NHÓM
8

Sắp xếp lịch làm cho nhân viên bằng Gurobi

Nhóm 8

Thành viên bao gồm

Dương Đức Vương

22000133

K67 Toán tin

Nguyễn Đình Đương 18001119

K63 MT&KHTT

Trần Văn Lâm

22001268

K67 KHDL

Nội dung

Ứng dụng của tối ưu hóa vào sắp sếp lịch?

- Giới thiệu tổng quan
- Mô hình hóa bài toán sắp xếp lịch làm cho nhân viên
- Giải quyết bài toán bằng Gurobi

I. Giới thiệu

Sắp xếp lịch làm. Vai trò của công cụ sếp lịch trong kế hoạch về nhân sự.

Sắp xếp lịch làm

Sắp xếp nhân viên vào đúng ca dựa trên thực lực và một số yếu tố khác

Các ràng buộc số nhân viên mỗi ca, thời gian làm việc của từng nhân viên,...

COMPLEVO

BUSINESS OPTIMIZATION

FREIZEIT
SOFTWARE FÜR BESSERE PERSONALPLANUNG

Gesamtplan Mitarbeiter

Zeitraum: 15.12.2014 bis 21.12.2
Erstellt am: 26.02.2015 15:04:34
Erstellt von: Admin
Unterschrift:

Hauptstandort	Name	Soll [h]	Ist [h]	Pause [h]	Ausl. [%]	KW 51, 15.12.-21.12.14 Mo (15)	Di (16)	Mi (17)	Do (18)	Fr (19)	Sa (20)
					89,5%						
Charlottenburg	Baier, Dietlinde	26,09	24,75		94,9%	06:15-12:00 Ch Verkaufen (S) (VS) 14:30-18:15 Ch Verkaufen (S) (VS)	06:15-12:00 Ch Verkaufen (S) (VS) 14:30-18:15 Ch Verkaufen (S) (VS)	14:30-18:15 Ch Verkaufen (S) (VS)			08:00-10:00 Ch Verkaufen (L) (VL)
Charlottenburg	Koch, Johanna										
Friedrichshain	Fuhrmann, Brigitte	40,00	19,50	0,50	48,8%	15:00-18:15 F Verkaufen (L) (VL)	15:00-18:15 F Verkaufen (L) (VL)	05:45-13:00 F Verkaufen (S) (VS)		15:00-18:15 F Verkaufen (L) (VL)	07:30-10:30 F Verkaufen (L) (VL)
Mitte	Herzberg, Silke	40,00	9,50	0,25	23,8%		05:30-12:00 M Verkaufen (L) (VL)		15:00-18:15 F Verkaufen (L) (VL)		
Mitte	Junker, Mandy	17,39	30,50		175,4%			06:15-12:00 Ch Verkaufen (S) (VS)	06:15-12:00 Ch Verkaufen (S) (VS) 14:30-18:15 Ch Verkaufen (S) (VS)	06:15-12:00 Ch Verkaufen (S) (VS) 14:30-18:15 Ch Verkaufen (S) (VS)	06:30-12:15 Ch Verkaufen (S) (VS)
Mitte	König, Torsten	40,00	24,00	2,00	60,0%	05:30-12:00 M Verkaufen (S) (VS)	05:30-12:00 M Verkaufen (S) (VS)	05:30-12:00 M Verkaufen (S) (VS)	05:30-12:00 M Verkaufen (S) (VS)		
Mitte	Moench, Kerstin	40,00	38,75	1,75	96,9%	05:30-12:00 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (L) (VL)	05:30-12:00 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (L) (VL)	05:30-13:30 M Verkaufen (L) (VL)
Mitte	Pfeiffer, Anna	17,39	11,00	0,50	63,3%			08:00-13:00 P Verkaufen (S) (VS)		05:30-12:00 M Verkaufen (L) (VL)	
Mitte	Reiniger, Mathilde	17,39	39,50		227,1%	05:45-13:00 F Verkaufen (S) (VS)	05:45-13:00 F Verkaufen (S) (VS)	15:00-18:15 F Verkaufen (L) (VL)	05:45-13:00 F Verkaufen (S) (VS)	05:45-13:00 F Verkaufen (S) (VS)	05:45-13:00 F Verkaufen (S) (VS)
Mitte	Winkel, Lisa	26,09	39,50		151,4%	12:00-18:30 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (S) (VS)	05:30-12:00 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (S) (VS)	05:30-12:00 M Verkaufen (S) (VS)	06:30-13:30 M Verkaufen (L) (VL)
Mitte	Zimmermann, Leon	40,00	31,50	2,50	78,8%	12:00-18:30 M Verkaufen (S) (VS)		12:00-18:30 M Verkaufen (S) (VS)	12:00-18:30 M Verkaufen (L) (VL)	12:00-18:30 M Verkaufen (S) (VS)	05:30-13:30 M Verkaufen (S) (VS)
						08:00-12:30 P Verkaufen (S) (VS)	08:00-12:30 P Verkaufen (S) (VS)		08:00-12:30 P Verkaufen (S) (VS)	08:00-12:30 P Verkaufen (S) (VS)	08:00-13:00 P Verkaufen (S) (VS)

COMPLEVO sử dụng Gurobi trong việc lập kế hoạch nhân sự của mình và kết quả là làm tăng sự hài lòng và năng suất của nhân viên.

Thử thách

Để tối ưu hóa phạm vi thay đổi nhân viên và quản lý nhân sự cho các doanh nghiệp cỡ vừa.

Kết quả

Với Gurobi, Complevo đã giảm được chi phí nhân viên và thời gian làm thêm cũng như tăng cường sự hài lòng trong tuyển dụng và nhân viên.

