sum_{j=1}^{D} \left(h_{41} F_{10j} + h_{42} F_{11j} + h_{43} F_{12j} \right)$$

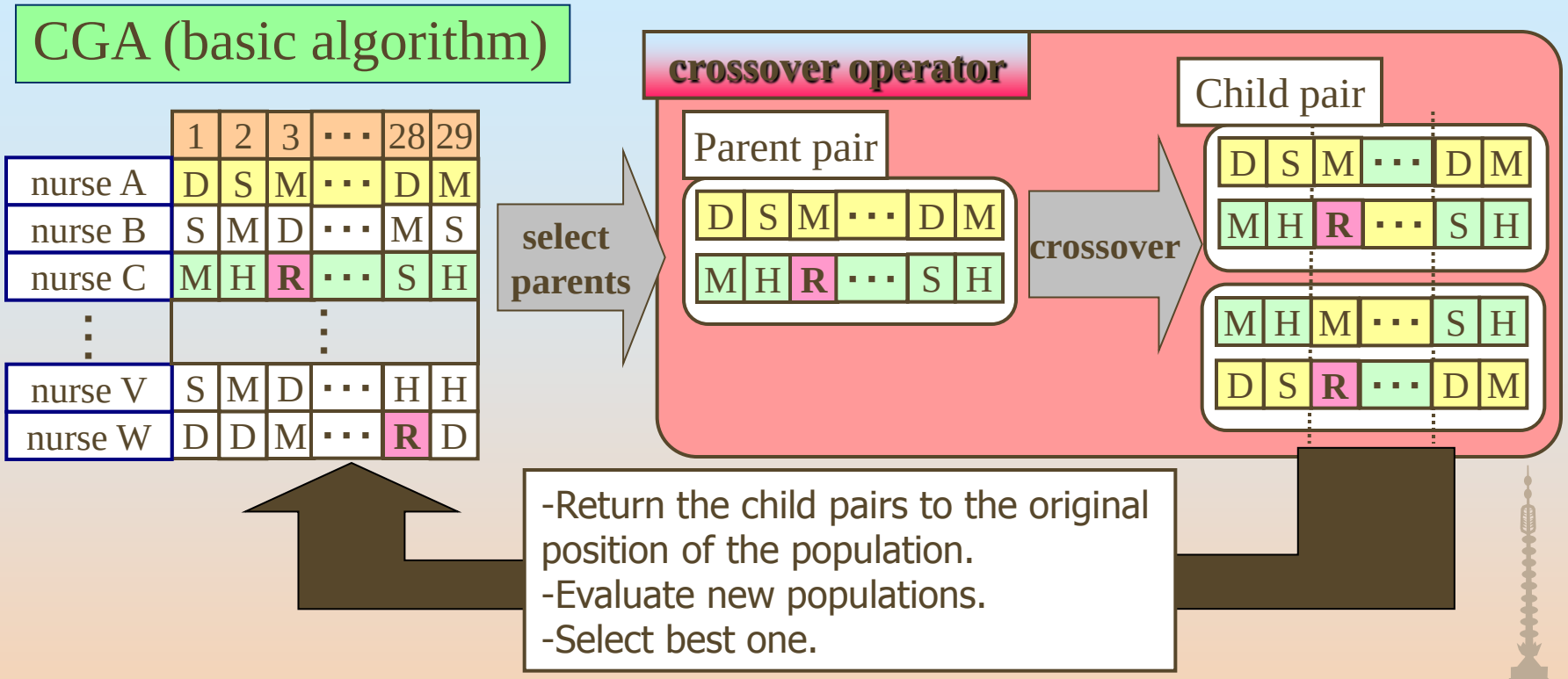
2. Cooperative GA

CGA (initialization)

- First, the CGA initialize the population.
- The requested holiday, the meeting and the training are treated as fixed duty, which CGA does not move them.
- CGA put them onto the population.
- The number of nurses in the day, the semi-night and the midnight shift are defined as 6, 3 and 3 in the application here.
- CGA randomly assigns the duty symbols satisfying the specific numbers.

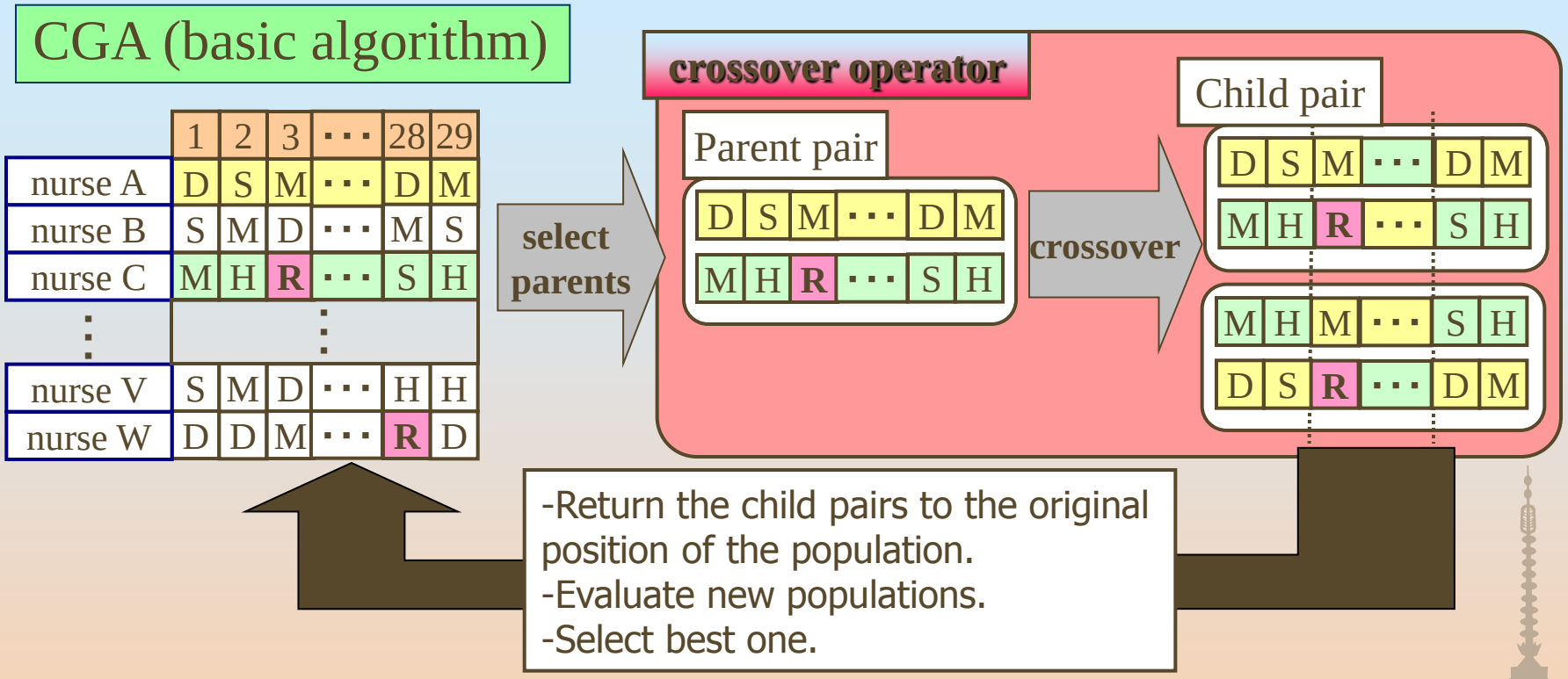
D: day shift, S: semi-night shift, M: midnight shift
R: requested holiday, H: holiday
m: meeting, T: training

