Modeling for flexible task scheduling with multi-skilled workforce

Enxi Lin

University of Alberta

31 December 2022

This paper presents a mixed-integer programming model for flexible tasks scheduling problems with multi-skilled workforce constraints. Each task has a predetermined time window, predecessors, required skill, and which room and project it belongs to. Each employee has a skill set, working time window. The objective is to minimize the number of unassigned tasks, total working time and total employee-project variability. We implemented the programming in Python API for CPLEX using docplex.mp package. A numeric example is provided to check the performance of the model.

1.Introduction

Scheduling is a very important step in planning the workload for any organization. If doing it manually, it is very time-consuming and might not lead to the most optimal solution. Therefore, we use mixed integer programming to help us solve complex scheduling problems. By formulating the scheduling as a mixed integer programming problem with multiple constraints, we can obtain the best possible solution.

At present, MIP models have been applied to a wide variety of industrial-level problems. Since our scenario is set as a laboratory, the achievement in the medical operations and management related field deserves our attention. Seeing that, we have the same objective: improve capacity utilization, control costs and increase operational efficiency. For example, Cote et al. (2007) developed MIP for locating new treatment centers while minimizing the sum of patients' costs. Bard and Purnomo (2007) presented a model that combines cyclic as well as nurse preference scheduling by formulating an Integer Program. Lamiri et al. (2008) proposed a model for operating room planning with two types of demand for surgery: elective and emergency in order

to minimize the sum of the elective patient related costs and overtime costs of the operating rooms.

The rest of this report will be organized as the following: Section 2 discusses the problem description. Section 3 and 4 are the mathematical modeling and numerical experiments of the basic model. In section 5, a complex model will be proposed. Finally, section 6 presents the conclusion and future work.

2. Problem Description

The scenario is set as a preclinical and clinical laboratory. Our allocation plan is based on one day's task needs. The tasks include taking care of laboratory animals, dosing oral or parenteral, monitoring animal behavior, collecting biological specimens and so on. In this basic model, we assume each task needs at least half-hour. And we split one day into 48 half-hour working sections, which means we account for two sections even if the test only needs 50 minutes. Detail information about task and employee is as follows,

1. Each task has positive and negative allowable deviations from its predetermined time window. Deviations are measured in half an hour.
2. Each task has a project number that shows which project it belongs to. We have a decision variable "employee-project variability" that counts the number of projects each employee is involved in.
3. Each task has a room number indicating in which room it should be done. For safety reasons, we can only do a maximum of one task in each room at the same time. However,

we can always change the maximum number of concurrent tasks according to the demand.

4. Some tasks may have predecessors, which means the predecessors should be done first otherwise neither task can be completed. Tasks and their predecessors should belong to the same project.
5. Each task only needs one employee. And one employee can only perform one task at the same time.
6. Each employee has an available time window. Employee's actual working time equal to the ending time of the last task assigned to him minus the starting time of the first test.

The objective of modeling is to maximize the number of assigned tasks. If an employee is involved in too many projects, it will increase the chance of making mistakes. And we prefer a relatively compact working schedule for employees, so they can arrive at the company later or leave early. On this basis, we also want to reduce the employee-project variability and the employee's actual working time.

3.Mathematical Modeling

This section will describe the process of developing the model. The set and variables we used are listed as follows.

Set and Parameter	Description
E	a set of available employee
S	a set of skill
R	a set of room
P	a set of project

Ψ	a set of all working section (index by i), $\Psi = \{0, 0.5, 1,..., 23, 23.5\}$
ω	a set of all original task (index by k)
D_k	For original task k, a set of all tasks with different time windows that can occur within the allowable time deviation. (index by d)
T	A set of all tasks (index by t). T= $D_1 \cup D_2 \cup ... \cup D_k$
TR_r	a set of tasks should be done in room r
ts_t	The skill required by task t
tp_t	The project which the task t belongs to
ES_e	The set of all skills that employee e have
TT_t	The set of schedule working sections for task t ie, for a half hour task which should be done at 8am, TT={8}
ET_e	The set of available working sections for employee e Ie, for a employee are available from 8am to 4pm and half hour lunch time at 12pm, ET={8,8.5,..,11.5,12.5,13,...,15.5}

