פרויקט גמר
מבוא לאופטימיזציה

שם המרצה : אריאל רוזנפלד

מספר הקורס : 89589

נושא הפרויקט :

פתרון לבעיית שיבוץ תורנויות של מתמחים

מגישות :

שובל הראל 315342360

נועם לחמני 318868312

בס"ד

<u>שיבוץ תורנויות של מתמחים</u>

ראשית דבר:

(שובל:) אחותי היא מתמחה בגינקולוגיה כבר מספר שנים. לאחרונה, היא קיבלה תפקיד שבמסגרתו היא אחראית על שיבוץ המתמחים לתורנויות לילה. על מנת לשבץ את המתמחים לתורנויות היא נאלצת לקחת בחשבון גם שיבוצים למרפאות (במידה ויש במהלך החודש), חופשים של מתמחים, ועוד אילוצי מערכת נוספים. מכיוון שהיא חדשה בתפקיד יצא לי לשמוע המון על כמה שזה "סיוט" ולא פשוט ושהיא יושבת על זה שעות(!!). מהרגע הראשון ששמעתי אותה מדברת על זה, זה מיד התחבר לי לנושא הקורס – בעיות אופטימיזציה.

לכן, כאשר התקרבנו לעבודת גמר הקורס, ישר ידעתי שארצה להציע לנועם שננסה לפתור את הבעיה הזו ביחד. (עד כאן שובל)

לפני שניגשנו לביצוע העבודה, ביצענו שיחה עם אחות של שובל כדי לקבל את התמונה המלאה – מה המידע הנתון כל חודש ומהם אילוצי המערכת. במהלך העבודה על הבעיה הזו, נאלצנו ללמוד ספריות חדשות ולתרגם את הבעיה לקוד שמוגדר על פי מה שלמדנו בכיתה. היו לא מעט אתגרים, אבל הרצון לפתור בעיה <u>אמיתית</u> מהחיים האמיתיים היה גדול ואנחנו שמחות מאוד בתוצאה הסופית שהתקבלה.

אין ספק שאחות של שובל מחכה בכיליון עיניים שנראה לה את מה שעשינו ונבצע איתה שיבוצים עם מידע רלוונטי.

<u>תיאור הבעיה:</u>

שם הבעיה – Schedule Residents (לו"ז מתמחים).

במהלך החודש, מתמחים צריכים להיות משובצים כתורנים לחדר לידה / מיון נשים / חדר יולדות, וכן למרפאות שונות שיש באותו החודש בהתאם לצרכי המרפאה.

מטרתנו לפתור את בעיית שיבוץ התורנויות למתמחים לחודש שלם (30 ימים).

הבעיה שלנו מוגדרת כבעיית מינימום – המטרה היא להקטין את מספר הימים הלא נוחים של המתמחים התורנים בהם הם משובצים בניגוד להעדפותיהם.

לכל מתמחה יש דרגה הממוספרת בין 1 ל3, כאשר 1 היא הדרגה הנמוכה ביותר, ו3 היא הדרגה הגבוהה ביותר.

דרגה 1 - מאפשרת למתמחה להיות תורן בחדר יולדות.

דרגה 2 - מוסיפה למתמחה את האפשרות להיות תורן גם במיון נשים.

דרגה 3 - מוסיפה למתמחה את האפשרות להיות תורן גם בחדר לידה.

נחדד - כל מתמחה בדרגה מסוימת יכול לבצע משימה שמותאמת לדרגתו או לאחת מהדרגות שמתחתיו, אך לא לדרגה מעליו.

מטרתנו ליצור לו"ז תורנויות לכל מתמחה שעומד באילוצים הבאים:

1. בכל משמרת מספר התורנים הוא 3 - תורן לחדר היולדות, למיון הנשים ולחדר הלידה.
2. סכום הדרגות של התורנים בכל יום יהיה לכל הפחות 6 (דרגת חדר לידה + דרגת חדר מיון + דרגת חדר יולדות לכל הפחות 6)
3. כל תורן לא יבצע יותר מ6 תורנויות בחודש.
4. בין כל תורנות יחלפו לפחות יומיים. לא ייווצר מצב של תורנות יום כן יום לא.
5. מספר השיבוצים לכל מתמחה לתורנויות יהיה בסטייה של לכל היותר יום אחד מממוצע מספר התורנויות של כלל המתמחים.
6. ביום שמתמחה משובץ למרפאה, הוא לא יוכל להיות משובץ לתורנות באותו יום, ולא ביום שלפניו.
7. סך המתמחים שמשובצים במרפאות יהיה כמספר השיבוצים הנדרשים למרפאה באותו היום.
8. מספר השיבוצים לכל מתמחה במרפאות יהיה בסטייה של לכל היותר יום אחד מממוצע מספר השיבוצים של כלל המתמחים למרפאות.