2. Cooperative GA



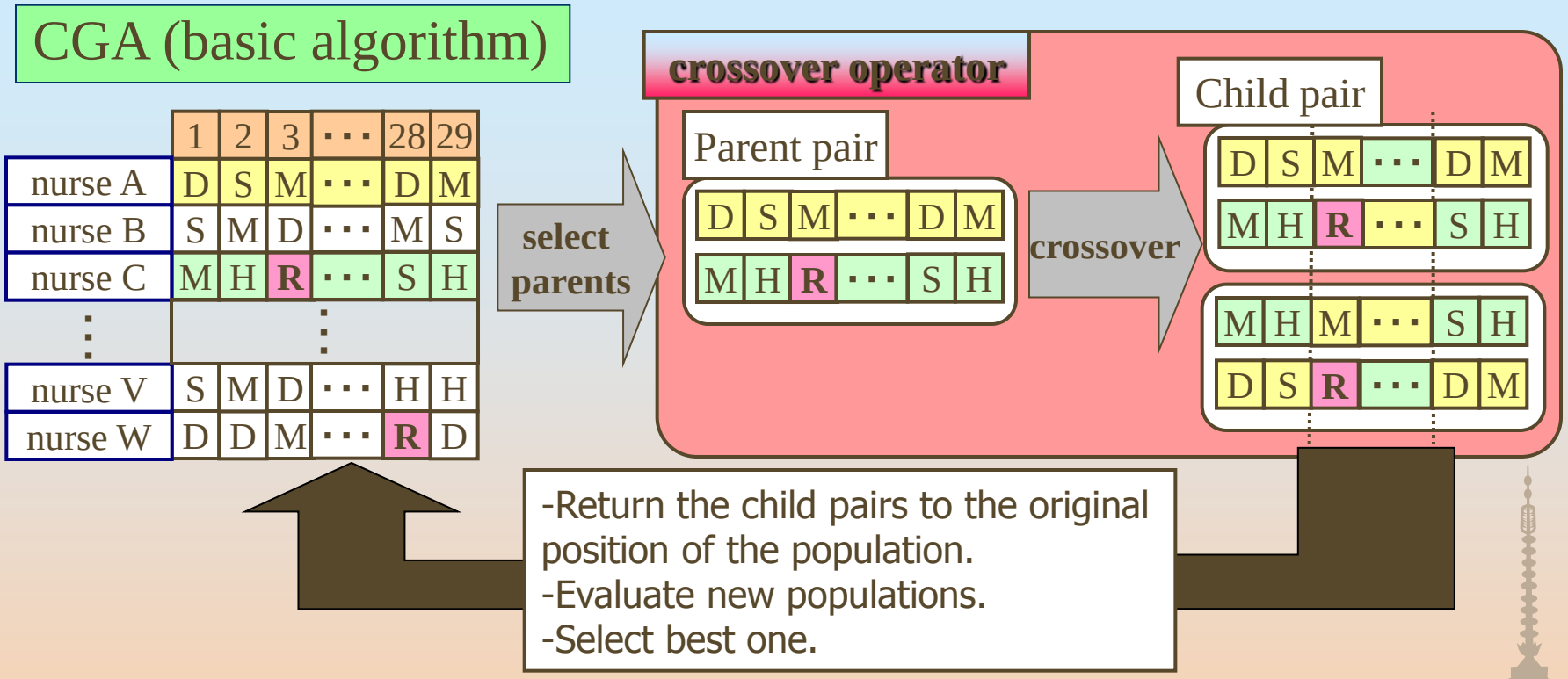
- This figure shows the basic algorithm of the CGA.
- CGA searches good solution by basically using the crossover operator.

2. Cooperative GA



- 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.

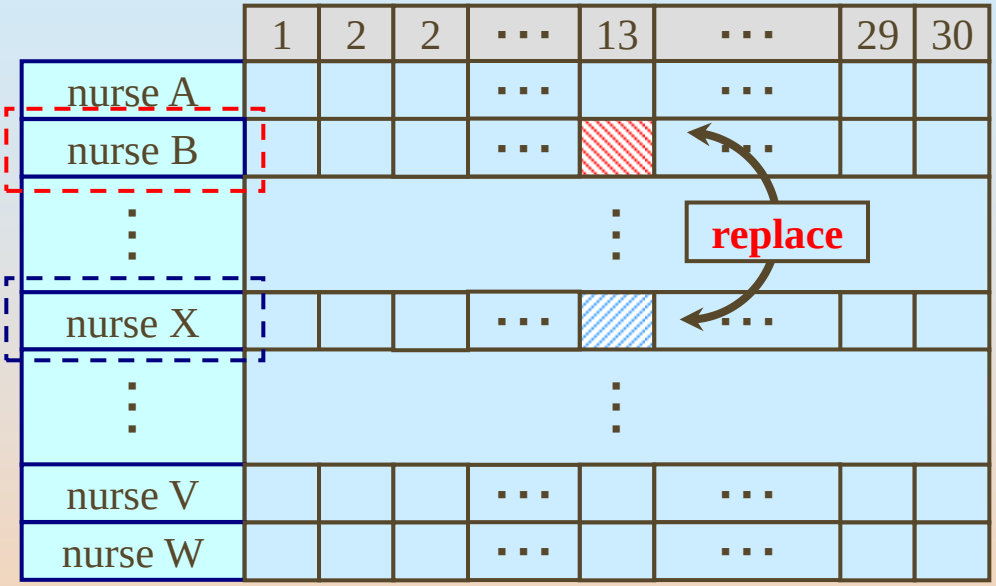
2. Cooperative GA



- Setting these new individuals back to the original positions of their parent, the population is evaluated by the function, *E*.
- This procedure is applied to 100 parent pairs in 1 generation.

2. Cooperative GA

CGA (mutation operator)



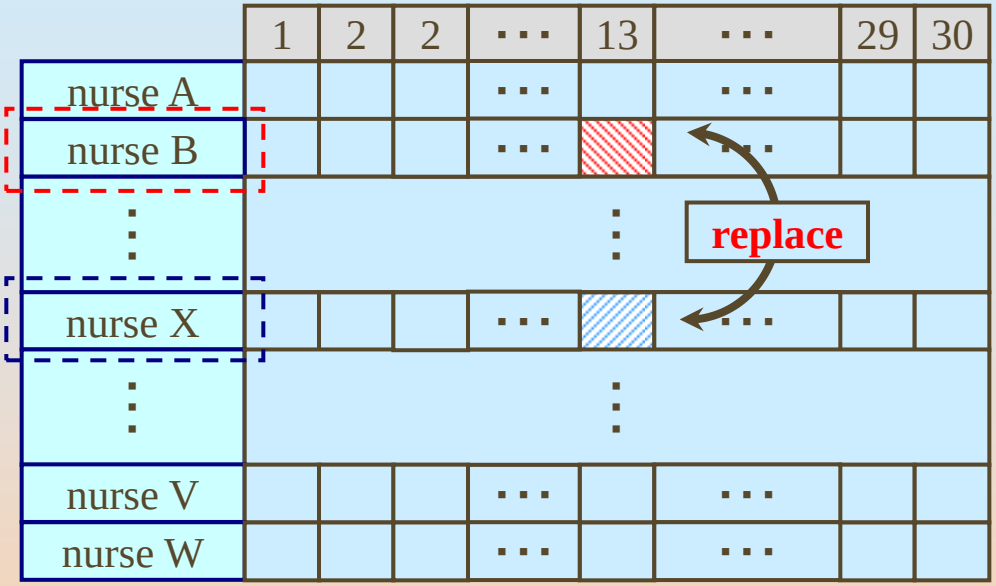
The following procedures are executed by a G_M generation period.

① Randomly select one of duty dates.

② Randomly select two nurses, where a nurse with the fixed duty is not selected.

2. Cooperative GA

CGA (mutation operator)



③ Replace these two duty contents.

④ The new schedule provided in this way is generally worsened, but a global search is enabled by receiving this forcibly.