3.1 Decision variables

Variable	Description
$X_{e,t}$	Binary variable indicate whether task t is assigned to employee e
WE_e	Continuous variable equal to the ending time of last task assigned to employee e
WS_e	Continuous variable equal to 1/the starting time of first task assigned to employee e
EWT_e	Continuous variable indicate the total working time of employee e
$PN_{p,e}$	Continuous variable indicate the number of task assigned to employee which belongs to project p
$N_{p,e}$	Binary variable indicate whether employee e is assigned a task belongs to project p

3.2 Constraints

Constraint 1: When employees are unavailable, we can not assign any task starting at that time to him. In other words, if the set of scheduled time for a task is not a subset of the set of available working sections of the employee, we do not assign the task to the employee.

$$\text{For } \forall e, \forall t, \text{ if } TT_t \not\subset ET_e: \quad X_{e,t} < 1$$

Constraint 2: We require each employee to only perform one task at each time, which means we can not assign overlapping tasks.

$$\text{For } \forall t \text{ that } t1 \neq t2, \text{ if } TT_{t1} \cap TT_{t2} \neq \emptyset:$$

$$For\ \forall e,\ X_{e,t1} + X_{e,t2} \leq 1$$

Constraint 3: If the employee e does not have the skill required by task t, we can not assign the task to this employee.

$$\text{For } \forall e, \forall t, \text{ if } ts_t \notin ES_e: \quad X_{e,t} < 1$$

Constraint 4: One task only needs one employee.

$$\text{For } \forall t: \sum_{e \in E} X_{e,t} \leq 1$$

Constraint 5: For each task, it has positive and negative allowable deviations from its predetermined time window. Deviations are measured in half an hour. We just need to treat them as separate, independent tasks, and only select one among all its deviation tasks. For example, we have a half-hour task that needs to be done at 8 am and allow one hour of positive and negative deviation. Then, we need to select one task among five half-hour tasks which should be done at 7 am,7:30 am, 8 am, 8:30 am and 9 am.

$$\text{For } \forall k \in \omega: \sum_{d \in D_k} X_{e,d} \leq 1$$

Constraint 6: Each room only allows one concurrent task.

$$\text{For } \forall i \in \Psi, \forall r \in R:$$

$$\text{For } t \in TR_r \,\&\, i \in TT_t: \sum_{e \in E} \sum_{t} X_{e,t} \leq 1$$

Constraint 7: In this section, we have to ensure that the predecessor is completed prior to the start of the given task.

t: the task it self ; t′: predecessors of t

This is accomplished by: (i) The total number of assigned tasks t should always be less or equal than the total number of assigned predecessors t′.

$$\sum_{e \in E} x_{e,t} \leq \sum_{e \in E} x_{e,t'}$$

(ii) If the start time of a given task is less than the end time of the predecessor, the total number of assigned tasks and its predecessor should be less and equal than one. Combine with (i), the result would be only assign the predecessor or neither task is assigned.

$$\text{If } \exists t_1 \in TT_t \; and \; \exists t_2 \in TT_{t'}, \text{ such that } t_2 \geq t_1:$$

$$\sum_{e \in E} x_{e,t} + \sum_{e \in E} x_{e,t'} \leq 1$$

3.3 Objective and Aim function

Total number of assigned tasks: $\sum_{e \in E} \sum_{t \in T} x_{e,t}$

Total working time: $\sum_{e \in E} ewt_e$

Total employee-project variability: $\sum_{e \in E} \sum_{p \in P} n_{p,e}$

Our final aim function is now expressed as,

$$\text{Maximum} \quad (A * \sum_{e \in E} \sum_{t \in T} x_{e,t} - B * \sum_{e \in E} ewt_e - C * \sum_{e \in E} \sum_{p \in P} n_{p,e}),$$

where A, B and C should be positive values.

3.4 Analysis of the results

3.4.1 Employee working time

In order to find the earliest starting time of assigned task for each employee more easily during programming, we set the equation for variable WS_e = max ($x_{e,t} / minTT_t, \forall t \in T$). It will return the value 1/(starting time of earliest assigned task) for that employee. And total working time variable $EWT_e = WS_e + WE_e$, which is not exactly equal to the end time of the latest assigned task minus the starting time of the earliest assigned task. However the aim function still will push

the model to choose the task with smallest WS_e value. After solving the model, we can calculate the exact and correct total employee working time according to the solution of the model.

