

דו"ח מסכם  
תרגיל בית 1

יונתן בתן  
302279138  
[Yonibettan@gmail.com](mailto:Yonibettan@gmail.com)

עמרי פרוינד  
301695490  
[omrifro@gamil.com](mailto:omrifro@gamil.com)

## חלק א'

### סעיף 1:

מספר הפרמוטציות עבור  $k = 1, 2, \dots, 10$  ו-  $l = 5$

| K  | ללא אילוץ דלק<br>$k!$ | עם אילוץ דלק<br>$(k!)l^{k-1}$ |
|----|-----------------------|-------------------------------|
| 1  | 1                     | 1                             |
| 2  | 2                     | 10                            |
| 3  | 6                     | 150                           |
| 4  | 24                    | 3000                          |
| 5  | 120                   | 75000                         |
| 6  | 720                   | 2250000                       |
| 7  | 5040                  | 78750000                      |
| 8  | 40320                 | 3150000000                    |
| 9  | 362880                | 141750000000                  |
| 10 | 3628800               | 7087500000000                 |

## חלק ב'

אין תרגילים

## חלק ג'

### סעיף 2:

בגבול העליון יש לנו  $k$  לקוחות הממתינים למשלוח ו-  $l$  תחנות דלק אפשריות ולכן מקדם הסיעוף המקסימלי הינו  $k + l$ . בגבול התחתון כל ההזמנות סופקו וכבר ביקרנו בכל תחנות הדלק ולכן מקדם הסיעוף הינו 0.

סה"כ נקבל  $max = k + l$ ,  $min = 0$ .

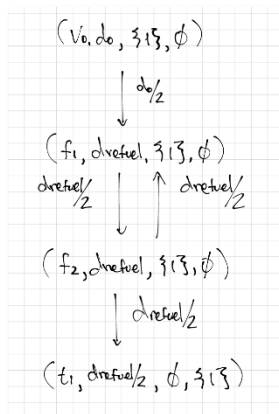
### סעיף 3:

יתכנו מעגלים.

כיוון שהגעה לתחנת דלק אינה משנה את רשימות המשלוחים ומצב הדלק במצבים אלו הינו  $d_{refuel}$  כל עוד המרחק בין 2 תחנות דלק קטן מ-  $d_{refuel}$  יהיה קיים אופרטור מאחת לשנייה ונקבל מעגל.

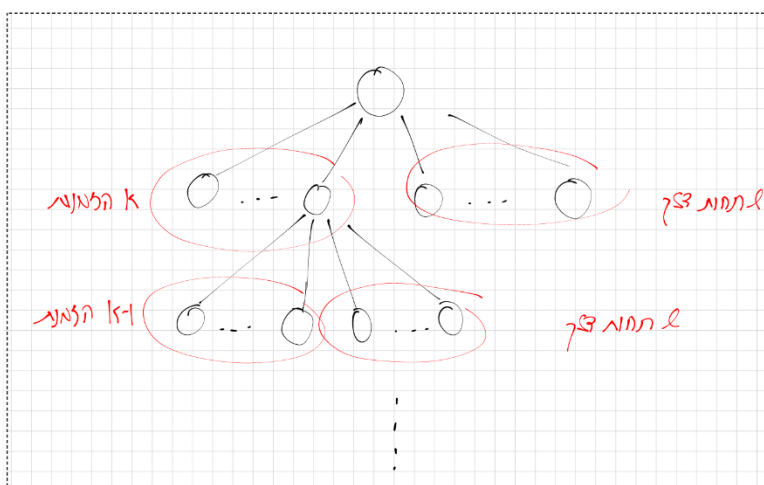
לדוגמא עבור  $GasStations = \{f1, f2\}$ ,  $Ord = \{t1\}$  וקבוצת המצבים

$S_d = \{(v_0, d_0, \{1\}, \emptyset), (f_1, d_{refuel}, \{1\}, \emptyset), (f_2, d_{refuel}, \{1\}, \emptyset), (t_1, d_{refuel}, \emptyset, \{1\})\}$  כאשר  $dist(u, v)$  מסומן על הקשתות לכל  $u, v$  צמתים של מצבים מסוימים נקבל את המעגל הבא:



### סעיף 4:

- מהמצב הראשון יש לנו  $k + l$  יורשים אפשריים
- מהמצב השני יש לנו  $(k - 1) + l$  יורשים אפשריים
- אנו נעצור כאשר לכל צומת יש רק  $l$  עלים
- סה"כ יש לנו  $\frac{(k+l)!}{l!}$  מצבים



### סעיף 5:

יתכנו בורות.

אם החלטנו ללכת לספק משלוח, שאינו המשלוח האחרון, במקום ללכת לתדלק והגענו למצב שאין לנו מספיק דלק כדי להגיע למשלוח הבא או לתחנת דלק קרובה נתקע בבור.

$$Succ_t((v_1, d_1, T_1, F_1)) = \left\{ (t_i, d, T, F) \left| \begin{array}{l} t_i \in Ord \\ d_1 - dist(v_1, t_i) \geq 0 \\ d = d_1 - dist(v_1, t_i) \\ T = T_1 \setminus \{i\} \\ F = F_1 \cup \{i\} \end{array} \right. \right\}$$

$$Succ_f((v_1, d_1, T_1, F_1)) = \left\{ (f, d, T, F) \left| \begin{array}{l} f \in GasStations \\ d_1 - dist(v_1, f) \geq 0 \\ d = d_{refuel} \\ T = T_1 \\ F = F_1 \end{array} \right. \right\}$$

$$Succ((v_1, d_1, T_1, F_1)) = Succ_t((v_1, d_1, T_1, F_1)) \cup Succ_f((v_1, d_1, T_1, F_1))$$

במקרה המינימלי יהיה לנו ד"י דלק בכדי לספק את כל המשלוחים ללא עצירה בתחנת דלק ולכן עבור הוספת צומת ההתחלה נקבל עומק מינימלי של  $k$ .