3. Parallel Processing of the CGA

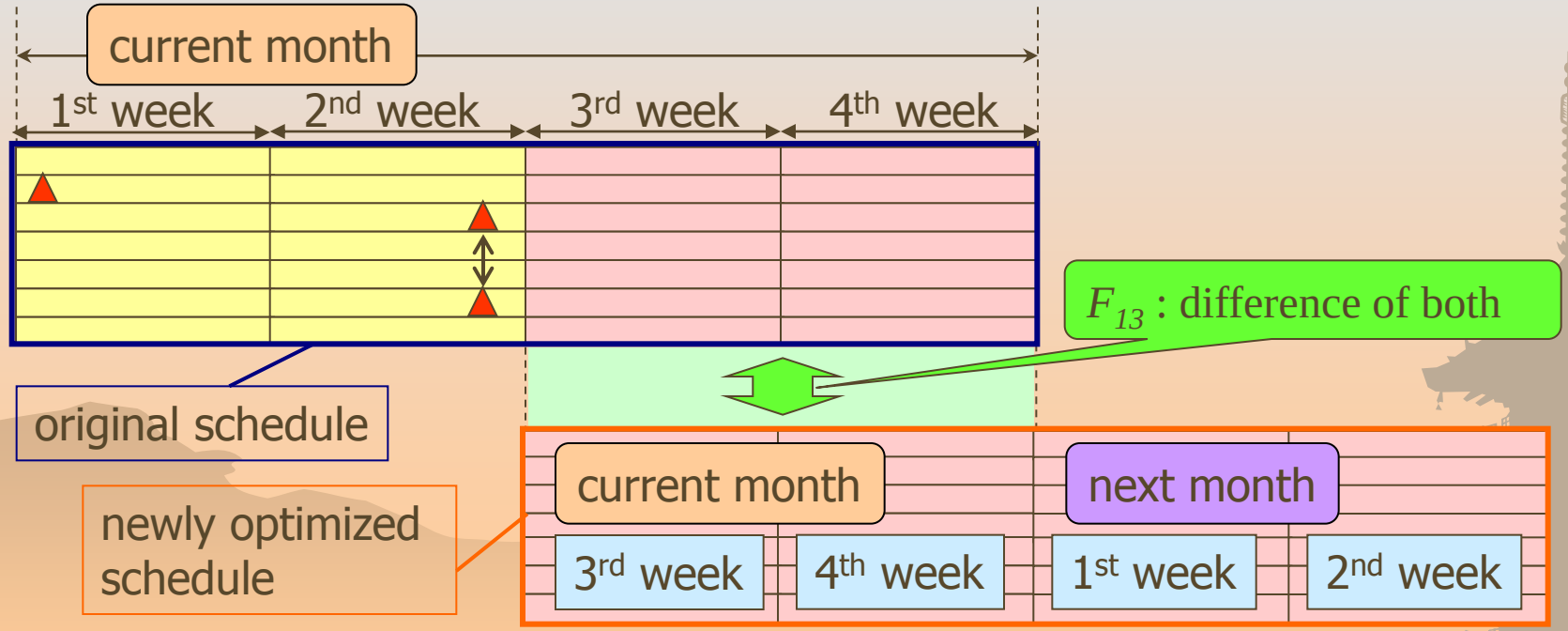
Expansion of the Nurse Scheduling

- In the real hospital, the nurse schedule is changed because of ...
  - the unplanned absence of a certain nurse,
  - the unplanned attendance of a certain nurse,
  - the new assignment of a new face,
  - the resignation of a certain nurse, and
  - the shift replacement between certain two nurses.
- Such the schedule changes lead to the nurse's load disproportion.
- This causes the overwork of specific nurses if such situation is ignored.
- Furthermore, it leads to not only the fall of the nursing level but also the medical accident.
- This research shows a **re-optimization** of the changed schedule **without changing an original schedule as much as possible**.

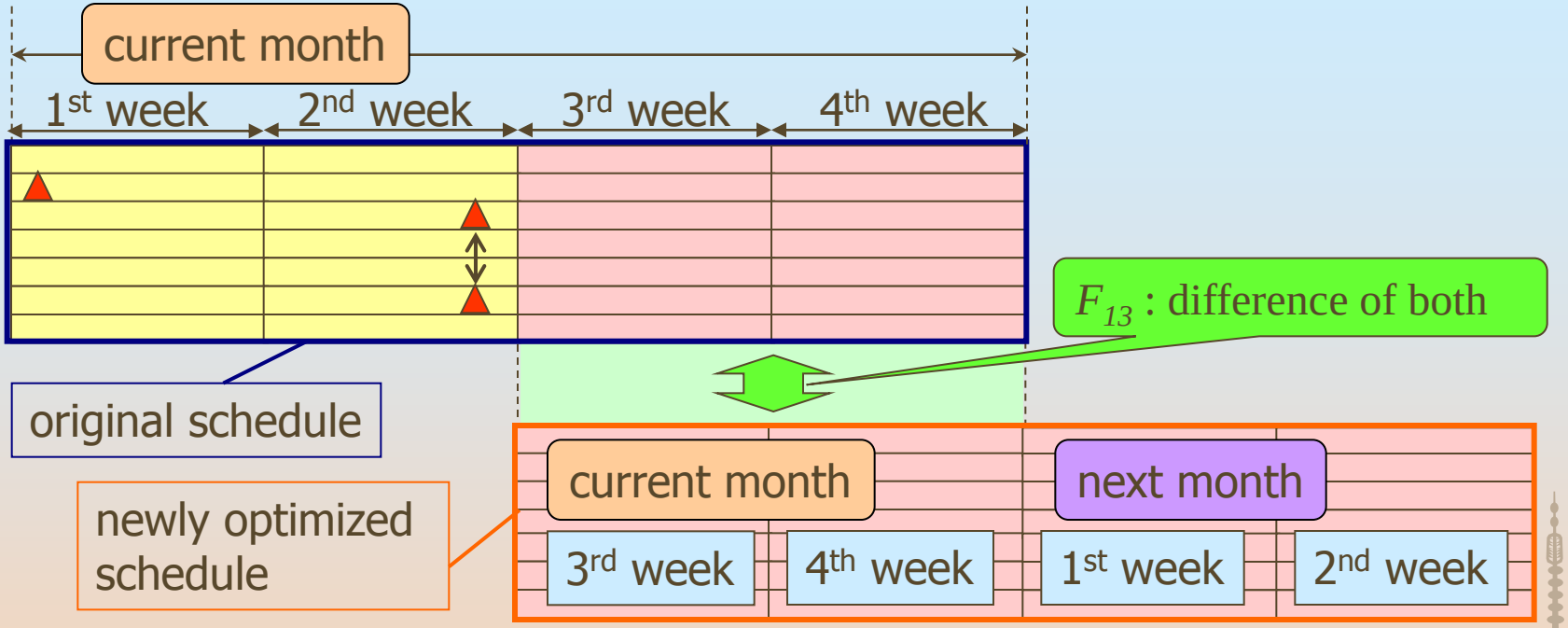
3. Parallel Processing of the CGA

Expansion of the Nurse Scheduling

- In this problem, we have two opposite requirements.
- We must change the schedule to correct disproportion, but want to keep the change of the schedule at the minimum.
- We define a new penalty function to solve this problem.



3. Parallel Processing of the CGA



- Suppose that two weeks passed.
- There are several changes in the passed two weeks.
- Now, we want to cancel disproportion by these changes in the coming four weeks.
- We define a penalty function (F13) which denotes the difference between the original and the newly optimized schedules.

3. Parallel Processing of the CGA

Parallel Processing of the CGA

Actually, we must perform several times optimization for the 1,000,000th generation to get the good schedule.