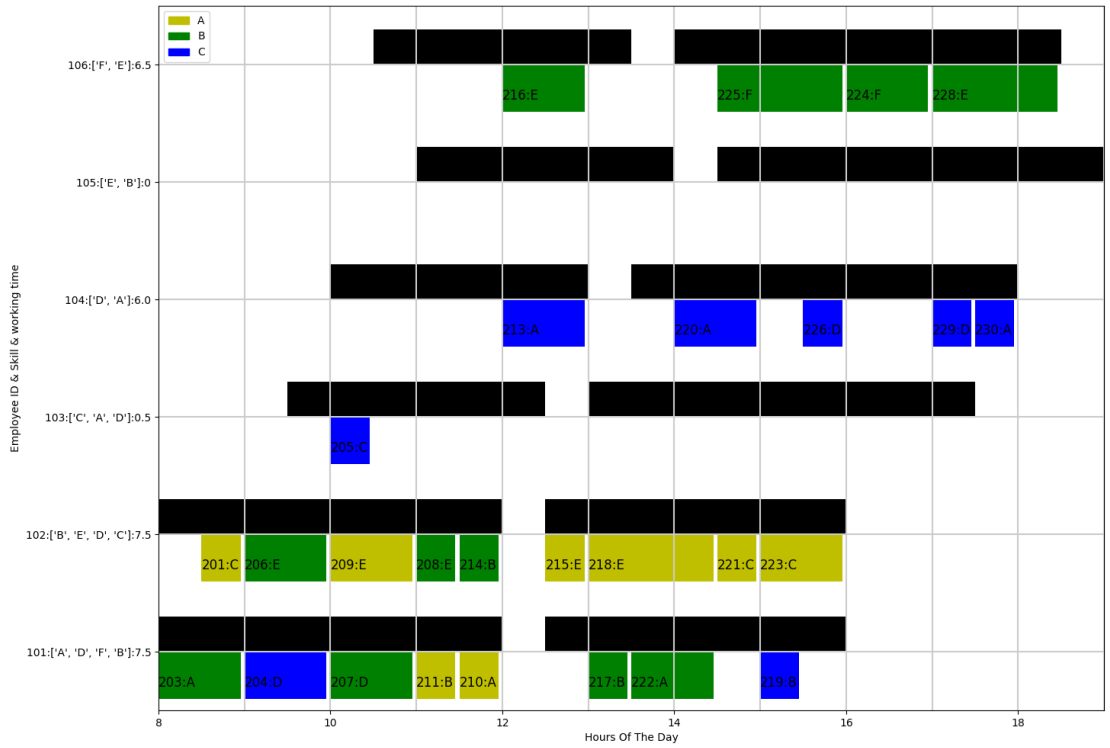
3.4.2 Classify unassigned task

After solving the model, we will obtain a list of unassigned tasks. It's necessary to figure out the reason why it can not be assigned. Then, we can make adjustments accordingly to improve the production efficiency. We classify the unassigned tasks into four categories: Predecessors, Room,Time and Skills. 'Predecessors' means the task was not assigned since the predecessor was not assigned or completed. 'Room' means the appointed room for the task has already been occupied. 'Time' means the task can be assigned if some employee with required skill can work overtime. 'Skills' means we have to train more employees to master the skill for the task.

4. Application to An Example.

Here, we computed an example of 30 tasks with 30 minutes of positive and negative allowable deviation, 6 employees, 3 projects, 6 skills and 3 rooms. The details of data are shown in the appendix. The value of parameters A, B and C in the aim function should depend on the data. To make sure that the change in 'Total working time' and 'Total employee-project variability' won't influence the 'Total number of assigned tasks', we should assign parameter A a relatively large value. In this case, we let A=10, B=0.1 and C=0.1. Picture1 shows the results of the example. The black bar represents employee availability. The color bar represents which project it belongs