- ניתן לראות את הפתרון לבעיית המפה עבור אלגוריתם חיפוש UniformCost עבור משקלי קשתות 1 ולכן האלגוריתם קביל ואנו מקבלים פתרון אופטימלי.

```
(ai_hw1) ybettan hw1 (devel) $ python main.py
load_map_from_csv: 1.87sec

Solve the map problem.
Map(src: 54 dst: 549) UniformCost time: 0.01 #dev: 212 total_cost: 12.00
000 |path|: 13 path: [ 54, 55, 56, 57, 58, 59, 60, 28893, 14580, 14590, 14591, 14592, 14593]
```

- עבור אותה בעיה כעת משקלי הקשתות אינם זהים ומייצגים את המרחק האווירי בין צמתים שונים במפה לכן מן הסתם שנקבל פתרון שונה שהרי המפה שונה.

```
(ai_hw1) ybettan hw1 (devel) $ python main.py
load_map_from_csv: 1.60sec

Solve the map problem.
Map(src: 54 dst: 549) UniformCost time: 0.64 #dev: 17355 total_cost: 7465.52560 |path|: 137 path: [ 54, 55, 56, 57, 58, 59, 60, 28893, 14580, 14590, 14591, 14592, 14593, 14594, 14595, 14596, 14597, 14598, 14599, 14600, 14601, 14602, 14603, 14604, 14605, 14606, 14607, 14608, 14609, 14610, 14611, 14612, 14613, 14614, 14615, 14616, 14617, 14618, 14619, 14620, 14621, 14622, 14623, 14624, 14625, 14626, 14627, 14628, 14629, 14630, 14631, 14632, 14633, 14634, 14635, 14636, 14637, 14638, 14639, 14640, 14641, 14642, 14643, 14644, 14645, 14646, 14647, 14648, 14649, 14650, 14651, 14652, 14653, 14654, 14655, 14656, 14657, 14658, 14659, 14660, 14661, 14662, 14663, 14664, 14665, 14666, 14667, 14668, 14669, 14670, 14671, 14672, 14673, 14674, 14675, 14676, 14677, 14678, 14679, 14680, 14681, 14682, 14683, 14684, 14685, 14686, 14687, 14688, 14689, 14690, 14691, 14692, 14693, 14694, 14695, 14696, 14697, 14698, 14699, 14700, 14701, 14702, 14703, 14704, 14705, 14706, 14707, 14708, 14709, 14710, 14711, 14712, 14713, 14714, 14715, 14716, 14717, 14718, 14719, 14720, 14721, 14722, 14723, 14724, 14725, 14726, 14727, 14728, 14729, 14730, 14731, 14732, 14733, 14734, 14735, 14736, 14737, 14738, 14739, 14740, 14741, 14742, 14743, 14744, 14745, 14746, 14747, 14748, 14749, 14750, 14751, 14752, 14753, 14754, 14755, 14756, 14757, 14758, 14759, 14760, 14761, 14762, 14763, 14764, 14765, 14766, 14767, 14768, 14769, 14770, 14771, 14772, 14773, 14774, 14775, 14776, 14777, 14778, 14779, 14780, 14781, 14782, 14783, 14784, 14785, 14786, 14787, 14788, 14789, 14790, 14791, 14792, 14793, 14794, 14795, 14796, 14797, 14798, 14799, 14800, 14801, 14802, 14803, 14804, 14805, 14806, 14807, 14808, 14809, 14810, 14811, 14812, 14813, 14814, 14815, 14816, 14817, 14818, 14819, 14820, 14821, 14822, 14823, 14824, 14825, 14826, 14827, 14828, 14829, 14830, 14831, 14832, 14833, 14834, 14835, 14836, 14837, 14838, 14839, 14840, 14841, 14842, 14843, 14844, 14845, 14846, 14847, 14848, 14849, 14850, 14851, 14852, 14853, 14854, 14855, 14856, 14857, 14858, 14859, 14860, 14861, 14862, 14863, 14864, 14865, 14866, 14867, 14868, 14869, 14870, 14871, 14872, 14873, 14874, 14875, 14876, 14877, 14878, 14879, 14880, 14881, 14882, 14883, 14884, 14885, 14886, 14887, 14888, 14889, 14890, 14891, 14892, 14893, 14894, 14895, 14896, 14897, 14898, 14899, 14900, 14901, 14902, 14903, 14904, 14905, 14906, 14907, 14908, 14909, 14910, 14911, 14912, 14913, 14914, 14915, 14916, 14917, 14918, 14919, 14920, 14921, 14922, 14923, 14924, 14925, 14926, 14927, 14928, 14929, 14930, 14931, 14932, 14933, 14934, 14935, 14936, 14937, 14938, 14939, 14940, 14941, 14942, 14943, 14944, 14945, 14946, 14947, 14948, 14949, 14950, 14951, 14952, 14953, 14954, 14955, 14956, 14957, 14958, 14959, 14960, 14961, 14962, 14963, 14964, 14965, 14966, 14967, 14968, 14969, 14970, 14971, 14972, 14973, 14974, 14975, 14976, 14977, 14978, 14979, 14980, 14981, 14982, 14983, 14984, 14985, 14986, 14987, 14988, 14989, 14990, 14991, 14992, 14993, 14994, 14995, 14996, 14997, 14998, 14999, 15000]
```