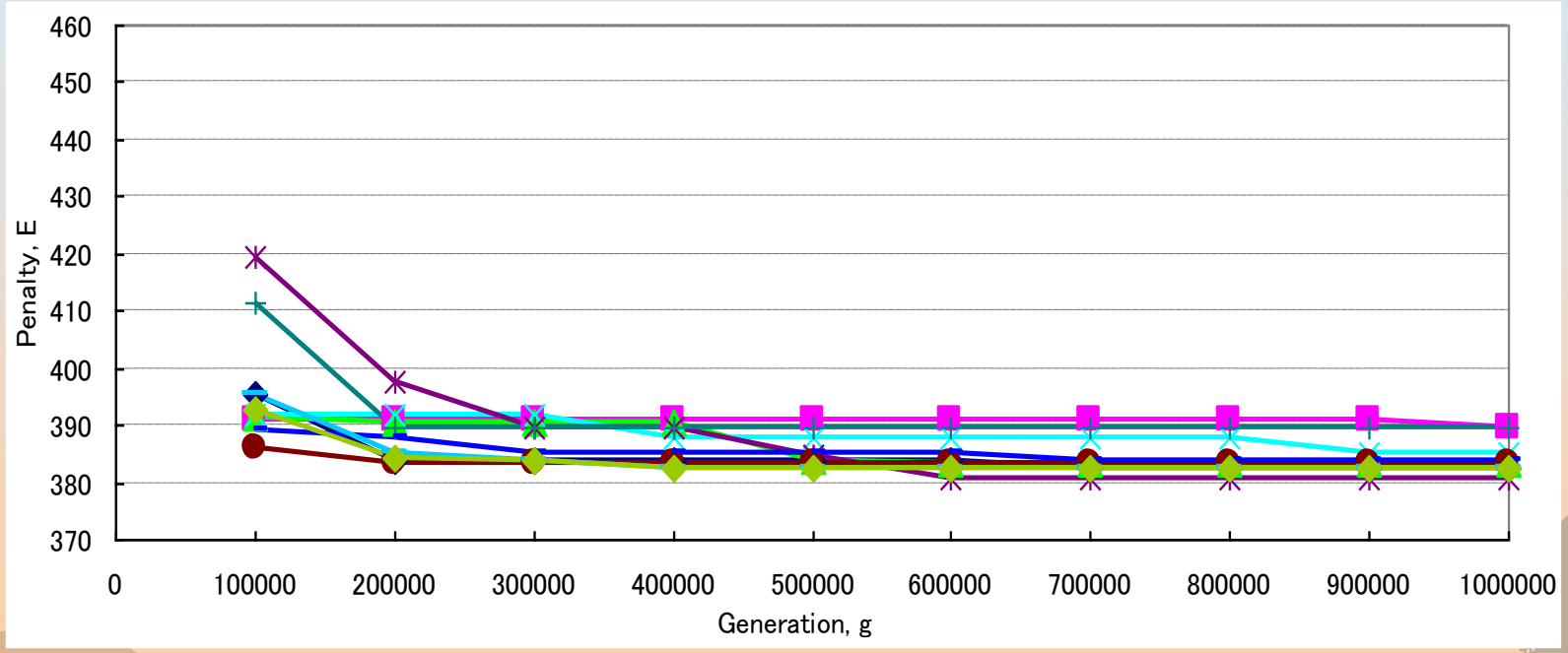
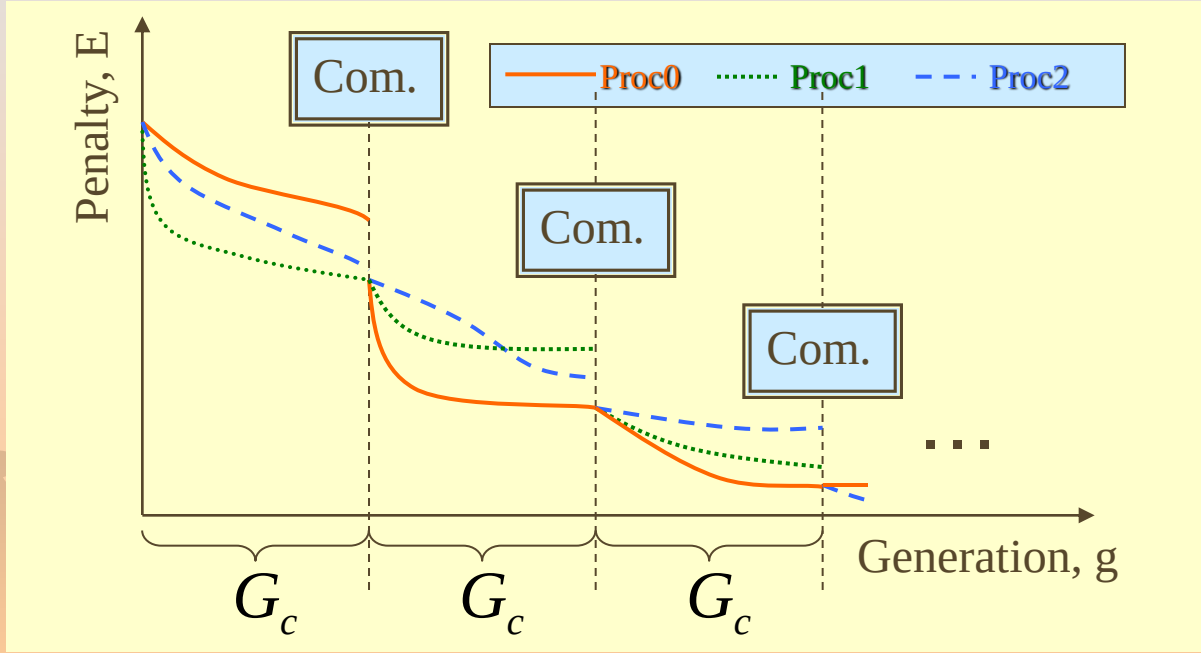


Fig. 4. An example of ten times of trials of the optimization for the 1,000,000th generation.

3. Parallel Processing of the CGA

Parallel Processing of the CGA

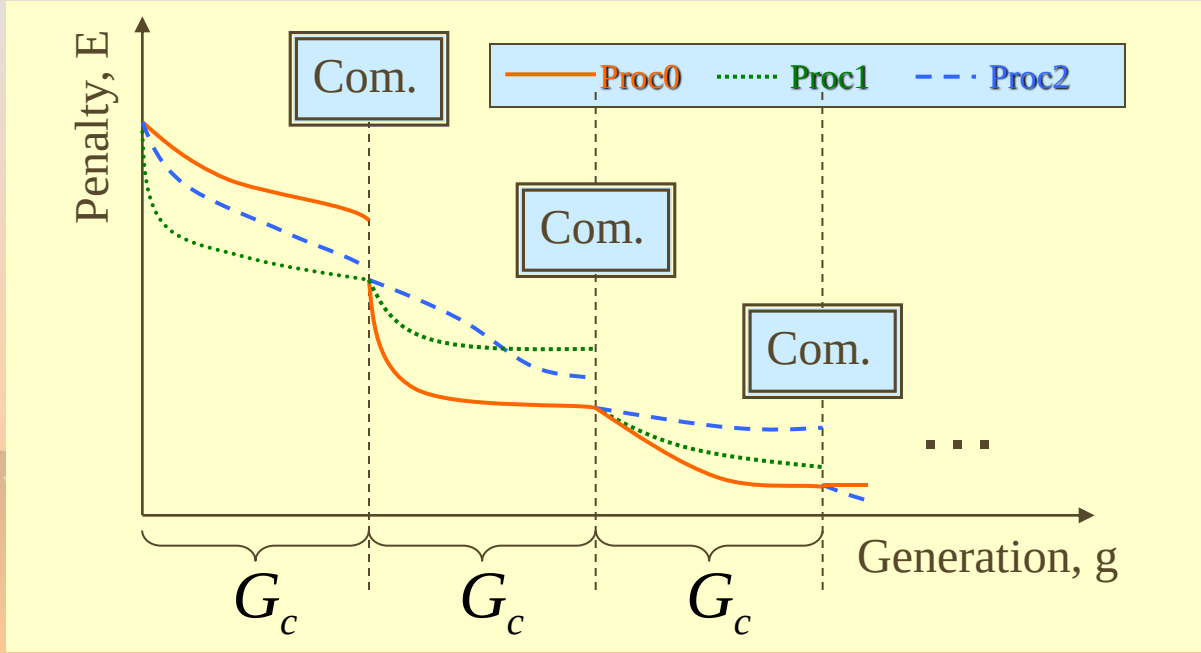
- An effective schedule may be provided in a short time if such a difficult optimization is effectively performed in parallel.
- Besides, there are a lot of PCs which are not used at the night hospital.
- Therefore, we propose a technique to execute the CGA in parallel.