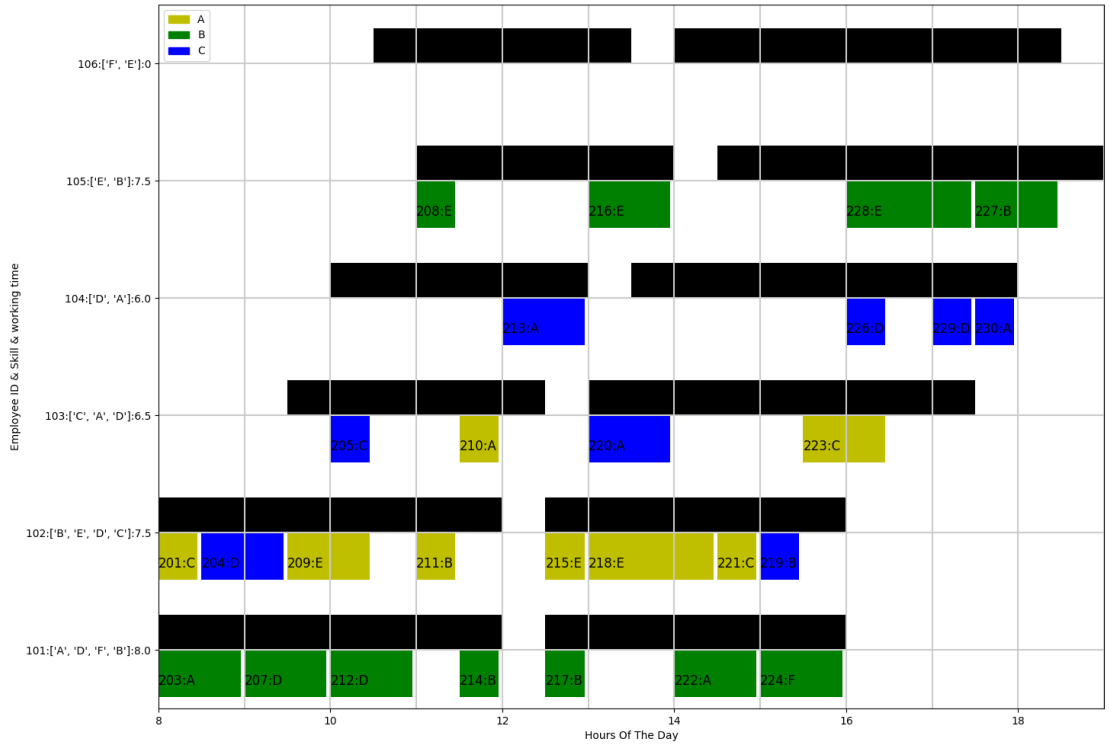
to, and is labeled by its task id and required skill. Labels for the y-axis are employee ID, skills and actual working time.



Picture 1: Result of numerical example

We have three unassigned tasks 202, 212 and 227. Task 202 needs time 18:30-19:00 and skill D. It wasn't assigned because there is no available employee. It can be completed if employee 104 works one extra hour. Task 212 needs time 10:30-11:30 and room B. It wasn't assigned because room B was occupied by tasks 207, 208 and 214 from 10:00 to 12:00. Task 227 needs time 17:00-18:00 and room B. It wasn't assigned because room B was occupied by task 228.

We can also change the weight of three objectives in the aim function, the result could be different. For example, if we set B, the weight for ‘total working time’, equal to zero. The result will be as shown in Picture 2. The total actual working hours increased from 28 to 35.5. However, the total employee-project variability decreased from 8 to 7.



Picture 2: result when we delete ‘Total working time’ from objective function

5. Complex Model

In the basic model, We split one day into 48 half-hour working sections. Tasks and its deviation are measured in working sections. The basic model has three main defects:

1. Under the assumption, some spare time would be wasted after some tasks are finished. A fifty minutes' task can waste about ten minutes.
2. The tasks schedule is not flexible enough. Since its scheduled time and deviation can only be adjusted by working sections.
3. We treat each half-hour deviation as a new and independent task. As a result, the calculation amount of the model will increase very fast when we increase the length of allowable deviations.

In order to remedy those defects, we came up with a new model which set the actual starting time of the task as a decision variable. In this case, the actual starting time of the task we obtain from the optimization model can be accurate to minute. The information we need for tasks and employees does not change. However, we need to change the form of input data. For example:

Task	…	Start Time	End Time	Duration (minutes)	…	
201	…	7:30		30	…	Task 201 need 30 minutes to complete and should start after 7:30
202	…		17:30	30	…	Task 202 need 30 minutes to compete and should be completed before 17:30
203	…	8:00	12:00	60	…	Task 203 need 60 minutes and should start after 8:00 and be completed before 12:00

And we divide the employee's availability into two sections: before and after the lunch.