גישת הפתרון:

בעיה זו היא בעיית integer and binary linear programming, ודרך הפתרון בה בחרנו לפתור את הבעיה היא באמצעות Branch and cut. Branch and cut , היא דרך מתוחכמת יותר מ-Branch and Bound כאשר מתבצעים חיתוכים של ענפים שלא רלוונטיים (cutting planes).
הסיבה בה בחרנו לפתור בדרך זו, היא מכיוון שהמשתנים שלנו הם בינאריים והבעיה אותה אנו רוצים לפתור היא ליניארית.

נרחיב קצת על Branch and Cut :

כפי שתיארנו, Branch and Cut זה בעצם שילוב של Branch and Bound עם cutting planes.
נזכיר את הרעיון של Branch and Bound עבור פתרון בינארי: הרעיון הוא לקחת משתנה ולחלק לשניים: פעם אחת שהמשתנה הוא 1 ופעם שניה כשהמשתנה הוא 0. כלומר אנחנו לוקחים את הבעיה המקורית ומחלקים לשתי תתי בעיות.
כעת, בשורש העץ נמצאים כל המשתנים שלנו. לאחר בחירת משתנה נוציא שני בנים לשורש הזה, כאשר בן אחד הוא עבור השמה 0 למשתנה, והבן השני הוא כאשר ההשמה למשתנה היא 1.
כך נמשיך וניצור עוד "בנים" מכל קודקוד, כאשר בכל פעם נבחר משתנה שעוד לא ביצענו לו השמה בקודקודים מעלינו, ונפצל אותו להשמה של 0 ו-1. בכל קודקוד נבדוק את כל האילוצים עם ההשמות שביצענו עד כה. אם יש אילוץ שאינו מתקיים נבצע puring לענף הזה. כמו כן, בכל קודקוד ננסה לפתור את הבעיה ללא האילוץ הבינארי. כאשר נגיע לקודקוד שהפתרון האופטימלי שלו (הלא בינארי שלו) מביא לתוצאה קטנה/גדולה בהתאם לבעיית המקסימום/מינימום מפתרון אחר שנמצא בקודקוד אחר– נסיק שאין

טעם להמשיך לפתח את העץ מהקודקוד הזה, כי גם במצב הכי אופטימי שלהם הם מביאים פתרון טוב. זה בעצם הייחוד של האלגוריתם הזה, כי במקום לעבור על 2^n נעבור על הרבה פחות כאשר נוכל לחתוך ענפים לא רלוונטיים.

על Branch and Bound נוסיף את cutting planes . הרעיון של cutting planes הוא הוספת אילוצים כך שנוכל לחתוך ענפים בשלב מוקדם ככל האפשר.
שילוב של השניים מביא לתוצאות מהירות יותר מאשר ה- Branch and Bound הרגיל.

<u>הגדרת הקלט:</u>

2 טבלאות אקסל:

1) <u>טבלת מתמחים residents</u>

הטבלה מציינת עבור כל מתמחה את הדרגה שלו על ידי הספרות 1-3.
בנוסף הטבלה מכילה את כל תאריכי החודש ועבור כל מתמחה מסומנת ההעדפה שלו לעבוד ביום זה ע"פ המחוון הבא:
אם המתמחה יכול לעבוד באותו יום יסומן 1.
אם המתמחה מעדיף שלא לעבוד באותו יום יסומן 50.
אם המתמחה כלל לא יכול לעבוד באותו יום יסומן 500.

2) <u>טבלת מרפאות clinics</u>

בטבלה זו מוצגים כל התאריכים שבהם יש מרפאות בבית החולים, ועבור כל תאריך כזה מופיע מספר המתמחים הנדרשים.