## חלק ה'

### סעיף 9:

בקוד

### סעיף 10:

כפי שניתן לראות אכן קיבלנו את אותה תוצאה עבור uniform\_cost ו-a\_star עם יוריסטיקה  $h = 0$ .

```
(ai_hwl) ybetta@hwl (dev) $ python main.py
load_map_from_csv: 1.66sec

Solve the map problem.
Map(src: 54 dst: 549)
UniformCost time: 0.59 #dev: 17355 total cost: 7465.52560 |path|: 137 path: [ 54, 55, 56, 57, 58, 59, 60, 28893, 14580
, 14590, 14591, 14592, 14593, 81892, 25814, 81, 26236, 26234, 1188, 33068, 33069, 33070, 15474, 33071, 5020, 21699, 33072, 33073, 33074, 16203, 9847, 9848, 9849, 9850, 9851, 335, 9852, 82906,
82907, 82908, 82909, 95454, 96539, 72369, 94627, 38553, 72367, 29007, 94632, 96540, 9269, 82890, 29049, 29026, 82682, 71897, 83380, 96541, 82904, 96542, 96543, 96544, 96545, 96546, 96547, 82911, 82928, 24
841, 24842, 24843, 5215, 24844, 9274, 24845, 24846, 24847, 24848, 24849, 24850, 24851, 24852, 24853, 24854, 24855, 24856, 24857, 24858, 24859, 24860, 24861, 24862, 24863, 24864, 24865, 24866, 82208, 8220
9, 82210, 21518, 21431, 21432, 21433, 21434, 21435, 21436, 21437, 21438, 21439, 21440, 21441, 21442, 21443, 21444, 21445, 21446, 21447, 21448, 21449, 21450, 21451, 621, 21452, 21453, 21454, 21495, 21496,
539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549]
Map(src: 54 dst: 549)
A* (h=0, w=0.500) time: 0.71 #dev: 17355 total cost: 7465.52560 |path|: 137 path: [ 54, 55, 56, 57, 58, 59, 60, 28893, 14580
, 14590, 14591, 14592, 14593, 81892, 25814, 81, 26236, 26234, 1188, 33068, 33069, 33070, 15474, 33071, 5020, 21699, 33072, 33073, 33074, 16203, 9847, 9848, 9849, 9850, 9851, 335, 9852, 82906,
82907, 82908, 82909, 95454, 96539, 72369, 94627, 38553, 72367, 29007, 94632, 96540, 9269, 82890, 29049, 29026, 82682, 71897, 83380, 96541, 82904, 96542, 96543, 96544, 96545, 96546, 96547, 82911, 82928, 24
841, 24842, 24843, 5215, 24844, 9274, 24845, 24846, 24847, 24848, 24849, 24850, 24851, 24852, 24853, 24854, 24855, 24856, 24857, 24858, 24859, 24860, 24861, 24862, 24863, 24864, 24865, 24866, 82208, 8220
9, 82210, 21518, 21431, 21432, 21433, 21434, 21435, 21436, 21437, 21438, 21439, 21440, 21441, 21442, 21443, 21444, 21445, 21446, 21447, 21448, 21449, 21450, 21451, 621, 21452, 21453, 21454, 21495, 21496,
539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549]
```

### סעיף 11:

כמתוכנן כיוון שהיוריסטיקה קבילה קיבלנו פתרון קביל (זהה ל- uniform cost) אך יעיל יותר מפתרון עם יוריסטיקה "טיפשה" שאינה מוסיפה מידע לאלגוריתם מעבר למחיר הצומת הנוכחי.

```
Map(src: 54 dst: 549)
A* (h=AirDist, w=0.500) time: 0.11 #dev: 2016 total cost: 7465.52560 |path|: 137 path: [ 54, 55, 56, 57, 58, 59, 60, 28893, 14580
, 14590, 14591, 14592, 14593, 81892, 25814, 81, 26236, 26234, 1188, 33068, 33069, 33070, 15474, 33071, 5020, 21699, 33072, 33073, 33074, 16203, 9847, 9848, 9849, 9850, 9851, 335, 9852, 82906,
82907, 82908, 82909, 95454, 96539, 72369, 94627, 38553, 72367, 29007, 94632, 96540, 9269, 82890, 29049, 29026, 82682, 71897, 83380, 96541, 82904, 96542, 96543, 96544, 96545, 96546, 96547, 82911, 82928, 24
841, 24842, 24843, 5215, 24844, 9274, 24845, 24846, 24847, 24848, 24849, 24850, 24851, 24852, 24853, 24854, 24855, 24856, 24857, 24858, 24859, 24860, 24861, 24862, 24863, 24864, 24865, 24866, 82208, 8220
9, 82210, 21518, 21431, 21432, 21433, 21434, 21435, 21436, 21437, 21438, 21439, 21440, 21441, 21442, 21443, 21444, 21445, 21446, 21447, 21448, 21449, 21450, 21451, 621, 21452, 21453, 21454, 21495, 21496,
539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549]
```