Bestaff

Tanca

Office

NỀN TẢNG QUẢN TRỊ **TỔNG THỂ** DOANH NGHIỆP

Tại sao cần phải có công cụ để sắp xếp lịch làm

Hiệu quả công việc

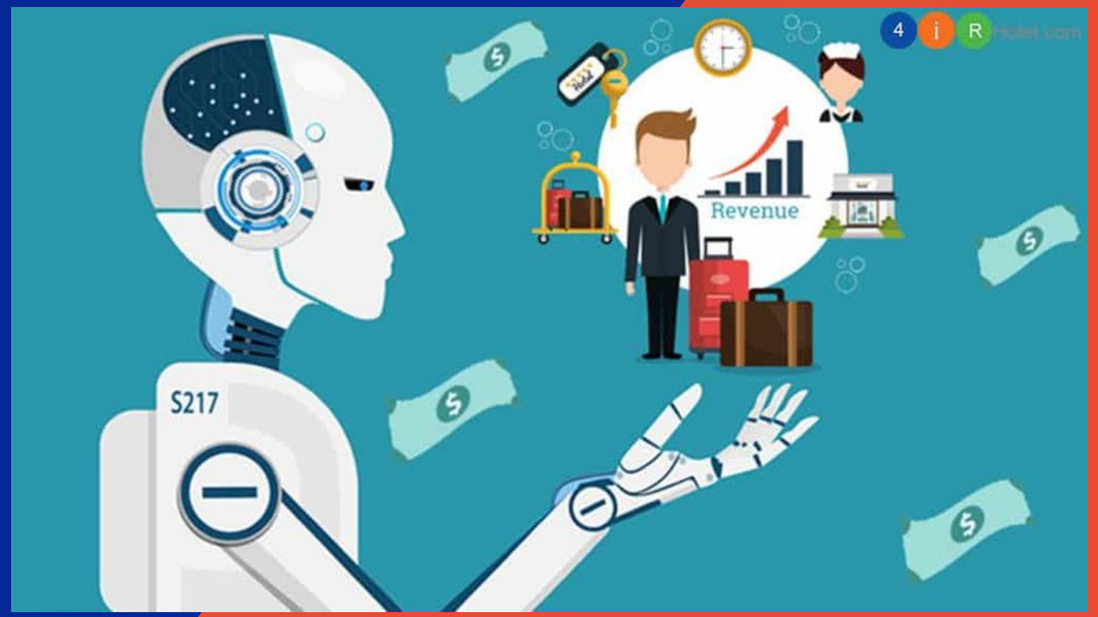
Đảm bảo chất lượng dịch vụ

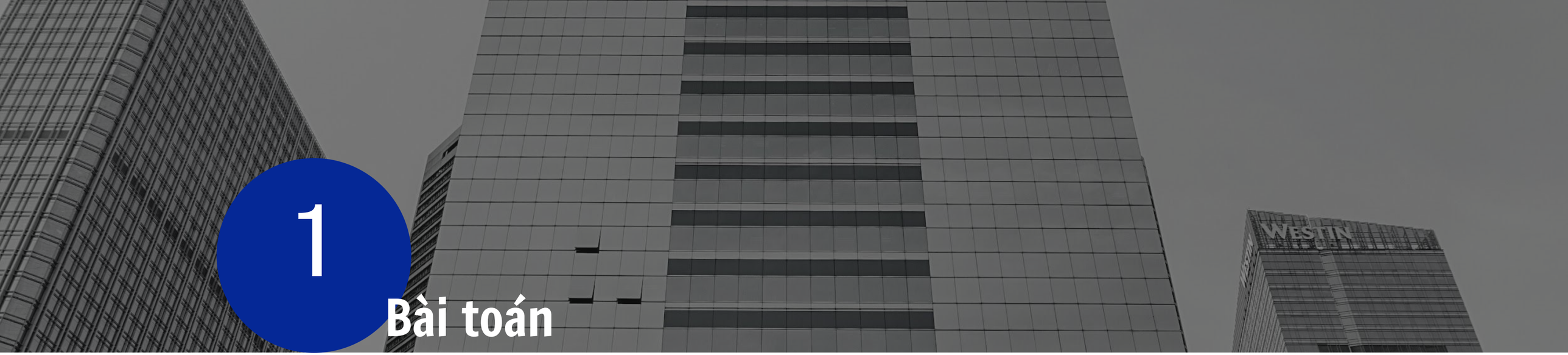
Tối ưu hóa lịch làm việc

Tiết kiệm thời gian và nguồn lực

II. Mô hình hóa bài toán

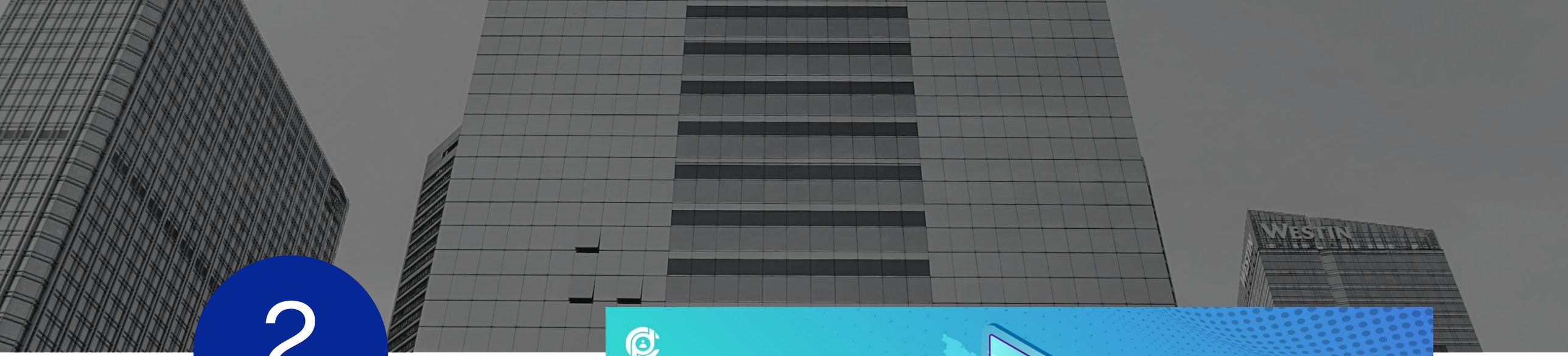
Sắp xếp lịch tuần cho nhân viên phù hợp dựa trên thực lực của từng nhân viên trong từng ca, đáp ứng đủ yêu cầu về nhân lực và một số ràng buộc khác