3. Parallel Processing of the CGA

Parallel Processing of the CGA

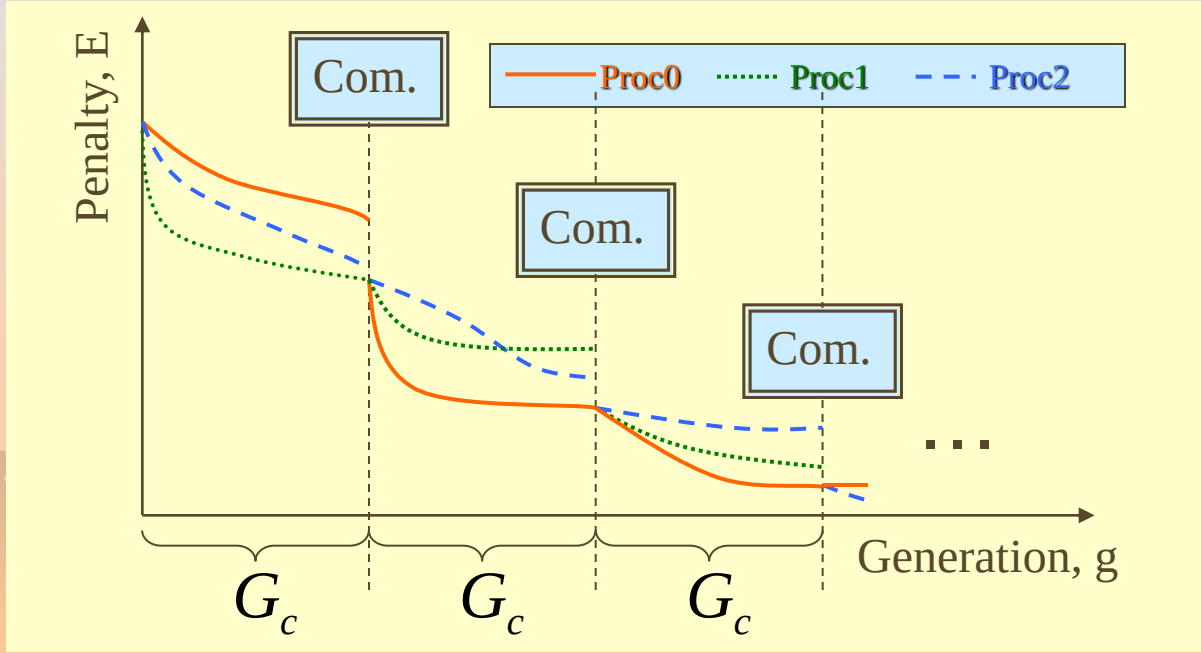
- Parallel processing with 3 CPUs are shown in this Fig.
- The first process, Proc0, is generated on a PC of which the user starts the nurse scheduling.
- The first process generates several child processes on other PCs.



3. Parallel Processing of the CGA

Parallel Processing of the CGA

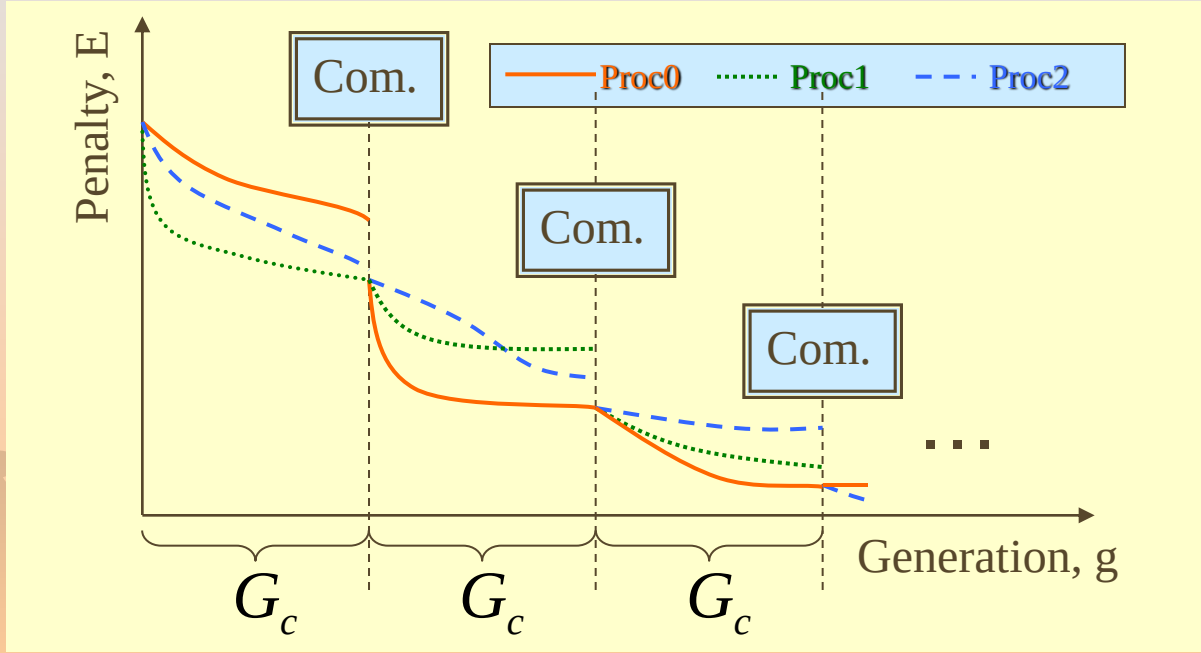
- Those processes communicate every G_C generation period.
- *GC* should be defined to a multiple of the mutation period, G_M.
- Proc0 manages the communication.
- In the communication, each process sends the best schedule acquired by the optimization of G_C generations to Proc0.



3. Parallel Processing of the CGA

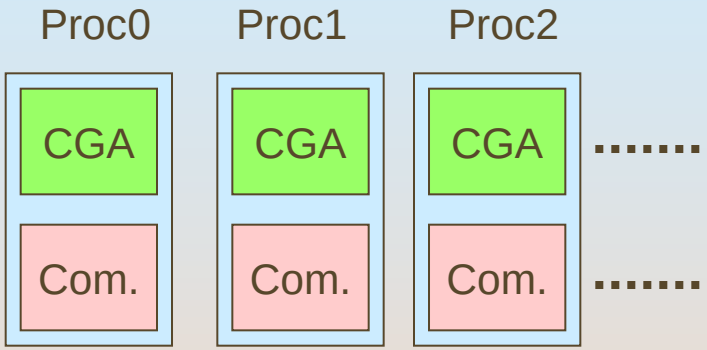
Parallel Processing of the CGA

- Proc0 selects the best schedule among them and sends it to all other processes.
- Each process starts their optimization again with the best schedule.



3. Parallel Processing of the CGA

Parallel Processing of the CGA



- We have implemented this parallel algorithm by using MPI technology.
- Each process is composes of two threads, CGA thread and Com. thread.
- The Com. thread works only when a message comes in.
- The main thread, CGA thread, always works for optimization.