<u>הגדרת הפלט</u>:

טבלת שיבוץ של המתמחים לתורנויות ומרפאות לכל החודש לפי תאריך ומשימה (יולדות / נשים/ לידה / מרפאות)

<u>משתני החלטה:</u>

<u>נסמן</u>:

מספר המתמחים – $n = 25$

מספר הימים – $D = 30$

סוג תורנות - $i \in \{1,2,3\}$ כאשר 1=יולדות, 2=מיון נשים, 3=חדר לידה

אינדקס ליום ספציפי – $d \in \{1,2,\dots,D\}$

אינדקס למתמחה ספציפי $j \in \{1,2,\dots,n\}$

דרגת מתמחה j: $r_j \in \{1,2,3\}$

מספר התורנים הנדרשים למרפאות ביום ספציפי $c_d \in \{0,1,2,\dots,n\}$

העדפת שיבוץ – $p \in \{1,50,500\}$ כאשר $p_{j,d}$ מסמל את ההעדפה של תורן j ביום d.

משתני ההחלטה הם:

Shifts – $x(j,d)$ עבור מתמחה j שמשובץ לתורנות ביום d.

Clinics - $y(j,d)$– עבור מתמחה j שמשובץ למרפאה ביום d.

פונקציית הobjective:

$$\min\sum_{j=1}^{n}\sum_{d=1}^{D} x(j,d) * p_{j,d}$$

הגדרת האילוצים:

1. בכל יום יש בדיוק 3 מתמחים בתורנות

מתמחה אחד בחדר יולדות, מתמחה שני במיון נשים ומתמחה שלישי בחדר לידה.

עבור כל יום d בחודש, נוסיף את האילוץ הבא:

$$\sum_{j=1}^{n} x(j,d) = 3$$

```
# Every day we need exactly 3 residents for hospital shifts
for day in range(MONTH_DAYS):
    constraint = None
    for resident in data.keys():
        constraint += data[resident]["my_hospital_shifts"][day]
    prob += constraint == 3
```

2. סכום הדרגות של המתמחים המשובצים לתורנויות בכל יום יהיה לכל הפחות 6:

נרצה שבכל יום ישוריינו כל שלושת התורנויות. הסכום המינימלי של הדרגות שיכול לקיים את זה הוא 6 (1+2+3).

עבור כל יום d בחודש, נוסיף את האילוץ הבא:

$$\sum_{j=1}^{n} r_j * x(j,d) \geq 6$$

```
# Sum of all residents degree that work on specific day has to be at least 6
for day in range(MONTH_DAYS):
    constraint = None
    for resident in data.keys():
        constraint += data[resident]["my_hospital_shifts"][day] * data[resident]["degree"]
    prob += constraint >= 6
```

3. כל מתמחה לא יבצע יותר מ6 תורנויות בחודש

עבור כל מתמחה j יתקיים האילוץ הבא:

$$\sum_{d=1}^{D} x(j,d) \leq 6$$

```
# Resident can work at most 6 hospital shifts in a month
for resident in data.keys():
    constraint = None
    for day in range(MONTH_DAYS):
        constraint += data[resident]["my_hospital_shifts"][day]
    prob += constraint <= 6
```

4. <u>לא יום כן יום לא –</u>

נרצה שלכל מתמחה יתקיים שיחלפו לפחות יומיים בין התורנויות שלו.

לכן הגדרנו את האילוץ כך שלכל מתמחה, סכום כל שלשה של ימים צמודים יהיה לכל היותר 1.

לכל מתמחה j :

לכל d כך ש : $1 \leq d \leq D-2$ יתקיים האילוץ הבא :

$$x(j,d) + x(j,d+1) + x(j,d+2) \leq 1$$

```
# a resident can't work day on day off. At least 2 shifts free between residents shift
for resident in data.keys():
    for day in range(MONTH_DAYS - 2):
        prob += data[resident]["my_hospital_shifts"][day] + \
                data[resident]["my_hospital_shifts"][day + 1] + data[resident]["my_hospital_shifts"][day + 2] <= 1
```

5. <u>מספר התורנויות לכל מתמחה יהיה בסטייה של לכל היותר יום אחד מממוצע מספר התורנויות של כלל המתמחים</u>

אילוץ זה חשוב מפני שנרצה לשמור כמה שיותר על חלוקה הוגנת בין המתמחים.