1 Bài toán

- Open: 6h00 – 18h00
- Đầu vào:
    + Thời gian rảnh của từng nhân viên trong 1 tuần
    + Xếp hạng độ thạo việc của từng nhân viên
- Đầu ra: lịch làm việc cho tất cả nhân viên trong 1 tuần
- Yêu cầu:
    + Nhân viên có xếp hạng cao trong độ thạo việc thì ưu tiên được làm nhiều hơn
    + Số lượng ca làm tối đa trong một ngày của từng nhân viên là 2 ca
    + Số lượng ca sáng, trưa + chiều, tối lần lượt là 3, 9,



2

Mô hình hóa

- Đầu vào là thời gian nhân viên đăng ký, ta có thể đưa nó về từng ca.
- Đánh số các ca sáng, trưa, chiều, tối lần lượt là 1, 2, 3, 4.

Các thông số đầu vào

Giả sử có n nhân viên, có 4 ca mỗi ngày, thứ 2 đến thứ 6 hàng tuần

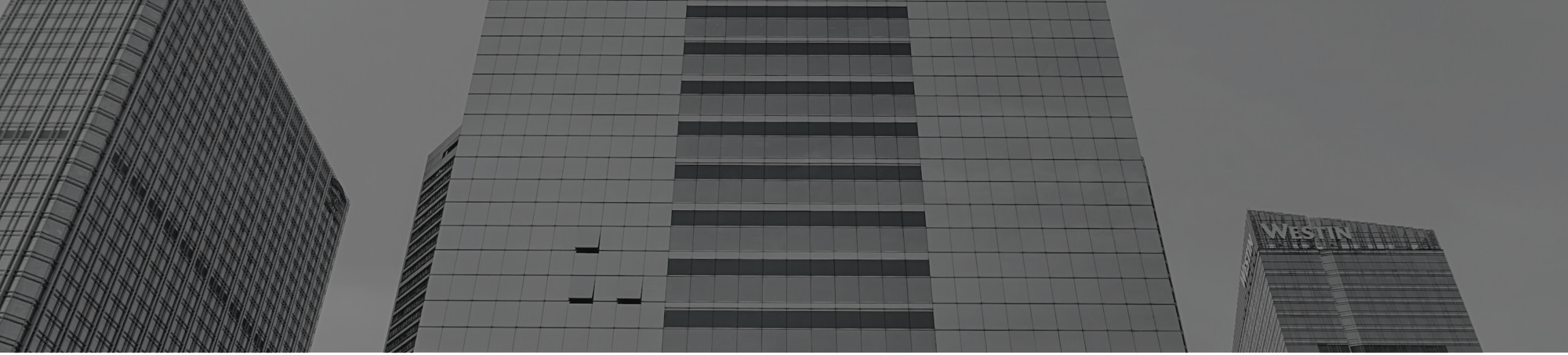
$t_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ *có thể làm được ca* $\boldsymbol{h}$ *trong thứ* $\boldsymbol{d}$ *hay không*

$r_{i,h}$: *xếp hạng độ thành thạo công việc của nhân viên* $\boldsymbol{i}$ *trong ca* $\boldsymbol{h}$

$$i = \{1, \dots, n\}$$

$$d = \{2, \dots, 6\}$$

$$h = \{1, \dots, 4\}$$



Đặt Biến

Quyết định lịch làm cho nhân viên

$s_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ *được xếp vào ca* $\boldsymbol{h}$ *thứ* $\boldsymbol{d}$ *hay không*

$o_{i,d}$: *nhân viên* $\boldsymbol{i}$ *có nghỉ vào thứ* $\boldsymbol{d}$ *hay không*

$i = \{1, \dots, n\}$

$d = \{2, \dots, 6\}$

$h = \{1, \dots, 4\}$

Ràng buộc

Ràng buộc số lượng nhân viên trong mỗi ca

- **Ca sáng cần 3 người.**
- **Ca trưa + chiều cần 9 người.**
- **Ca tối cần 2 người.**

$$\sum_{i=1}^{n} s_{i,d,1} = 3$$

$$\sum_{i=1}^{n} s_{i,d,2} + \sum_{i=1}^{n} s_{i,d,3} = 9$$

$$\sum_{i=1}^{n} s_{i,d,4} = 2$$

$$d = \{2, \dots, 6\}$$

Ràng buộc

Ràng buộc thời gian làm việc của nhân viên

- Mỗi nhân viên làm không quá 2 ca một ngày

$$\sum_{h=1}^{4} s_{i,d,h} \leq 2(1 - o_{i,d}) \quad i = \{1, \dots, n\}, \; d = \{2, \dots, 6\}$$

- Đảm bảo nhân viên i nghỉ trong thứ d thì không có lịch làm trong thứ d

$$s_{i,d,h} + o_{i,d} \leq 1 \quad i = \{1, \dots, n\}, \; d = \{2, \dots, 6\}, \; h = \{1, \dots, 4\}$$