4. Practical Experiment

- We have tried practical experiment of the nurse scheduling with practical data.

  the number of nurses : 23
  the number of changes : 1
  mutation period, G_M : 200
  communication period, G_C : 50000
  total generations : 1,000,000

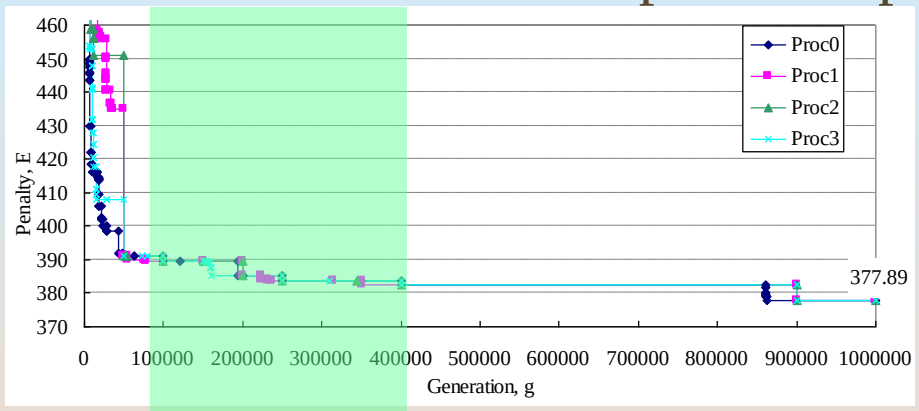
- We have prepared two PCs with two CPUs for the experiment.

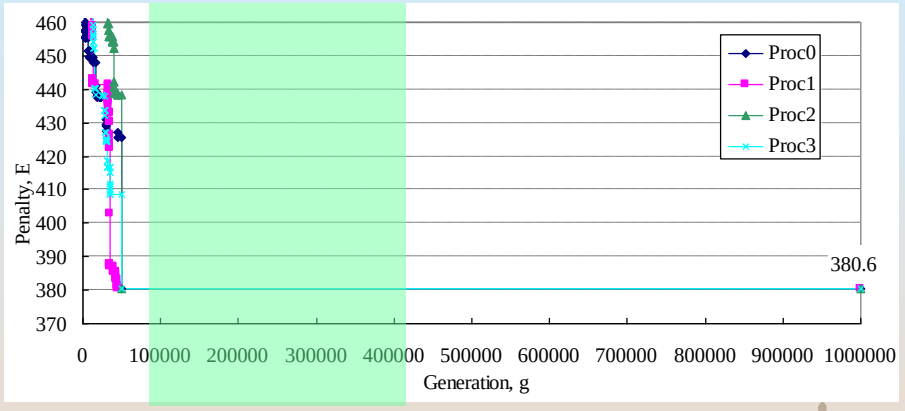
  - PC1: Intel ® core ™2 CPU 6600, 2.4GHz
    with 2GB RAM
  - PC2: AMD ® Athlon™ Dual Core Processor 4200+, 2.20GHz
    with 2GB RAM

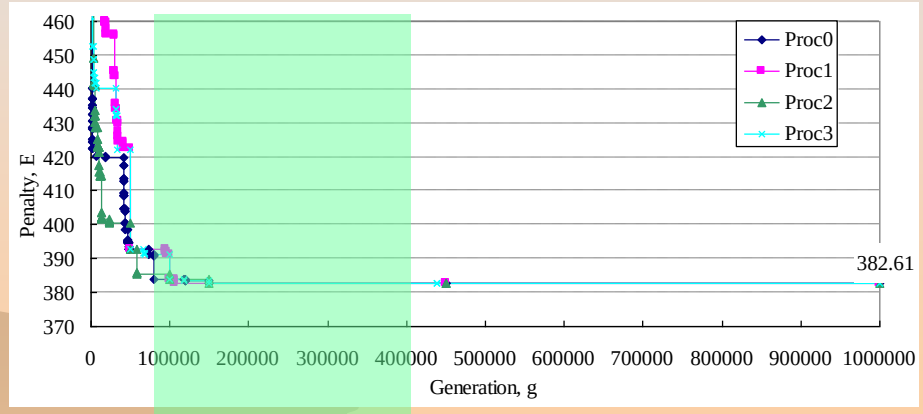
- Totally 4 CPUs are prepared for the experiment.

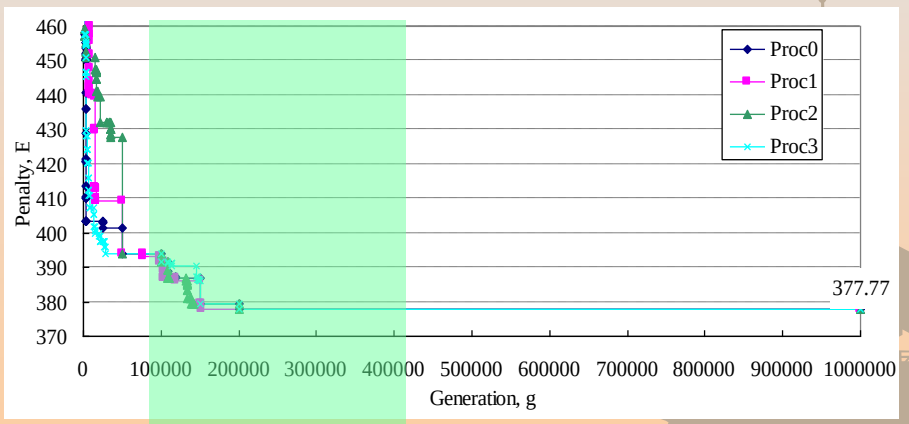
4. Practical Experiment

- We have tried the parallel optimization four times.