נחשב את הממוצע – $\text{avg} = \frac{D*3}{n}$

לכל מתמחה j מתקיימים שני האילוצים הבאים :

$$\sum_{d=1}^{D} x(j,d) \leq avg + 1$$

$$\sum_{d=1}^{D} x(j,d) \geq avg - 1$$

```
# The amount of days a resident has hospital shifts in a month will be +-average
average = 90 / NUM_RESIDENTS
for resident in data.keys():
    constraint = None
    for day in range(MONTH_DAYS):
        constraint += data[resident]["my_hospital_shifts"][day]
    prob += constraint <= average + 1
    prob += constraint >= average - 1
return prob
```

6. <u>ביום שמתמחה משובץ למרפאה, הוא לא יוכל להיות משובץ לתורנות באותו יום, ולא ביום שלפניו (בהנחה שלא מדובר ביום הראשון של החודש – נתייחס לכך בהמשך)</u>

לכל מתמחה j :

לכל d כך ש : $1 \leq d \leq D-2$

יתקיים האילוץ הבא :

אם d=1 :

$$x(j,d) + y(j,d) \leq 1$$

אחרת :

$$x(j,d) + \frac{1}{2}y(j,d) + \frac{1}{2}y(j,d-1) \ \leq 1$$

```
# When filling a clinic shift, resident cannot have a hospital shift on same day or day before
for resident in data.keys():
    for day in range(MONTH_DAYS):
        if day==0:
            prob += data[resident]["my_clinics"][day]+data[resident]["my_hospital_shifts"][day]<=1
        else:
            prob += data[resident]["my_clinics"][day] + 0.5*data[resident]["my_hospital_shifts"][day] + \
                    0.5*data[resident]["my_hospital_shifts"][day - 1] <= 1
```

7. סך המתמחים שמשובצים במרפאות יהיה כמספר המתמחים הנדרשים למרפאה באותו היום.

לכל יום d בחודש מתקיים האילוץ הבא :

$$\sum_{j=1}^{n} y(j,d) = c_d$$

```
# number of residents that do a clinic shift must be exactly the number of residents needed at that day in clinics
for day in range(MONTH_DAYS):
    sum_residents_per_day = None
    for resident in data.keys():
        sum_residents_per_day += data[resident]["my_clinics"][day]
    prob += sum_residents_per_day == clinics[day]
```

8. <u>מספר השיבוצים לכל מתמחה במרפאה יהיה בסטייה של לכל היותר יום אחד מממוצע מספר השיבוצים למרפאות של כלל המתמחים.</u>

נחשב את הממוצע – $\text{avg} = \frac{\sum c_d}{n}$

לכל מתמחה j מתקיימים שני האילוצים הבאים:

$$\sum_{d=1}^{D} y(j,d) \leq avg + 1$$

```
# The amount of days a resident has clinic shifts in a month will be +-average
average_clinics = num_of_clinics / NUM_RESIDENTS
for resident in data.keys():
    constraint = None
    for day in range(MONTH_DAYS):
        constraint += data[resident]["my_clinics"][day]
    prob += constraint <= average_clinics + 1
    prob += constraint >= average_clinics - 1
return prob
```

$$\sum_{d=1}^{D} y(j,d) \geq avg - 1$$

<u>הרצת האלגוריתם:</u>

<u>הרצה מספר 1:</u>

הרצנו את האלגוריתם עם כל האילוצים שהגדרנו. קיבלנו את הפלט הבא:

```
Objective value:                678.00000000
Enumerated nodes:               11132
Total iterations:               329855
Time (CPU seconds):             51.76
Time (Wallclock seconds):       51.76
```