- Đảm bảo lịch làm của nhân viên i không vượt quá thời gian mà nhân viên đó đăng ký

$$s_{i,d,h} \leq t_{i,d,h}(1 - o_{i,d}) \quad i = \{1, \dots, n\}, \; d = \{2, \dots, 6\}, \; h = \{1, \dots, 4\}$$

$t_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ *có thể làm* được *ca* $\boldsymbol{h}$ *trong thứ* $\boldsymbol{d}$ *hay không*

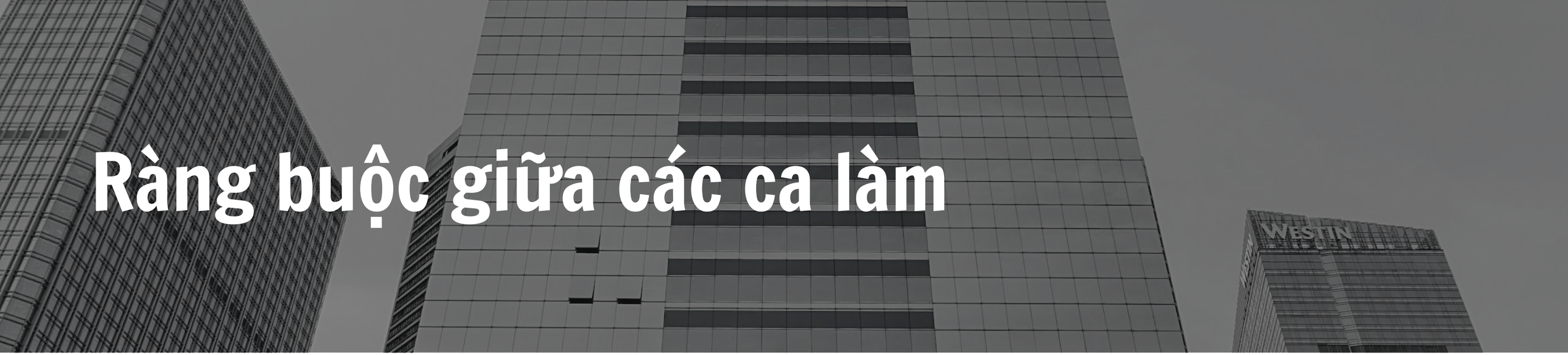
NHÂN VIÊN A	THỨ 2	THỨ 3	THỨ 4	THỨ 5	THỨ 6
SÁNG	1	1	1	0	1
TRƯA	1	1	1	1	1
CHIỀU	1	0	1	1	1
TỐI	0	0	0	0	0

t

NHÂN VIÊN A	THỨ 2	THỨ 3	THỨ 4	THỨ 5	THỨ 6
SÁNG	1	1	1	0	1
TRƯA	1	0	1	1	1
CHIỀU	0	0	0	1	0
TỐI	0	0	0	0	0

s

$s_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ được *xếp vào ca* $\boldsymbol{h}$ *thứ* $\boldsymbol{d}$ *hay không*



Ràng buộc giữa các ca làm

Không làm hai ca không liền nhau

$$s_{i,d,1} + s_{i,d,3} \leq 1 - o_{i,d}$$

$$s_{i,d,1} + s_{i,d,4} \leq 1 - o_{i,d}$$

Không làm hai có khoảng thời gian lồng nhau

$$s_{i,d,2} + s_{i,d,3} \leq 1 - o_{i,d}$$

$$i = \{1, \dots, n\}$$

$$d = \{2, \dots, 6\}$$

Hàm mục tiêu

min

$$\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,1}s_{i,d,1} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,2}s_{i,d,2} + 2\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,3}s_{i,d,3} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,4}s_{i,d,4}$$

Hạn chế số lượng nhân viên ca chiều

Ưu tiên các nhân viên thành thạo công việc

min $$\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,1}s_{i,d,1} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,2}s_{i,d,2} + 2\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,3}s_{i,d,3} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,4}s_{i,d,4}$$

s.t.