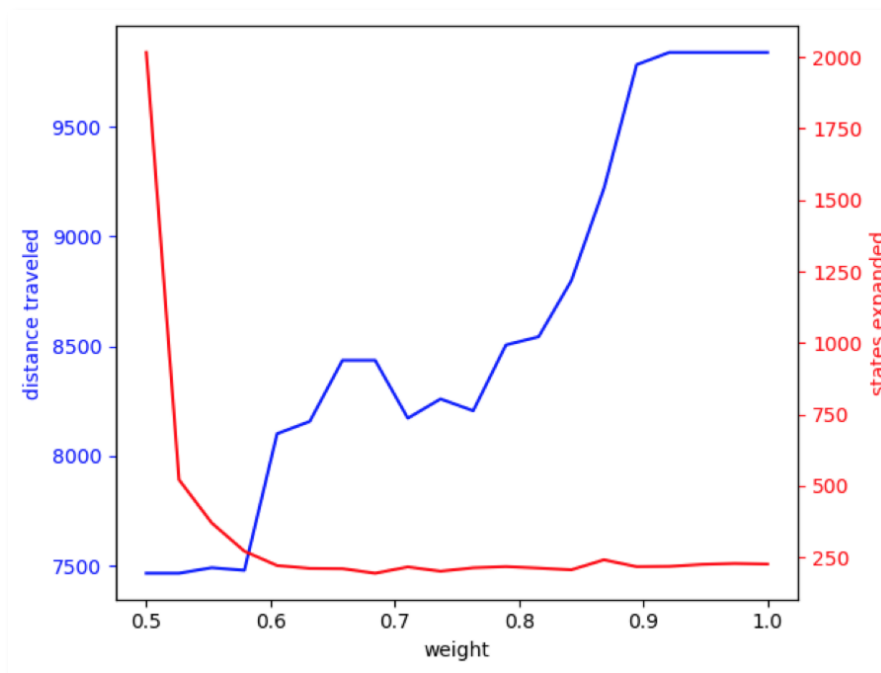
## סעיף 12:

כפי שלמדנו ככל שליוורסטיקה משקל כבד יותר האלגוריתם רץ כמה שיותר מהר למטרה על חשבון אופטימליות הפתרון וכלל שמשקל היוורסטיקה נמוך יותר כך אנחנו משפרים את אופטימליות הפתרון על חשבון זמן הריצה ולכן הגרף שקיבלנו מסתדר עם התיאוריה.

עקומת המרחק אינה מונוטונית לחלוטין שכן היוורסטיקה אינה מושלמת וניתן לתכנן גרף באופן כזה שלא יהיה פגיעה במחיר הפתרון ולכן כפי שניתן לראות המגמה מראה פגיעה באופטימליות הפתרון עבור הגדלת משקל היוורסטיקה אך לא באופן "חלק"

נקודה מעניינת היא שהקטנה המשקל הניתן ליוורסטיקה בקנה מידה קטנה פוגע בצורה משמעותית באופטימליות הפתרון.

במקרה הנ"ל  $weight \approx 0.58$  היה נותן פתרון אופטימלי כמעט באותה מידה אך בעלות זולה הרבה יותר.



## חלק ו'

## סעיף 13:

בקוד

## סעיף 14:

היוורסטיקה  $h(s) = \max_{i \in T} \text{AirDist}(s, v, t_i)$  הינה קבילה.

במידה ויש הזמנה יחידה  $j$  אז נקבל  $h(s) = \text{AirDist}(s, v, t_j) \leq \text{AirDist}(s, v, t_j)$  אחרת יש יותר מהזמנה יחידה שעדיין לא סופקה אז בהכרח סכום המרחקים האוויריים שלהם בהכרח גדול מהמרחק האווירי המקסימלי מבין כל אחת בנפרד.

## סעיף 15:

בקוד

## סעיף 16:

קיבלנו מסלול חוקי באורך 11 צמתים העובר ב- 2 תחנות דלק

```
Solve the relaxed deliveries problem.
RelaxedDeliveries(big_delivery) A* (h=MaxAirDist, w=0.500) time: 4.50 #dev: 3908 total_cost: 40844.21165 lpathl: 11 path: [33919, 18409, 77726, 26690, 31221, 63050, 84034, 60664, 70557, 94941, 31008] gas-stations: [31221, 70557]
```

## סעיף 17:

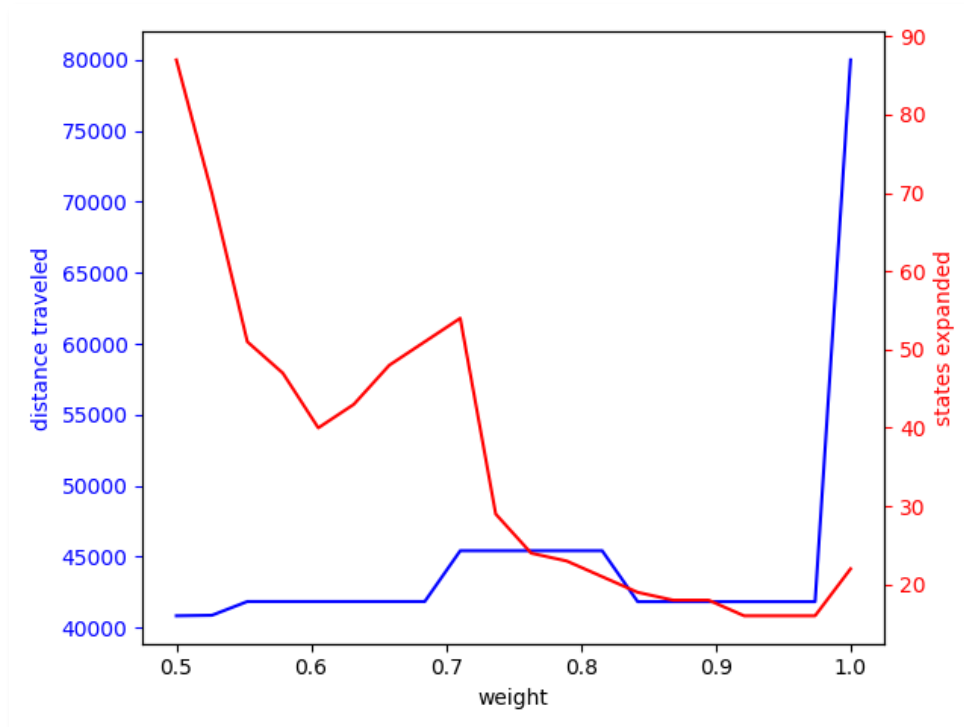
כפי שמוסבר בתרגיל למרות שהיוריסטיקה אינה מאוד מיועדת היא עדיין חוסכת לנו המון פיתוחים, וכמובן עדיין מחזירה את הפתרון האופטימלי כיוון שיוריסטיקה זו קבילה.

```
Solve the relaxed deliveries problem.
RelaxedDeliveries(big_delivery) A* (h=MaxAirDist, w=0.500) time: 5.01 #dev: 3908 total_cost: 40844.21165 lpathl: 11 path: [33919, 18409, 77726, 26690, 31221, 63050, 84034, 60664, 70557, 94941, 31008] gas-stations: [31221, 70557]
RelaxedDeliveries(big_delivery) A* (h=MSTAirDist, w=0.500) time: 1.50 #dev: 87 total_cost: 40844.21165 lpathl: 11 path: [33919, 18409, 77726, 26690, 31221, 63050, 84034, 60664, 70557, 94941, 31008] gas-stations: [31221, 70557]
```