In this optimization model there are the following sets and variables included.

Set	Description
E	Set of available employee
T	Set of all tasks

D	Set of sections: before the lunch(1), after the lunch(2)
$\Psi_{d,e}$	Set of employee e's availability on section d. Ie, $\Psi_{1,101}$is the availability of employee 101 before lunch time.
tb_t	Scheduled task start time from the input data
te_t	Scheduled task end time from the input data
td_t	Scheduled task duration from the input data
ts_t	Required skill by task t
ES_e	The set of all skills that employee e have
TR_r	a set of tasks should be done in room r

5.1 Decision variables

Variables	Description
$x_{e,t,s}$	A binary variable indicates whether task t was assigned to section s of employee e
y_t	An integer variable shows the actual starting time of task t in minutes. For example, 780 means this task was assigned to start at 1 pm
WE_e	Integer variables equal to the ending time of last task assigned to employee e
WS_e	Integer variable equal to 1/the starting time of first task assigned to employee e
EWT_e	Continuous variable indicate the total working time of employee e
$PN_{p,e}$	Continuous variable indicate the number of task assigned to employee which belongs to project p
$N_{p,e}$	Binary variable indicate whether employee e is assigned a task belongs to project p

5.2 Constraints

Constraint 1: The actual starting time of a task should be larger or equal to its scheduled starting time. And the actual ending time should be smaller or equal to its scheduled ending time.

$$\text{For } \forall t: y_t \geq tb_t$$

$$y_t \leq te_t - td_t$$

Constraint 2: If the employee e does not have the skill required by task t, we can not assign the task to this employee.

$$\text{For } \forall e, \forall t: \text{if } ts_t \notin ES_e : \sum_{d \in D} x_{e,t,d} < 1$$

Constraint 3: One task only needs one employee.

$$\text{For } \forall t \in T: \sum_{d \in D} \sum_{e \in E} x_{e,t,d} \leq 1$$

Constrint 4: When employees are unavailable, we can not assign any task starting at that time to him. Let M be a sufficiently large coefficient.

$$\text{For } \forall d \in D, \forall e \in E, \forall t \in T:$$

$$y_t \geq \min\Psi_{d,e} - (1 - x_{e,t,d}) * M$$

$$y_t + td_t \leq \max\Psi_{d,e} + (1 - x_{e,t,d}) * M$$

Constrint 5: We have to ensure that the predecessor is completed prior to the start of the given task.

t: the task it self ; t′: predecessors of t

For ∀t ∈ T: $y_t \geq y_{t'} + td_{t'}$ - (1- $\sum_{d\in D}\sum_{e\in E} x_{e,t,d}$) * M

Constraint 6: employees can only perform one task at each time. In other words, tasks can not be assigned to the same employee if they are overlapped. This constraint use a binary variables $p_{i,j}$. $p_{t1,t2}$ =1 indicates task t1 completely precedes task t2. We only want to implies the non-overlapping constraint if and only if two tasks are assigned to the same employee ($\sum_{d\in D} x_{e,t1,d} = \sum_{d\in D} x_{e,t2,d} = 1$).

For ∀e ∈ E, ∀t ∈ T that $t1 \neq t2$:

$$y_{t1} + td_{t1} \leq y_{t2} + \text{M} * (1 - p_{t1,t2}) + \text{M} * (2 - \sum_{d\in D} x_{e,t1,d} - \sum_{d\in D} x_{e,t2,d})$$

$$y_{t2} + td_{t2} \leq y_{t1} + \text{M}* p_{t1,t2} + \text{M} * (2 - \sum_{d\in D} x_{e,t1,d} - \sum_{d\in D} x_{e,t2,d})$$

These two formulas of constraint 6 obtained from the study of Aronsson et al (2011).

Constraint 7: There is no more than one task being performed in the room at a given time. In other words, two tasks requiring the same room can not both be assigned. This constraint is very similar to the last constraint and we only need to make subtle changes.