	Yoldot	Nashim	Leda	clinics
Jan-01	resident 3	resident 22	resident 13	['resident 5', 'resident 14', 'resident 15', 'resident 20', 'resident 23']
Jan-02	resident 24	resident 23	resident 20	
Jan-03	resident 11	resident 8	resident 7	
Jan-04	resident 6	resident 22	resident 15	
Jan-05	resident 12	resident 16	resident 14	
Jan-06	resident 19	resident 2	resident 13	['resident 5', 'resident 6', 'resident 7', 'resident 18', 'resident 22']
Jan-07	resident 3	resident 8	resident 21	
Jan-08	resident 10	resident 22	resident 14	['resident 16']
Jan-09	resident 25	resident 17	resident 7	['resident 6', 'resident 9']
Jan-10	resident 6	resident 16	resident 1	
Jan-11	resident 24	resident 5	resident 4	
Jan-12	resident 18	resident 22	resident 14	['resident 1', 'resident 2', 'resident 16', 'resident 19']
Jan-13	resident 9	resident 23	resident 1	['resident 7', 'resident 8']
Jan-14	resident 11	resident 2	resident 21	['resident 4', 'resident 8', 'resident 17', 'resident 24']
Jan-15	resident 3	resident 5	resident 15	['resident 1', 'resident 10', 'resident 18']
Jan-16	resident 18	resident 17	resident 13	
Jan-17	resident 9	resident 2	resident 7	
Jan-18	resident 12	resident 23	resident 21	
Jan-19	resident 10	resident 8	resident 20	['resident 3', 'resident 4', 'resident 13', 'resident 15', 'resident 19']
Jan-20	resident 18	resident 16	resident 15	
Jan-21	resident 6	resident 7	resident 14	
Jan-22	resident 11	resident 5	resident 4	['resident 2', 'resident 9', 'resident 12', 'resident 13', 'resident 17', 'resident 23', 'resident 25']
Jan-23	resident 10	resident 2	resident 1	
Jan-24	resident 24	resident 17	resident 20	['resident 3', 'resident 11', 'resident 21']
Jan-25	resident 19	resident 5	resident 15	
Jan-26	resident 25	resident 16	resident 21	
Jan-27	resident 9	resident 8	resident 4	
Jan-28	resident 19	resident 23	resident 1	['resident 10', 'resident 12', 'resident 14', 'resident 20', 'resident 21', 'resident 22', 'resident 25']
Jan-29	resident 25	resident 17	resident 13	
Jan-30	resident 12	resident 4	resident 20	

```
resident 1, Jan-09, Jan-12, Jan-22, Jan-27
resident 2, Jan-05, Jan-13, Jan-16, Jan-22
resident 3, Jan-30, Jan-06, Jan-14
resident 4, Jan-10, Jan-21, Jan-26, Jan-29
resident 5, Jan-10, Jan-14, Jan-21, Jan-24
resident 6, Jan-03, Jan-09, Jan-20
resident 7, Jan-02, Jan-08, Jan-16, Jan-20
resident 8, Jan-02, Jan-06, Jan-18, Jan-26
resident 9, Jan-12, Jan-16, Jan-26
resident 10, Jan-07, Jan-18, Jan-22
resident 11, Jan-02, Jan-13, Jan-21
resident 12, Jan-04, Jan-17, Jan-29
resident 13, Jan-30, Jan-05, Jan-15, Jan-28
resident 14, Jan-04, Jan-07, Jan-11, Jan-20
resident 15, Jan-03, Jan-14, Jan-19, Jan-24
resident 16, Jan-04, Jan-09, Jan-19, Jan-25
resident 17, Jan-08, Jan-15, Jan-23, Jan-28
resident 18, Jan-11, Jan-15, Jan-19
resident 19, Jan-05, Jan-24, Jan-27
resident 20, Jan-01, Jan-18, Jan-23, Jan-29
resident 21, Jan-06, Jan-13, Jan-17, Jan-25
resident 22, Jan-30, Jan-03, Jan-07, Jan-11
resident 23, Jan-01, Jan-12, Jan-17, Jan-27
resident 24, Jan-01, Jan-10, Jan-23
resident 25, Jan-08, Jan-25, Jan-28
```

מימין ניתן לראות את השיבוצים לתורנויות לפי מתמחה.

הפלט שהתקבל תואם בדיוק לקלט הנדרש, ועומד בכל האילוצים. למשל, ניתן לראות שהחלוקה של המתמחים לתורנויות הוגנת, כל מתמחה מקיים בין 3 ל4 תורנויות. בנוסף ניתן לראות שימי התורנויות לא צמודים.