## סעיף 18:

אמנם הבעיה אותה אנחנו פותרים הינה שונה מבעיית המפה אך עדיין נוכל לקבל תחושה כללית אינדיקטיבית. ניתן לראות שאופטימליות הפתרון יחסית חסין מפני הגדלת המשקל של היוריסטיקה (עד גבול מסוים) שכן יוריסטיקה זו מיועדת יותר מהיוריסטיקה של מרחק אווירי (הכוונה היא למרחק אווירי ישיר למרחק מטרה ולא דרך סכום המרחקים האוויריים של צמתי הביניים).

בנוסף ככל שנגדיל את המשקל הניתן לערך היוריסטי (עד גבול מסוים) לעומת מחיר הצומת כך אנחנו בעצם "רצים" מהר יותר לכיוון המטרה עם פגיעה מינימלית באופטימליות הפתרון כפי שהוסבר קודם לכן.

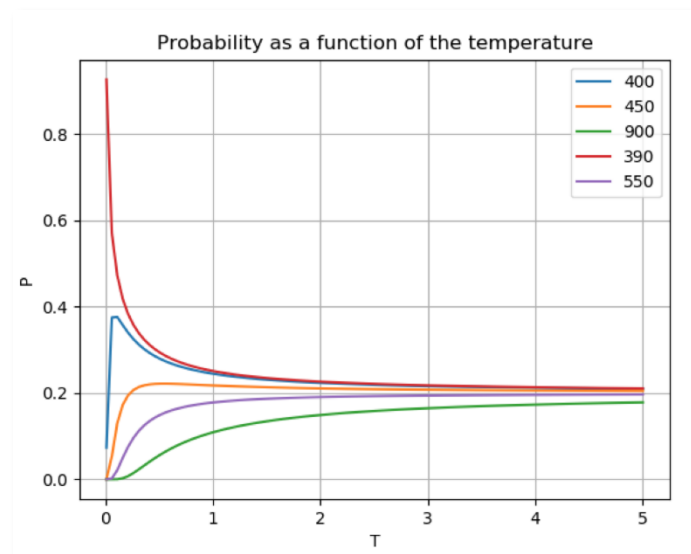


## סעיף 19:

$$\forall x_i \in x', \Pr_{normalized}(x) = \frac{\left(\frac{x_i}{\alpha}\right)^{-\frac{1}{T}}}{\sum_{pnt_h \in best\ N\ points} \left(\frac{x_h}{\alpha}\right)^{-\frac{1}{T}}} = \frac{\cancel{\left(\frac{1}{\alpha}\right)^{\frac{1}{T}}} \left(x_i\right)^{-\frac{1}{T}}}{\cancel{\left(\frac{1}{\alpha}\right)^{\frac{1}{T}}} \sum_{pnt_h \in best\ N\ points} \left(x_h\right)^{-\frac{1}{T}}} = \Pr_{not-normalized}(x)$$

## סעיף 20:

להלן הגרף שהתקבל, הסבר יינתן בסעיפים הבאים.



## סעיף 21:

כאשר הטמפרטורה אפסית אנחנו בעצם לא מאפשרים "רעש" כלומר לא נותנים לאלגוריתם הרבה חופש בחירה לבחור בצומת עם ערך יוריסטי שאינו המינימלי כלומר אינו מעניק "סקרנות" לאלגוריתם.

ניתן לראות שאכן האלגוריתם יבחר את הצומת הבא לפיתוח כאל הצומת כאל הצומת עם היוריסטיקה הנמוכה ביותר בהסתברות של כמעט 1 וכל צומת אחר בהסתברות כמעט 0, במילים אחרות יתנהג בצורה דומה לאלגוריתם דטרמיניסטי.

## סעיף 22:

עבור טמפרטורה מקסימלית האלגוריתם בוחר צומת כמעט באקראיות מוחלטת מבין  $N$  הצמתים בעלי ערך יוריסטי נמוך ביותר כלומר במקרה זה לאלגוריתם "מקסימום סטוכסטיות".