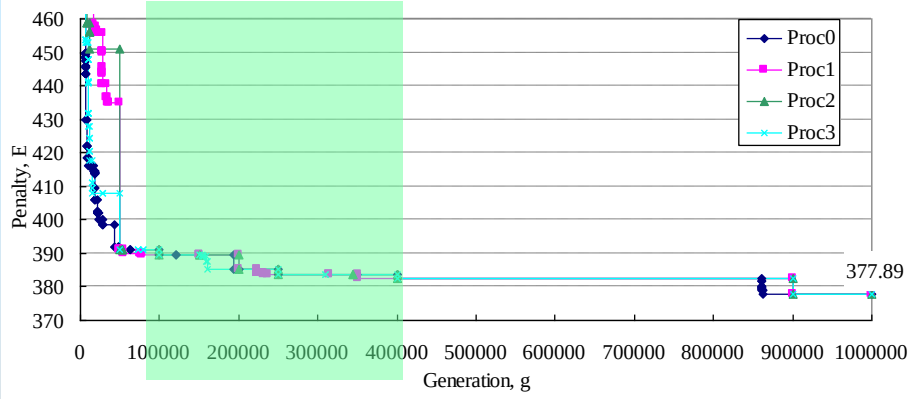


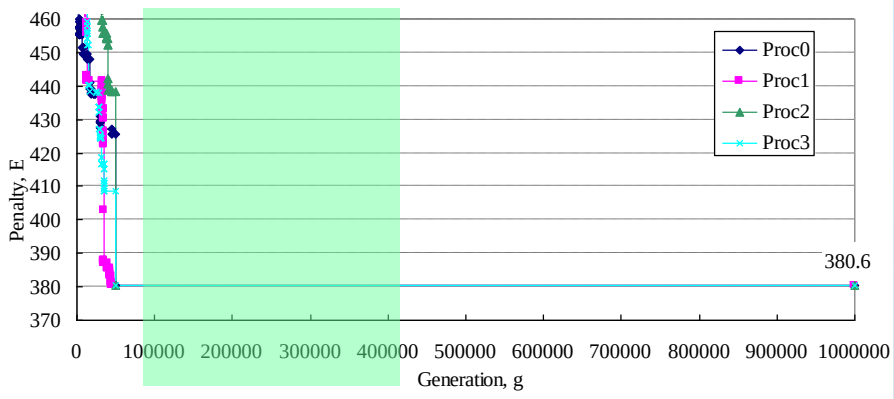


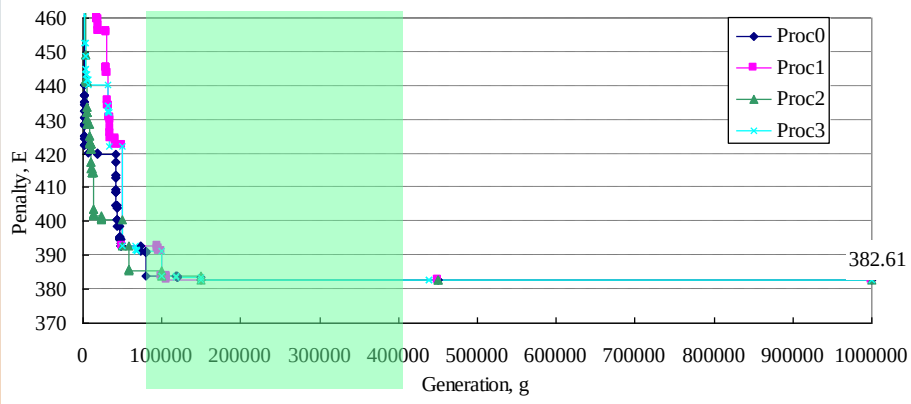


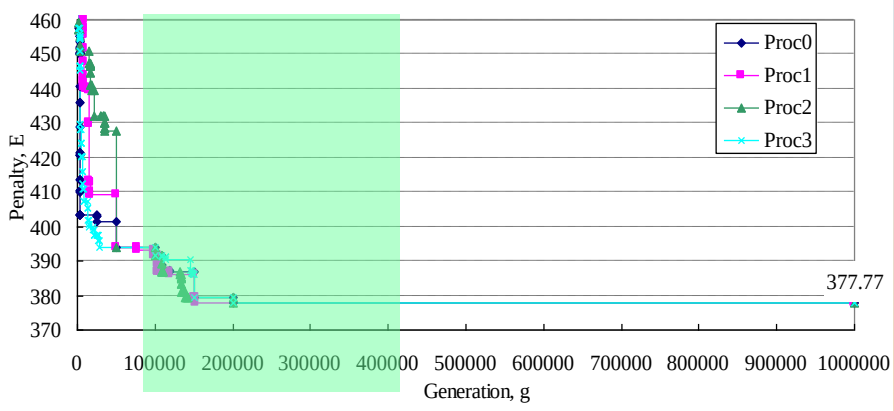


- In most cases, the parallel CGA gives the most suitable schedule from 100,000 generations to 400,000 generations.









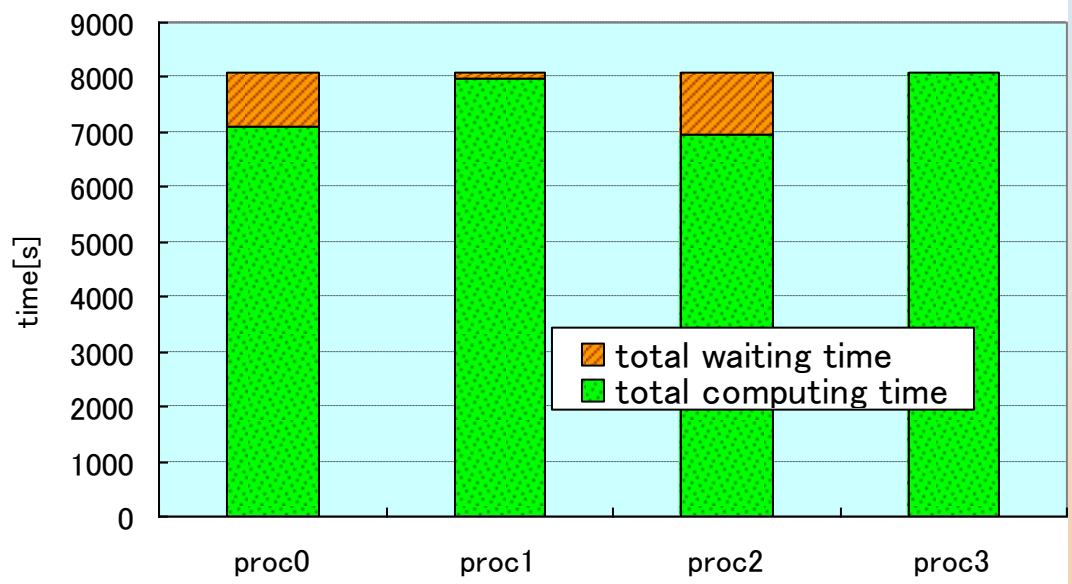
- This means that the optimization speed of the parallel CGA is about 10 times from 2.5 times of the conventional CGA.
- Furthermore, the parallel CGA has given better schedule than that given by ten times optimization by the conventional CGA.

5. Conclusion

- In this research, we have treaded the nurse scheduling by using CGA.
- The nurse scheduling is expanded to accept the changes of the shift schedule.
- By this expansion, optimization became difficult, and the number of the enormous generations has been necessary to get a good schedule.
- To provide the difficulty, we have proposed the parallel algorithm of the CGA.
- By using the parallel CGA, the better nurse schedule has been acquired in shorter computation time.

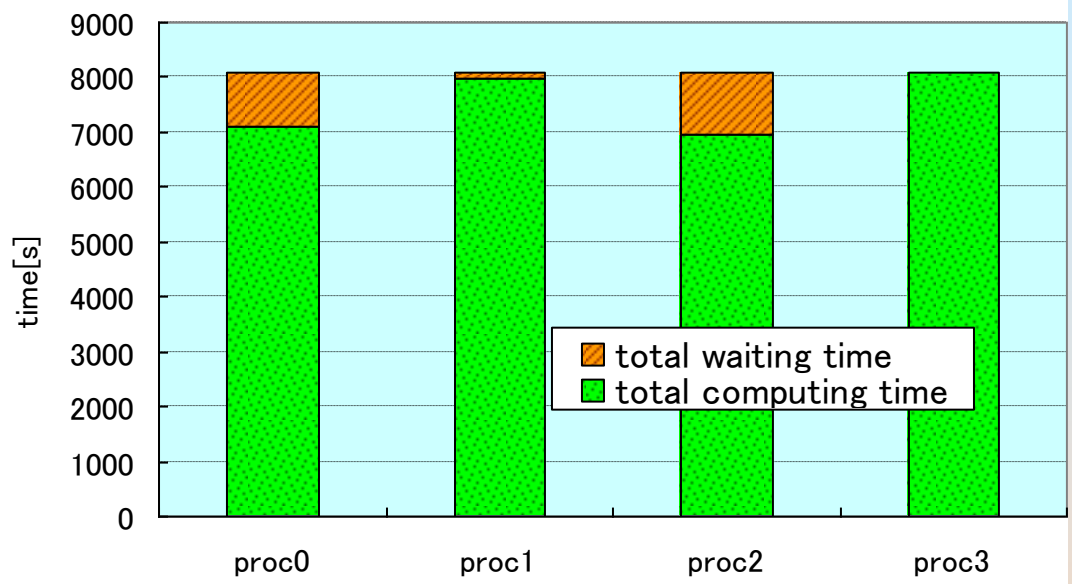
5. Conclusion

- We have applied the simple parallel algorithm to CGA.
- Therefore, some processes working on the faster CPU have to wait for the other processes working on the slower CPU.



- Proc0 and Proc2 are executed on the faster CPUs, PC1.
- On the other hand, other two processes are executed on the slower CPUs, PC2.