$$\sum_{i=1}^{n} s_{i,d,1} = 3$$

$$\sum_{i=1}^{n} s_{i,d,2} + \sum_{i=1}^{n} s_{i,d,3} = 9$$

$$\sum_{i=1}^{n} s_{i,d,4} = 2$$

$$\sum_{h=1}^{4} s_{i,d,h} \leq 2(1 - o_{i,d})$$

$$s_{i,d,h} + o_{i,d} \leq 1$$

$$s_{i,d,h} \leq t_{i,d,h}(1 - o_{i,d})$$

$$s_{i,d,2} + s_{i,d,3} \leq 1 - o_{i,d}$$

$$s_{i,d,1} + s_{i,d,3} \leq 1 - o_{i,d}$$

$$s_{i,d,1} + s_{i,d,4} \leq 1 - o_{i,d}$$

$i = \{1, \dots, n\}$

$d = \{2, \dots, 6\}$

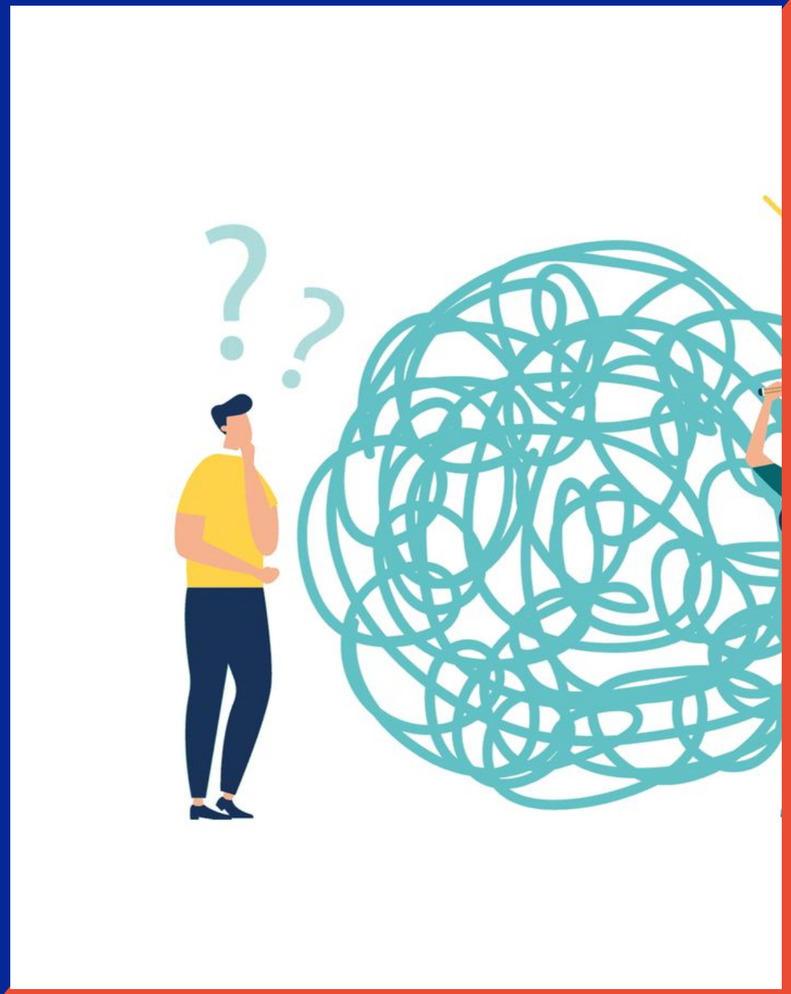
$h = \{1, \dots, 4\}$

$s_{i,d,h} \in [0,1],\ s_{i,d,h} \in \mathbb{N}$

$o_{i,d} \in [0,1],\ o_{i,d} \in \mathbb{N}$

III. Giải quyết bài toán bằng Gurobi

Có thể in ra lịch tối ưu cho toàn bộ nhân viên. Nếu không có lịch thoả mãn thì có thể cho biết cần bao nhiêu nhân viên trong ca nào không?



• Đầu vào

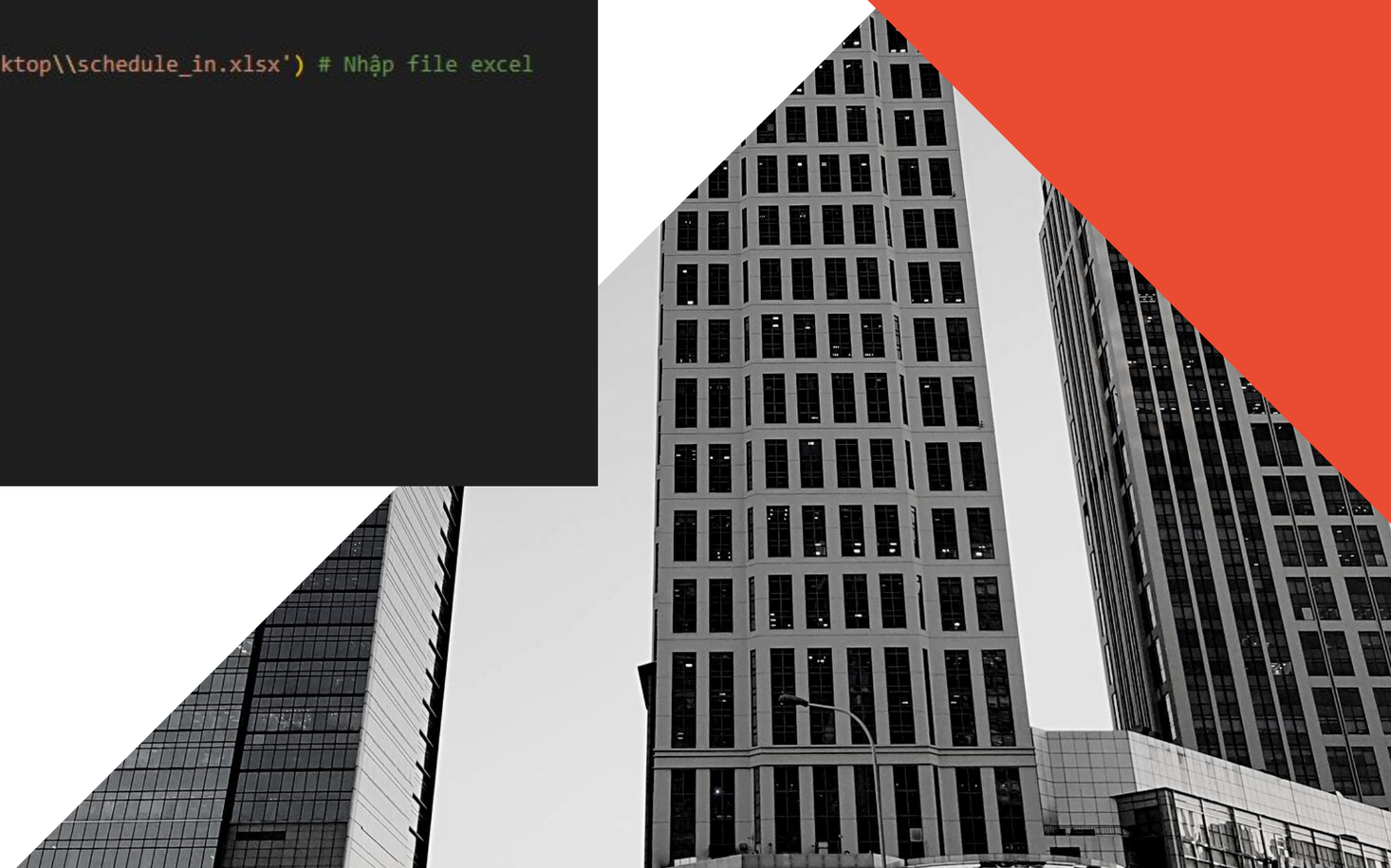
MON_IN	MON_OUT	TUE_IN	TUE_OUT	WED_IN	WED_OUT	THU_IN	THU_OUT	FRI_IN	FRI_OUT	RANK_MORNING	RANK_NOON	RANK_AFTERNOON	RANK_EVENING
6	14	6	14	6	14	6	14	6	14	2	3	3	1
10	18	10	18	10	18	10	18	10	18	2	2	2	3
12	16	6	15	6	14	0	0	10	14	3	3	3	1
0	0	0	0	9	13	0	0	0	0	1	3	1	1
12	18	0	0	10	18	12	18	0	0	2	3	1	3
9	13	0	0	9	13	6	14	0	0	3	3	1	3
6	16	0	0	12	16	12	16	9	13	1	2	1	2
12	16	9	13	12	16	12	16	9	13	3	1	2	2
0	0	6	10	0	0	6	11	6	10	3	1	2	1
15	18	0	0	9	18	9	13	10	18	3	1	1	3
10	18	10	16	0	0	0	0	10	18	1	3	1	3
16	18	12	18	16	18	12	18	16	18	3	3	2	2
6	18	6	15	10	14	0	0	10	15	3	2	1	2
0	0	10	14	9	13	9	13	9	13	3	2	1	2
9	13	0	0	11	16	0	0	9	13	1	3	2	3
10	15	10	15	0	0	10	15	0	0	2	2	1	2
6	18	6	18	6	18	6	18	6	18	2	2	3	2