כלומר פרמטר זה בעצם מאפשר לנו לבצע tuning על מידת "ההרפתקנות" של האלגוריתם אל מול הדטרמיניסטיות שלו.

## סעיף 23:

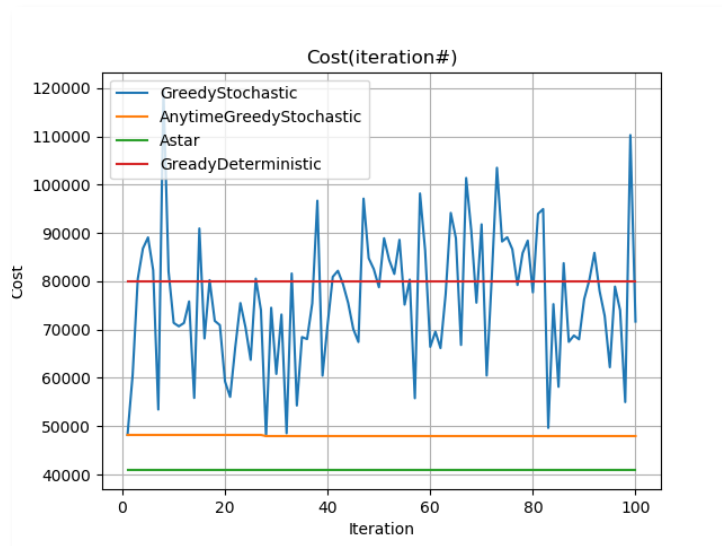
בקוד



## סעיף 24:

להלן התוצאות שקיבלנו, כיוון שבמקרה הריצה של האלגוריתם הסטוכסטי נתנה פתרון מוצלח כבר באיטרציה הראשונה קשה להבחין בגרף המונטוני יורד של any time greedy stochastic ולכן החלטנו לצרף את הגרף שהתקבלת מהמחשב שלי (שאר הגרפים הוצאו מהמחשב של השותף שלי).

כפי שציינו במייל אשר שלחנו לסגל מבנה הנתונים heapdict אשר מימשתם לצורך מימוש OPEN אינו דטרמיניסטי עבור צמתים בעלי ערך מיון זהה ולכן הגרפים של ה- greedy deterministic אינם זהים ב-2 המחשבים אך בכל זאת ניתן הסבר לגרף השני כיוון שהוא ברור יותר לצורך הסבר ההתנהגות.

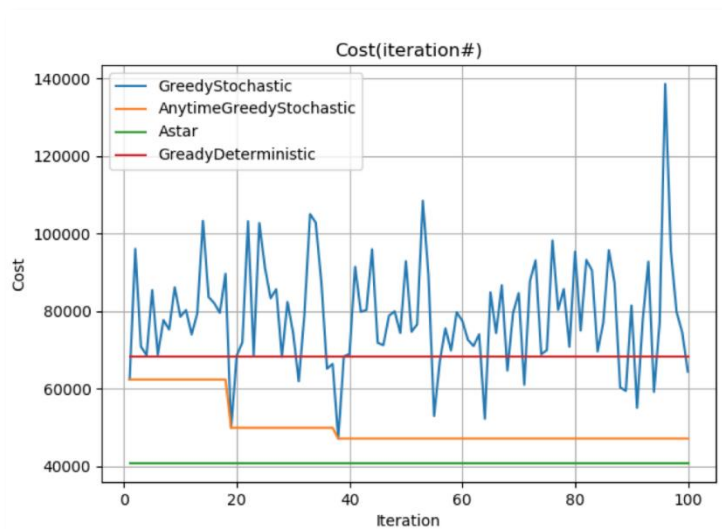


להלן הגרף שהתקבל מהמחשב שלי, נתחיל בברור מאליו A\* כידוע נותן פתרון אופטימלי עבור הירויסטיקה קבילה ולכן כצפוי לא התקבל פתרון טוב ממנו.

האלגוריתם greedy deterministic אינו נותן פתרון קביל וזה גם בהתאם לציפיות שגם אלגוריתם חמדן אינו בהכרח מספק פתרון אופטימלי.

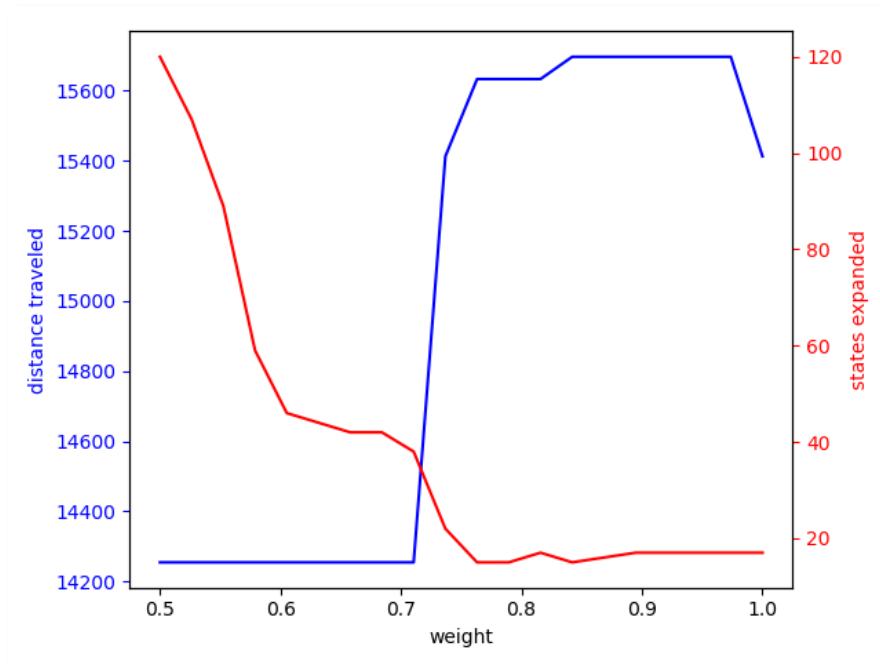
האלגוריתם greedy stochastic מאפשר בעצם יותר "הרפתקנות" מאשר הגרסה הדטרמיניסטית שלו ודבר זה גורם לעיטים למציאת פתרון טוב יותר ולעיטים פתרון טוב פחות אשר עולה בקנה אחד עם התיאוריה.