<u>הרצה מספר 2</u>:

כעת נוריד את האילוץ שמחייב לפחות יומיים בין תורנות לתורנות, וכן נגדיל את מספר התורנויות האפשריות עבור כל מתמחה מ6 ל8. להלן התוצאות:

```
Objective value:                580.00000000
Enumerated nodes:               0
Total iterations:               623
Time (CPU seconds):             0.99
Time (Wallclock seconds):       0.99
```

	Yoldot	Nashim	Leda	clinics
Jan-01	resident 6	resident 5	resident 15	['resident 14', 'resident 16', 'resident 17', 'resident 22', 'resident 23']
Jan-02	resident 24	resident 8	resident 14	
Jan-03	resident 11	resident 5	resident 21	
Jan-04	resident 10	resident 16	resident 15	
Jan-05	resident 12	resident 5	resident 21	
Jan-06	resident 3	resident 16	resident 14	['resident 6', 'resident 7', 'resident 15', 'resident 19', 'resident 22']
Jan-07	resident 3	resident 22	resident 21	
Jan-08	resident 18	resident 7	resident 21	['resident 9']
Jan-09	resident 6	resident 17	resident 14	['resident 2', 'resident 20']
Jan-10	resident 24	resident 22	resident 4	
Jan-11	resident 24	resident 17	resident 13	
Jan-12	resident 10	resident 23	resident 20	['resident 1', 'resident 2', 'resident 19', 'resident 21']
Jan-13	resident 18	resident 22	resident 1	['resident 8', 'resident 11']
Jan-14	resident 9	resident 23	resident 1	['resident 4', 'resident 8', 'resident 12', 'resident 24']
Jan-15	resident 3	resident 16	resident 1	['resident 5', 'resident 18', 'resident 25']
Jan-16	resident 18	resident 17	resident 13	
Jan-17	resident 11	resident 8	resident 4	
Jan-18	resident 19	resident 17	resident 20	
Jan-19	resident 10	resident 8	resident 20	['resident 3', 'resident 6', 'resident 13', 'resident 15', 'resident 16']
Jan-20	resident 12	resident 2	resident 15	
Jan-21	resident 6	resident 2	resident 7	
Jan-22	resident 11	resident 8	resident 4	['resident 5', 'resident 9', 'resident 13', 'resident 17', 'resident 21', 'resident 24', 'resident 25']
Jan-23	resident 19	resident 2	resident 1	
Jan-24	resident 9	resident 23	resident 7	['resident 3', 'resident 10', 'resident 11']
Jan-25	resident 19	resident 23	resident 14	
Jan-26	resident 25	resident 5	resident 13	
Jan-27	resident 25	resident 22	resident 15	
Jan-28	resident 25	resident 16	resident 13	['resident 4', 'resident 10', 'resident 12', 'resident 14', 'resident 18', 'resident 20', 'resident 23']
Jan-29	resident 9	resident 7	resident 20	
Jan-30	resident 12	resident 2	resident 4	

ניתן לראות שכמות האיטרציות פחתה בצורה משמעותית וכן גם זמן הריצה.

בנוסף גם ערך הobjective ירד.

נציג את טבלת השיבוצים מהרצה זו. ניתן לראות שישנם מתמחים שמבצעים תורנויות יום אחר יום, כלומר האילוץ אכן לא מתקיים. (הדגשנו כמה דוגמאות לכך)

מהתוצאות שהתקבלו אנו לומדים שכשמפחיתים אילוצים משמעותיים ומקלים על האלגוריתם, הוא מוצא פתרון בזמן קצר יותר.

<u>הרצה מספר 3 :</u>

כעת בחרנו עובד בדרגה 2, ו"קידמנו" אותו להיות בדרגה 3.

להלן התוצאות :

```
Objective value:                678.00000000
Enumerated nodes:               1
Total iterations:               665
Time (CPU seconds):             0.96
Time (Wallclock seconds):       0.96
```

הגענו לפתרון שונה, עם objective זהה ובזמן קצר יותר.