For ∀t ∈ T that $t1 \neq t2$:

$$y_{t1} + td_{t1} + \text{M} * p_{t1,t2} + \text{M} * \sum_{e\in E}\sum_{d\in D} x_{e,t1,d} + \text{M} * \sum_{e\in E}\sum_{d\in D} x_{e,t2,d} \leq y_{t2} + 3\text{M}$$

$$y_{t2} + td_{t2} + \text{M} * \sum_{e\in E}\sum_{d\in D} x_{e,t1,d} + \text{M} * \sum_{e\in E}\sum_{d\in D} x_{e,t2,d} \leq y_{t1} + \text{M}* p_{t1,t2} + 2\text{M}$$

Our aim function now expressed as

$$\text{Maximum} \quad (\text{A}* \sum_{d\in D}\sum_{e\in E}\sum_{t\in T} x_{e,t,d} - B * \sum_{e\in E} ewt_e - C * \sum_{e\in E}\sum_{p\in P} n_{p,e})$$

6. Conclusion and Future Work

This report studied flexible task scheduling with multi-skills workforce problems. We presented two mixed integer programming models: the basic model assigned the task to the employee and the fixed working section, and the complex model was setting the actual starting time of the task as a decision variable directly. In addition, we did a numerical experiment on the basic model and the corresponding result is correct and reasonable. However, the real data is very large,i.e. 400 employees, 100 skills and 700 tasks each day. As a result, we will figure out the coding for the complex model in future work, which has fewer constraints and more rapid computation.

We also have some inspiration and ideas that may be implied in future work.

1. We can assign priority to those tasks which must be done today.
2. Some tasks required a certain employee to complete them.
3. In our models, one task only needs one employee. If we change this constraint to more than one employee, we can consider the employee-employee association and employee incompatibilities.
4. We can rank the skills and give them an ‘inclusion relation’. For example, we have skills A, B and C. Employees who have skill C can also complete the tasks required for skill A and B.

Reference

Aronsson, M., Bohlin, M., & Kreuger, P. (2011, March 6). Mixed integer-linear formulations of cumulative scheduling constraints - A comparative study. *CORE*. https://core.ac.uk/reader/300994090. Accessed 31 December 2022.

Bard, J. F., & Purnomo, H. W. (2007). Cyclic preference scheduling of nurses using a lagrangian-based heuristic. *Journal of Scheduling*, *10*(1), 5–23. doi:10.1007/s10951-006-0323-7.

Côté, M. J., Syam, S. S., Vogel, W. B., & Cowper, D. C. (2007). A mixed integer programming model to locate traumatic brain injury treatment units in the Department of Veterans Affairs: A case study. *Health Care Management Science*, *10*(3), 253–267. doi:10.1007/s10729-007-9018-7.

Lamiri, M., Xie, X., Dolgui, A., & Grimaud, F. (2008). A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal of Operational Research*, *185*(3), 1026–1037. doi:10.1016/j.ejor.2006.02.057.

Appendix

Task	Skill	Start Time	End Time	Room	Predecessor	Project
201	C	8:00	8:30	C		C
202	D	18:30	17:00	C		C
203	A	8:00	9:00	B		B
204	D	8:30	9:15	A		A
205	C	9:30	10:00	A	204	A
206	E	8:45	9:30	B		B
207	D	9:30	10:25	B	206	B
208	E	10:30	10:45	B	207	B
209	E	10:00	10:45	C		C
210	A	11:00	11:30	C	209	C
211	B	11:00	11:30	C		C
212	D	10:30	11:30	B		B
213	A	11:30	12:15	A		A
214	B	12:00	12:30	B		B
215	E	12:00	12:30	C		C
216	E	12:30	13:15	B		B
217	B	13:00	13:30	B		B
218	E	12:30	14:00	C		C
219	B	14:30	15:00	A	220	A
220	A	13:30	14:15	A		A
221	C	14:00	14:30	C		C
222	A	14:00	15:00	B		B

223	C	15:30	16:15	C		C
224	F	15:30	16:30	B		B
225	F	15:00	16:30	B		B
226	D	16:00	16:30	A		A
227	B	17:00	18:00	B		B
228	E	16:30	18:00	B		B
229	D	17:30	18:00	A		A
230	A	18:00	18:30	A		A

Employee ID	Available Time	Lunch Time	Skills
101	8:00-16:00	12:00-12:30	A,B,D,F
102	8:00-16:00	12:00-12:30	B,C,D,E
103	9:30-17:30	12:30-13:00	A,C,D
104	10:00-18:00	13:00-13:30	A,D
105	11:00-19:00	14:00-14:30	B,E
106	10:30-18:30	13:30-14:00	E,F