לבסוף האלגוריתם any time greedy stochastic "דוגם" את הפתרון הטוב ביותר שהתקבל עד כה באמצעות greedy stochastic ולכן מסתדר לנו שהתקבל גרף מונטוני יורד בצורת "מדרגה"



ההסבר שקול לסעיף 12, עבור משקל כבד יותר להיוריסטיקה, האלגוריתם רץ מהר יותר על חשבון אופטימליות הפתרון וככל שמשקל ההיוריסטיקה נמוך יותר כך אנחנו משפרים את אופטימליות הפתרון על חשבון זמן הריצה. הגרף שהתקבל תואם איכותית לסעיף 12 וכמובן מתיישב עם התיאוריה.

ניתן לראות שעד משקל של כ-0.71 מספר המצבים שפותחו יורד (שיפור בזמן הריצה) אך ללא פגיעה באופטימליות הפתרון (המרחק הכולל) עבור הבעיה הנ"ל. החל מנקודה זו הגדלת משקל ההיוריסטיקה פוגע משמעותית באופטימליות הפתרון.



לכל מצב  $s$  במרחב המצבים של `StrictDeliveryProblem` נבחר את  $h(s)$  להיות מחיר הצומת הסופי בהפעלת מימוש `RelaxedDeliveryProblem` עם היוריסטיקת `MSTAirDist` על אותה הבעיה. אם אין פתרון (למשל אין מספיק דלק) נחזיר  $h = \infty$ . בהפעלת `StrictDeliveryProblem` נבצע את השינויים הבאים: המצב ההתחלתי הוא  $s$ , כמות הדלק ההתחלתית היא הדלק הנותר ב- $s$  ורשימת ההזמנות היא הרשימה שנותרה ב- $s$ .

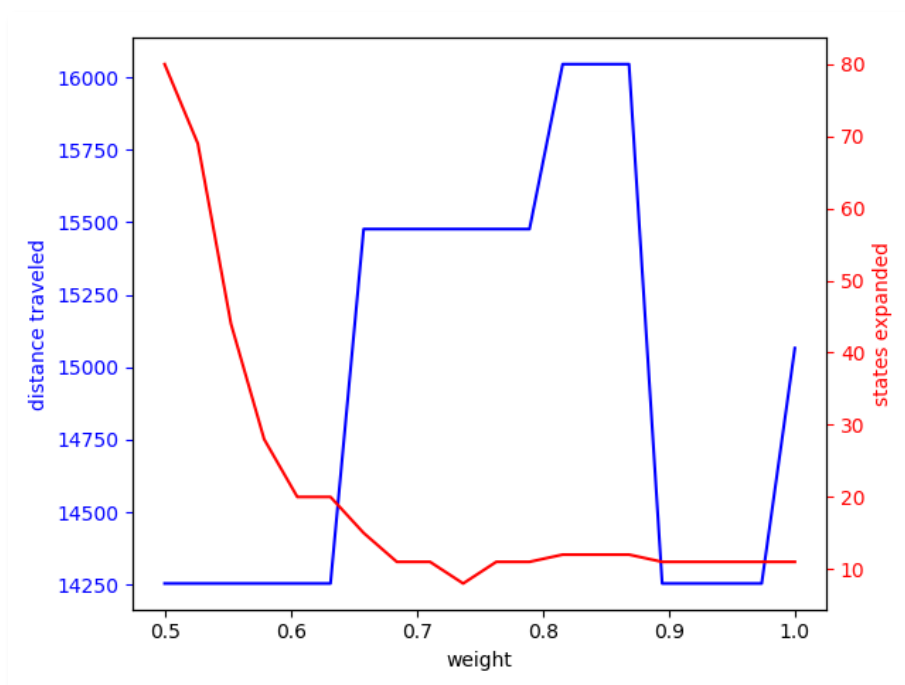
היוריסטיקה זו קבילה משום שראינו ש-`RelaxedDeliveryProblem` עם היוריסטיקת `MSTAirDist` היא קבילה – כלומר, הפעלתה מצומת מסוים יחזיר לנו מסלול אופטימלי למצב המטרה.

בעצם מובטח לנו שההערכה היוריסטית שלנו תחזיר עלות מדויקת כלומר קיבלנו את היוריסטיקה המושלמת  $h^*$  ומכיוון שלכל  $s$  מתקיים  $0 \leq h(s) = h^*(s) \leq h^*(s)$  אזי קיבלנו יוריסטיקה קבילה, שאגב היא המיועדת ביותר אשר עדיין קבילה.