Nhập các thông tin đầu vào: lịch nhân viên đăng ký

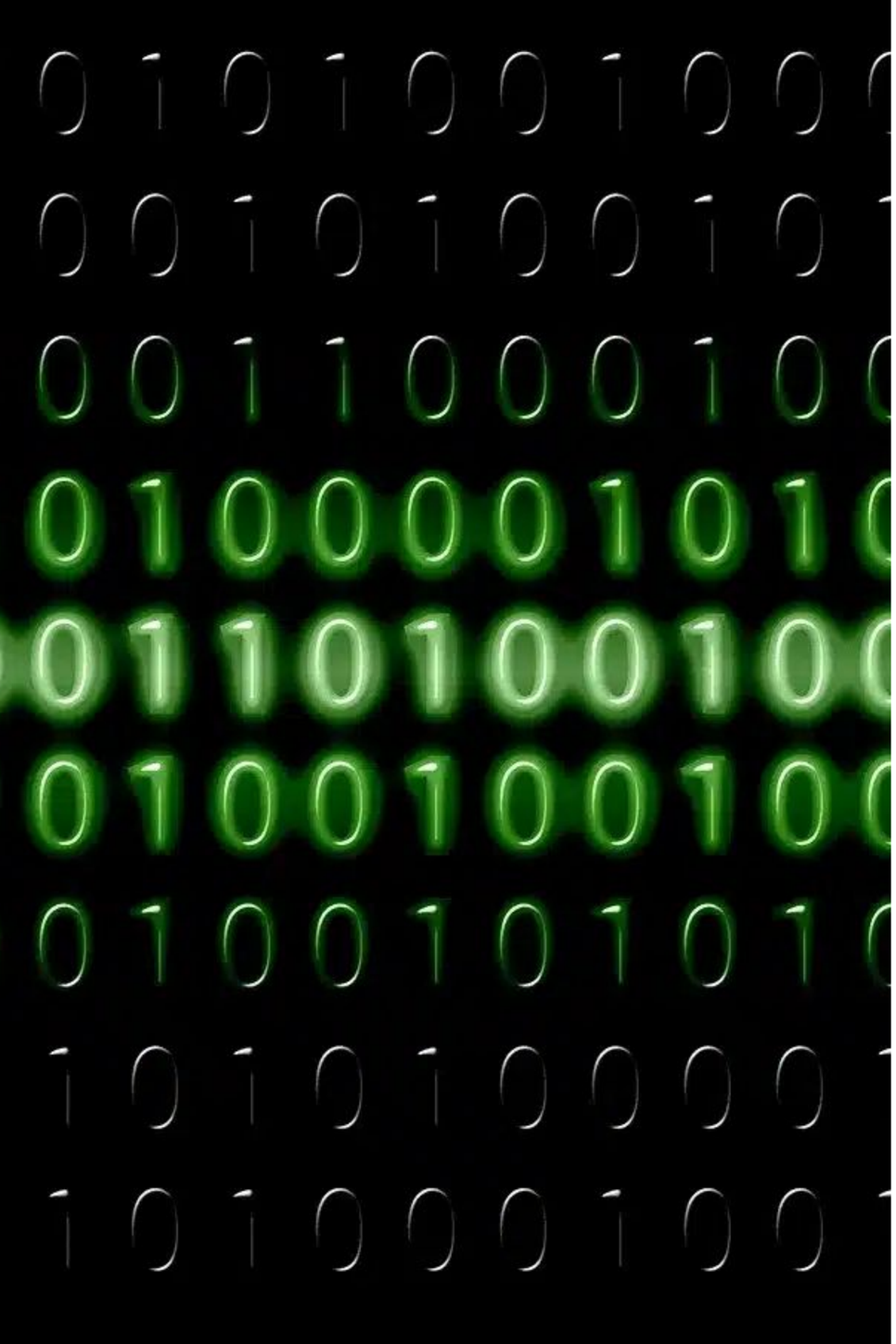
```
# Các thông tin đầu vào

free_times = pd.read_excel('C:\\Users\\Administrator\\OneDrive\\Desktop\\schedule_in.xlsx') # Nhập file excel

rank1 = free_times["RANK_MORNING"]
rank2 = free_times["RANK_NOON"]
rank3 = free_times["RANK_AFTERNOON"]
rank4 = free_times["RANK_EVENING"]
part_times = {
    "morning",
    "noon",
    "afternoon",
    "evening"
}
days = ("MON", "TUE", "WED", "THU", "FRI")
```