5. Conclusion

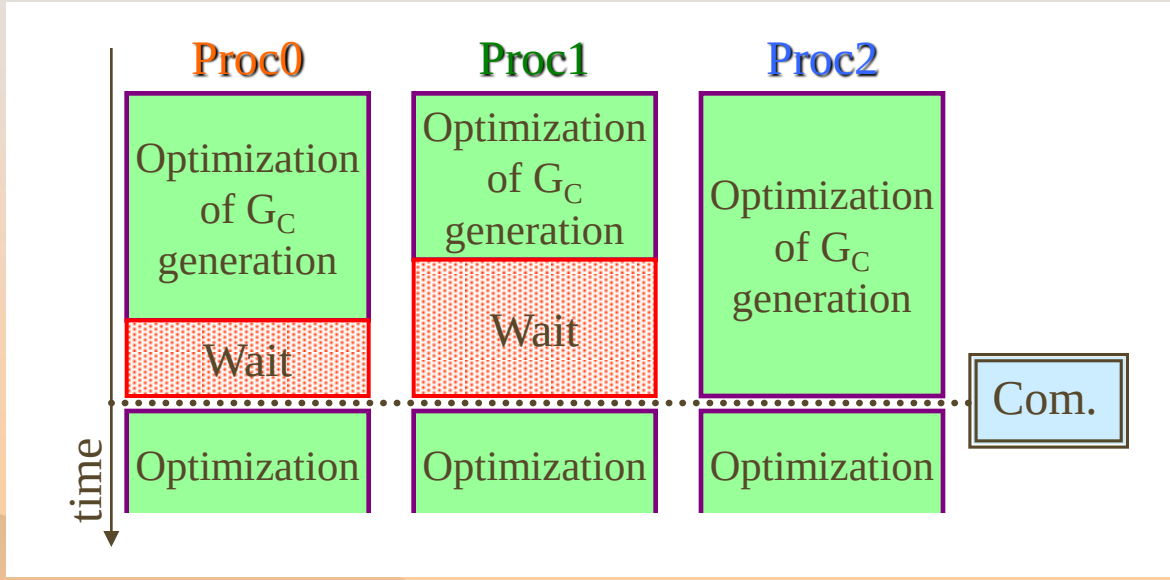


- In this experiment, the faster CPUs have consumed 12% of the computation capacity for waiting for other processes.
- We should consider an efficient parallel algorithm to reduce such an useless waiting time and to utilize CPU capacity.

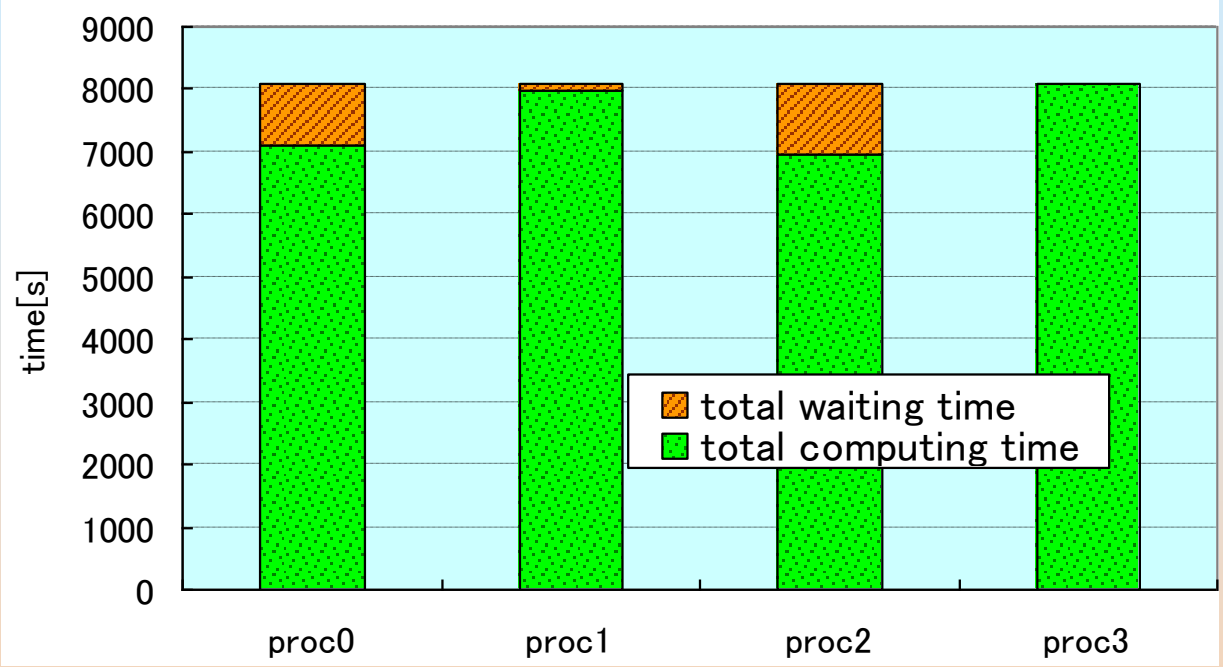
Thank you very much
for your kind attention!

Appendix: Waiting Time

- We apply the simple parallelized algorithm to CGA.
- In our algorithm, the processes on the fast CPU wait for the slow processes.
- The waiting time means the useless waste of computing resource.

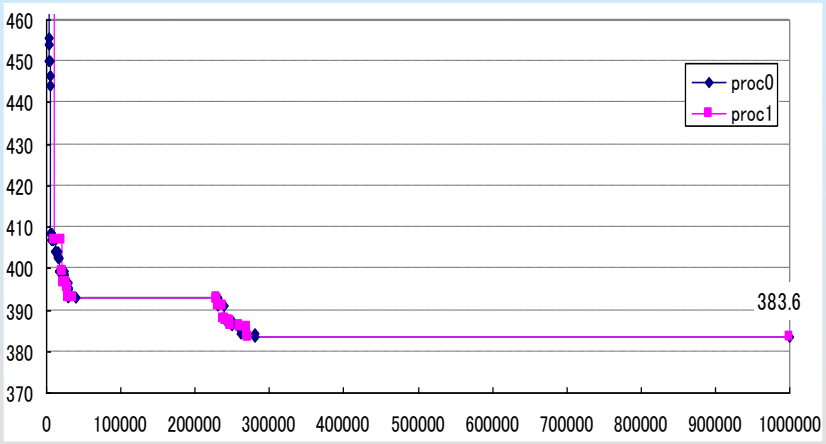


- An example of total computing time and total waiting time at each CPU in the practical experiment is as shown here.

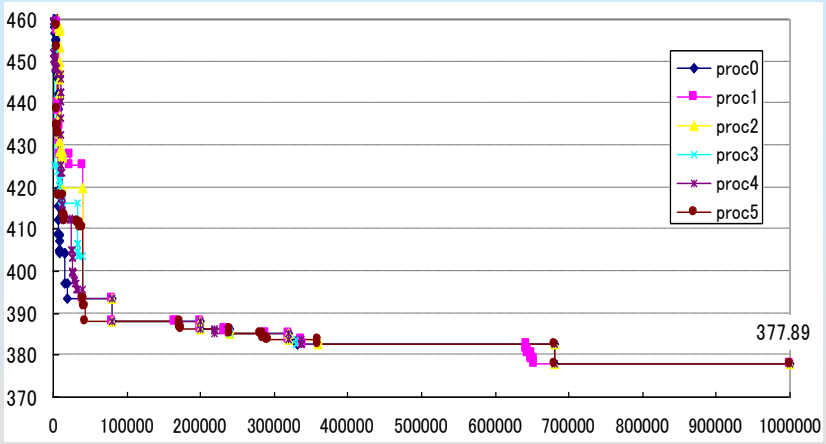


- The processes, Proc0 and Proc2, which is executed on the more rich CPU are waiting for the other processes about 12% of the total computing time.
- This useless computation time should be improved for the efficient computation.

Appendix: Other Experimental Result



2 CPUs



6 CPUs

- In the presentation, several transitions of the optimization have been shown when 4 CPUs are derived.
- Here, two more transitions of the optimization are shown when 2 and 6 CPUs are derived.
- When 2 CPUs are used, the optimization has been performed well, but the schedule to show less than 380 has not finally been provided.
- When 6 CPUs are used, the optimization has been performed very well. Then the best schedule has been provided.