בהשוואה לתוצאות מסעיף 26, ראשית נשים לב כי עבור משקל  $w=0.5$  (מימוש אלגור'  $A^*$  המקורי) שני האלגוריתמים מקבלים את אותו המחיר האופטימלי. תוצאה זו מחזקת את הטענה שלנו מסעיף 27 על פיה ההיוריסטיקה קבילה שכן אם היינו מקבלים מסלול ארוך יותר זה היה מראה שההיוריסטיקה לא קבילה. שנית, ניתן לראות שעם ההיוריסטיקה הנוכחית פיתחנו רק  $\sim 80$  צמתים לעומת  $\sim 120$  עם ההיוריסטיקה של סעיף 26 שכן זוהי היוריסטיקה מושלמת אשר מיועדת יותר מהיוריסטיקה של סעיף 26 ולכן "עושה פחות טעויות" מיוריסטיקה פחות מיועדת דבר הגורם לפיתוח מספר קטן של צמתים (פחות תיקונים של מסלולים אם בכלל).

לא היה משקל  $w$  עבורו מספר הפיתוחים בסעיף 26 היה נמוך ממספר הפיתוחים בסעיף 28

זמן הריצה השתפר. כידוע זמן הריצה מושפע ישירות ממספר הצמתים שפותחו. בסעיף 28, לכל משקל מספר הפיתוחים קטן משמעותית ולכן זמן הריצה השתפר כיוון שהיוריסטיקה מיועדת יותר.



## שאלה תיאורטית

א.

נתון כי  $h$  קבילה, מכאן שעבור כל מצב  $s$  בו  $h$  מוגדרת  $h_0(s) = h(s)$  ולכן  $h_0(s) \leq h^*(s)$ , עבור מצבים  $s$  בהם  $h$  אינה מוגדרת  $h_0(s) = 0 \leq h^*$  לכן  $h_0(s) \leq h^*$  לכל  $s$  ולכן קבילה.

ב.

נגדיר את היורסטיקה הבאה:

- אם  $Applicable_h(s) = h(s)$  אזי  $h_1(s) = h(s)$
- אם  $isGoal(s) = 0$  אזי  $h_1(s) = 0$
- אחרת נעבור על כל היורשים אשר להם מתקיים  $Applicable_h(succ)$  ונגדיר  $m$  להיות הערך המינימלי מבניהם.
- אם  $m$  מוגדר:
  - אם היורש הוא גם האב נגדיר  $h_1(s)$  להיות הערך היוריסטי של האב פחות מחיר הקשת שלו
  - או 0 אם יוצא שלילי
  - אחרת נגדיר  $h_1(s)$  להיות  $m$  בתוספת מחיר הקשת של הצאצא הרלוונטי
- אחרת:
  - $h_1(s) = 0$

ולמעשה נריץ עם יוריסטיקה זו  $A^*$  ונקבל את התוצאה הרצויה.

כיוון ש-  $h_1(s)$  נגזר מ-  $h(s)$  לכל צומת ואנו תמיד מוסיפים/מחסירים מחיר קשת בהתאם לכיוון המסלול אזי מובטח לנו שלא נפגע בקבילות.  
כלומר אם נניח בשלילה שקיבלנו ערך יוריסטי לא קביל עבור צומת כלשהו נקבל שהערך היוריסטי של האב/הצאצא אינו קביל בסתירה לכך ש-  $h$  קבילה.

ג.

נשתמש ביוריסטיקה של סעיף קודם בשינוי קל.  
במידה ונגיע לצומת שכבר ביקרנו בו נגדיר את הערך היוריסטי שלו להיות המקסימום מבין מה שהיה לערך הנגזר מהאב החדש שלו (כלומר נשתמש בפעולת החיסור ולא החיבור)

ד.

קיים אלגוריתם כנ"ל.  
נגדיר את היוריסטיקה הבאה:

$$h(s) = \begin{cases} h'(s) & s \text{ is root} \\ h(s, \text{father}) - \text{operator\_cost}(s, \text{father}, s) & \text{else} \end{cases}$$

נריץ  $A^*$  בעזרת היוריסטיקה  $h$ .

נוכיח:

- (1) היוריסטיקה קבילה ולכן האלגוריתם קביל
- (2)  $h$  מיועדת יותר מאשר  $h_0(h', s)$
- (3) האלגוריתם יעיל לפחות כמו  $A^*$  המשתמש ב-  $h_0(h', s)$

1.

לכל צומת  $s$  נוכיח באינדוקציה על מספר הקשתות מהשורש ל- $s$ .

בסיס:

עבור 0 קשתות מתקיים  $s = root$  ולכן  $h(root) = h'(root) = h^*(root) \leq h^*(root)$

צעד:

נניח ש- $h(t) \leq h^*(t)$  עבור צומת הנמצא במרחק  $k$  קשתות ונוכיח ש- $h(s) \leq h^*(s)$  עבור צומת הנמצא במרחק  $k + 1$  קשתות.

$$h(s) = h(s.father) - operatorCost(s.father, s) \leq h^*(s.father) - operatorCost(s.father, s) = h^*(s)$$

כאשר המעבר \* נובע מהנחת האינדוקציה.

2.

עבור  $s = root$  מתקיים  $h(s) = h_0(h', s) \geq h_0(h', s)$

עבור  $s \neq root$  נקבל  $h(s) = h(s.father) - operatorCost(s.father, s) > 0 = h_0(h', s)$

ולכן  $h$  הינה מיודעת יותר מ- $h_0(h', s)$  לכל  $s$

3.

לפי משפט מההרצאה אלגוריתם הרץ עם יריסטיקה מיודעת יותר יעיל לפחות כמו אלגוריתם הרץ עם יריסטיקה פחות מיודעת ולכן  $A^*$  עם  $h$  יעיל לפחות כמו  $A^*$  הרץ עם  $h_0(h', s)$