- Gurobipy
- Pandas
- Itertools



Đưa lịch nhân viên đăng ký thành dạng nhị phân

```python
# Đưa thời gian rảnh thành dạng nhị phân
binary_free_times = {}
for i in free_times.index:
    for day in days:
        if free_times[f'{day}_IN'][i] <= 6 and free_times[f'{day}_OUT'][i] >= 10 :
            binary_free_times[i, day, "morning"] = 1
        else: binary_free_times[i, day, "morning"] = 0
        if free_times[f'{day}_IN'][i] <= 11 and free_times[f'{day}_OUT'][i] >= 13:
            binary_free_times[i, day, "noon"] = 1
        else: binary_free_times[i, day, "noon"] = 0
        if free_times[f'{day}_IN'][i] <= 12 and free_times[f'{day}_OUT'][i] >= 13:
            binary_free_times[i, day, "afternoon"] = 1
        else: binary_free_times[i, day, "afternoon"] = 0
        if free_times[f'{day}_IN'][i] <= 16 and free_times[f'{day}_OUT'][i] >= 18:
            binary_free_times[i, day, "evening"] = 1
        else: binary_free_times[i, day, "evening"] = 0
```

Từ thời gian rảnh mà nhân viên đăng ký, ta sẽ biết được nhân viên này làm được ca nào

$t_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ *có thể làm được ca* $\boldsymbol{h}$ *trong thứ* $\boldsymbol{d}$ *hay không*

2. Giải bài toán

Tạo mô hình

Tạo biến

```
# Tạo mô hình
m = Model("LichLamViec")
# Ghi log
m.setParam('OutputFlag', 1)
m.setParam('LogFile', 'schedule.log')
# Tạo biến
schedule = {}
off = {}
for i in free_times.index:
    for day in days:
        for hour in part_times:
            schedule[i, day, hour] = m.addVar(vtype=GRB.BINARY, name=f"Lich_{i}_{day}_{hour}")
        off[i, day] = m.addVar(vtype=GRB.BINARY, name=f"IsOff_{i}_{day}")
```

$s_{i,d,h}$: *nhân viên* $\boldsymbol{i}$ *được xếp vào ca* $\boldsymbol{h}$ *thứ* $\boldsymbol{d}$ *hay không*

$o_{i,d}$: *nhân viên* $\boldsymbol{i}$ *có nghỉ vào thứ* $\boldsymbol{d}$ *hay không*

Ràng buộc

- Ràng buộc số lượng nhân viên trong mỗi ca
- Ràng buộc thời gian làm việc của nhân viên
- Ràng buộc giữa các ca làm

```
# Thêm ràng buộc số lượng nhân viên trong mỗi ca
for day in days:
    morning = quicksum(schedule[i, day, "morning"] for i in free_times.index)
    m.addConstr(morning == 3)

    noon = quicksum(schedule[i, day, "noon"] for i in free_times.index)
    afternoon = quicksum(schedule[i, day, "afternoon"] for i in free_times.index)
    m.addConstr(noon + afternoon == 9)

    evening = quicksum(schedule[i, day, "evening"] for i in free_times.index)
    m.addConstr(evening == 2)

# Thêm ràng buộc thời gian làm việc của từng nhân viên
for i in free_times.index:
    for day in days:
        total_hours = quicksum(schedule[i, day, ca] for ca in part_times)
        m.addConstr(total_hours <= 2 * (1 - off[i, day]))

for i in free_times.index:
    for day in days:
        for hour in part_times:
            m.addConstr(schedule[i, day, hour] + off[i, day] <=1)
for i in free_times.index:
    for day in days:
        for hour in part_times:
            m.addConstr(schedule[i, day, hour] <= binary_free_times[i, day, hour] * (1-off[i, day]))

# Thêm ràng buộc để không có nhân viên nào bị làm hai ca không sát nhau
for i in free_times.index:
    for day in days:
        m.addConstr(schedule[i, day, "noon"] + schedule[i, day, "afternoon"] <= 1*(1-off[i, day]))
        m.addConstr(schedule[i, day, "morning"] + schedule[i, day, "afternoon"] <= 1*(1-off[i, day]))
        m.addConstr(schedule[i, day, "morning"] + schedule[i, day, "evening"] <= 1*(1-off[i, day]))
```

Thiết lập hàm mục tiêu min

$$\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,1}s_{i,d,1} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,2}s_{i,d,2} + 2\sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,3}s_{i,d,3} + \sum_{i=1}^{n}\sum_{d=2}^{6} r_{i,4}s_{i,d,4}$$

```
# Thiết lập hàm mục tiêu
total_hours_all = quicksum(rank1[i]*schedule[i, day, "morning"] for i in free_times.index for day in days) + quicksum(
    rank2[i]*schedule[i, day, "noon"] for i in free_times.index for day in days) + 2 * quicksum(
    rank3[i]*schedule[i, day, "afternoon"] for i in free_times.index for day in days) + quicksum(
    rank4[i]*schedule[i, day, "evening"] for i in free_times.index for day in days)

m.setObjective(total_hours_all, GRB.MINIMIZE)

# Gọi solver để giải bài toán
m.optimize()
```