	Yoldot	Nashim	Leda	clinics
Jan-01	resident 19	resident 22	resident 13	['resident 5', 'resident 6', 'resident 14', 'resident 15', 'resident 23']
Jan-02	resident 12	resident 7	resident 20	
Jan-03	resident 11	resident 2	resident 21	
Jan-04	resident 24	resident 22	resident 15	
Jan-05	resident 12	resident 5	resident 14	
Jan-06	resident 11	resident 16	resident 17	['resident 6', 'resident 7', 'resident 13', 'resident 19', 'resident 22']
Jan-07	resident 3	resident 22	resident 21	
Jan-08	resident 10	resident 8	resident 14	['resident 9']
Jan-09	resident 12	resident 7	resident 17	['resident 3', 'resident 24']
Jan-10	resident 6	resident 22	resident 15	
Jan-11	resident 24	resident 4	resident 13	
Jan-12	resident 3	resident 8	resident 21	['resident 1', 'resident 2', 'resident 16', 'resident 19']
Jan-13	resident 6	resident 23	resident 17	['resident 1', 'resident 10']
Jan-14	resident 25	resident 5	resident 14	['resident 4', 'resident 8', 'resident 11', 'resident 24']
Jan-15	resident 3	resident 16	resident 1	['resident 9', 'resident 18', 'resident 22']
Jan-16	resident 18	resident 13	resident 20	
Jan-17	resident 9	resident 4	resident 7	
Jan-18	resident 19	resident 23	resident 17	
Jan-19	resident 10	resident 2	resident 1	['resident 4', 'resident 8', 'resident 13', 'resident 15', 'resident 16']
Jan-20	resident 18	resident 13	resident 20	
Jan-21	resident 9	resident 8	resident 14	
Jan-22	resident 6	resident 16	resident 4	['resident 2', 'resident 5', 'resident 12', 'resident 17', 'resident 20', 'resident 21', 'resident 25']
Jan-23	resident 10	resident 2	resident 1	
Jan-24	resident 24	resident 23	resident 7	['resident 11', 'resident 17', 'resident 21']
Jan-25	resident 19	resident 5	resident 15	
Jan-26	resident 25	resident 16	resident 21	
Jan-27	resident 9	resident 8	resident 4	
Jan-28	resident 11	resident 23	resident 1	['resident 7', 'resident 10', 'resident 12', 'resident 14', 'resident 18', 'resident 20', 'resident 25']
Jan-29	resident 25	resident 5	resident 15	
Jan-30	resident 18	resident 2	resident 20	

<u>מקרה קצה לפיתוח הפתרון:</u>

בפתרוננו אנו מתייחסים לכל חודש בנפרד. אחד האילוצים מחייב שלא ניתן לשבץ מתמחה לתורנות יום אחר יום. עם זאת בפתרוננו ייתכן מצב שבו מתמחה יעבוד ביום האחרון לחודש וביום הראשון לחודש הבא וזה עדיין ייחשב כפתרון חוקי. יש צורך להתמודד עם המצב הזה.

<u>סיכום ומסקנות:</u>

הבעיה אותה ניסינו לפתור היא שיבוץ מתמחים לתורנויות שונות בבית החולים ולמרפאות, תוך כדי התחשבות בהעדפותיהם של המתמחים לשיבוץ לתורנויות השונות, ושמירה על אילוצים שונים.

ראינו שכאשר מורידים אילוצים לאלגוריתם לוקח פחות זמן למצוא פתרון, והוא מבצע פחות איטרציות.

בנוסף, בדקנו את האלגוריתם על קלטים שונים, והבחנו שיש חשיבות להוספת אופציות לשיבוצים.

למשל, כאשר הגדלנו את מספר השיבוצים האפשריים על ידי הגדלת מספר המתמחים שדרגתם 3 (במקום דרגה 2), מספר האיטרציות קטן. הסיבה לכך היא שמתמחה בדרגה 3 יכול לעשות יותר דברים מאשר מתמחה בדרגה 2. לכן "הקלנו" על האלגוריתם והוא מצא פתרון בזמן קצר יותר.

ביצענו הרצות שונות, אותן לא הצגנו בעבודתנו. לא תמיד נמצא פתרון לבעיה ולכן במקרים כאלה היינו מציעות "לרכך" אילוצים שלא בהכרח לא מתאפשרים מבחינה פיזית. לדוגמה, שיבוץ במרפאה ובתורנות באותו היום הוא שיבוץ שייתכן שיפגע בנוחות התורנים, אם כי הוא שיבוץ אפשרי במידה ולא נמצא פתרון, ולכן כאן יש מקום להתגמש.

בנימה אישית, נהננו מאוד מהעבודה!

תודה רבה!