Gọi solver để giải bài toán

3. Xuất lịch

Dựa vào lịch nhân viên đăng ký để xếp giờ làm phù hợp

```
schedule_out = {"NAME": [], "MON": [], "TUE": [], "WED": [], "THU": [], "FRI": []}
for i in free_times.index:
    schedule_out["NAME"].append(free_times["NAME"][i])
    for day in days:
        if schedule[i, day, "morning"].x > 0.5 and schedule[i, day, "noon"].x < 0.5 and schedule[i, day, "afternoon"].x < 0.5 and schedule[i, day, "evening"].x < 0.5:
            schedule_out[day].append("6h-10h")
        elif schedule[i, day, "morning"].x > 0.5 and schedule[i, day, "noon"].x > 0.5:
            if free_times[f'{day}_OUT'][i] == 13 :
                schedule_out[day].append("6h-13h")
            else:
                schedule_out[day].append("6h-14h")
        elif schedule[i, day, "morning"].x < 0.5 and schedule[i, day, "noon"].x > 0.5 and schedule[i, day, "afternoon"].x < 0.5 and schedule[i, day, "evening"].x < 0.5:
            if free_times[f'{day}_OUT'][i] == 13 :
                schedule_out[day].append("9h-13h")
            elif free_times[f'{day}_OUT'][i] == 14:
                schedule_out[day].append("10h-14")
            elif free_times[f'{day}_IN'][i] == 11 or free_times[f'{day}_OUT'][i] == 15 :
                schedule_out[day].append("11h-15h")
            else:
                schedule_out[day].append("10h-14h")
        elif schedule[i, day, "morning"].x < 0.5 and schedule[i, day, "noon"].x > 0.5 and schedule[i, day, "afternoon"].x < 0.5 and schedule[i, day, "evening"].x > 0.5:
            if free_times[f'{day}_IN'][i] == 11:
                schedule_out[day].append("11h-18h")
            else:
                schedule_out[day].append("10h-18h")
        elif schedule[i, day, "morning"].x < 0.5 and schedule[i, day, "noon"].x < 0.5 and schedule[i, day, "afternoon"].x > 0.5 and schedule[i, day, "evening"].x < 0.5:
            if free_times[f'{day}_OUT'][i] == 13:
                schedule_out[day].append("12h-13h")
            elif free_times[f'{day}_OUT'][i] == 14:
                schedule_out[day].append("12h-14h")
            elif free_times[f'{day}_OUT'][i] == 15:
                schedule_out[day].append("12h-15h")
            else:
                schedule_out[day].append("12h-16h")
        elif schedule[i, day, "morning"].x < 0.5 and schedule[i, day, "noon"].x < 0.5 and schedule[i, day, "afternoon"].x > 0.5 and schedule[i, day, "evening"].x > 0.5:
            schedule_out[day].append("12h-18h")
        elif schedule[i, day, "morning"].x < 0.5 and schedule[i, day, "noon"].x < 0.5 and schedule[i, day, "afternoon"].x < 0.5 and schedule[i, day, "evening"].x > 0.5:
            if free_times[f'{day}_IN'][i] == 14:
                schedule_out[day].append("14h-18h")
            elif free_times[f'{day}_IN'][i] == 15:
                schedule_out[day].append("15h-18h")
            elif free_times[f'{day}_IN'][i] == 16:
                schedule_out[day].append("16h-18h")
            else:
                schedule_out[day].append("16h-18h")
        else: schedule_out[day].append("OFF")
data = pd.DataFrame(schedule_out)
print("\nLịch làm việc của các nhân viên:")
print(data)
data.to_excel('schedule_out.xlsx', index=False)
```

Kết quả

NAME	MON	TUE	WED	THU	FRI
A	6h-14h	6h-14h	6h-10h	6h-14h	6h-10h
B	10h-14h	10h-18h	10h-14h	10h-18h	10h-14h
C	OFF	6h-14h	6h-14h	OFF	10h-14
D	OFF	OFF	9h-13h	OFF	OFF
E	16h-18h	OFF	10h-18h	12h-18h	OFF
F	9h-13h	OFF	9h-13h	10h-14	OFF
G	6h-14h	OFF	12h-16h	12h-16h	9h-13h
H	OFF	9h-13h	OFF	OFF	9h-13h
I	OFF	6h-10h	OFF	6h-10h	6h-10h
J	15h-18h	OFF	10h-18h	9h-13h	10h-18h
K	10h-14h	10h-14h	OFF	OFF	10h-18h
L	OFF	OFF	OFF	OFF	OFF
M	6h-14h	11h-15h	OFF	OFF	11h-15h
N	OFF	10h-14	9h-13h	9h-13h	9h-13h
O	9h-13h	OFF	OFF	OFF	OFF
P	11h-15h	11h-15h	OFF	11h-15h	OFF
Q	10h-14h	10h-18h	6h-14h	6h-14h	6h-14h

Thank!

Code, slide và dữ liệu đầu vào:
https://drive.google.com/drive/folders/12vbTpCU2u0HoDu4p78kj_U9S4EWT5KH2?usp=drive_link

Tham khảo:

- Ví dụ trong slide: https://www.gurobi.com/case_studies/complevo-optimal-workforce-planning/
- Bài toán được lấy cảm hứng từ HUSCafeteria
- Nguồn dữ liệu: lấy từ lịch đăng ký giờ làm hàng tuần của các nhân viên HUSCafeteria