<u>תרגיל בית 1: שימוש באלגוריתמי חיפוש היוריסטיים לתכנון מסלולי חלוקה אופטימליים</u>

<u>מגישים:</u>

אריאל ווייס
תז: 204361315

דניאל בן חיון
תז: 308021054

חלק א' – מבוא והנחיות (3 נק' יבש + 3 נק' בונוס)

1. יבש (1 נק'):

ראשית נסדר את הדירות – !k , לאחר מכן יש m אפשרויות לכל מעבדה בין דירות + אפשרות ללא מעבדה- $(m+1)^k$ ונשארו m אפשרויות למעבדה האחרונה, סה"כ:

$$k! * (m+1)^k * m$$

2. יבש <u>בונוס</u> (3 נק'):

דומה לסידור הקודם, אך הפעם בין כל שתי דירות נחלק לשני מקרים- מקרה ראשון, לא עברנו במעבדה ולכך יש אפשרות אחת. מקרה שני, עברנו במעבדה שהתחילה רצף: כעת נבחר את מספר המעבדות (i) הנוספות שנעבור בהן ונסדר את המעבדות שנבחרו, ובנוסף נסדר את המעבדה שהתחילה את הרצף ברווחים בין המעבדות(2 אפשרויות, לשים או לא), כלומר:

$$\sum_{i}^{m} \binom{m}{i} * i! * 2^i$$

סה"כ:

$$k! * \left[1 + \sum_{i}^{m} \binom{m}{i} * i! * 2^i\right]^k * m$$

3. יבש (2 נק'):

k	m	*#possiblePaths*	Estimated calculation time
7	2	~22.04*10^6	~18.48 [secs]
7	3	~24.77*10^7	~3.85 [mins]
8	3	~79.27*10^8	~2.26 [hours]
8	4	~63.00*10^9	~19.56 [hours]
9	3	~28.54*10^10	~3.69 [days]
10	3	~11.42*10^12	~5.33 [months]
11	3	~50.23*10^13	~21.05 [years]
12	3	~24.11*10^15	~1.08 [thousand years]
12	4	~46.78*10^16	~22.41 [thousand years]
13	4	~30.41*10^18	~1.55 [million years]

חלק ג' – הגדרת מרחב החיפוש של בעיית מד"א (6 נק' יבש)

תרגילים

לטובת הסעיפים בחלק זה הנח שלאו דווקא קיים פתרון ישיג במרחב.

4. יבש (1 נק'):

במקרה המקסימלי, אנחנו נמצאים במצב ההתחלתי ונוכל להגיע לכל דירה או לכל מעבדה כלומר K + M

במקרה המינימלי, אנחנו במעבדה, ואם אנחנו במצב מטרה, אז הדרגה היא 0.

5. יבש (1 נק'):

לא ייתכנו- כי כדי שיווצר מעגל, נצטרך לבקר באותה דירה פעמיים או באותה מעבדה פעמיים:

לא נבקר באותה דירה, כי מהגדרת האופרטור נוכל לבקר בדירה רק אם $d_i \notin s.Taken$ ואחרי ביקור אחד, di מתווסף לTaken.

ביקור לשם לקיחת מטושים יקרה פעם אחת במעבדה. בביקור חוזר, הגענו עם Transferred מסוים, ובהכרח המספר ישתנה בהגעה השנייה למעבדה ולכן זהו לא אותו מצב אף פעם.

6. יבש (1 נק'):

מספר המצבים הוא אינסופי כיוון שיש אינסוף אפשרויות למספר המטושים (N טבעי). לא כל המצבים ישיגים, לדוגמא המצב בו מספר המטושים גדול יותר ממספר המטושים ההתחלתי + מספר המטושים הקיים בכל במעבדות.

7. יבש (1 נק'):

ייתכנו בורות ישיגים, לדוגמא המצב בו מספר המטושים הכולל נמוך ממספר הדיירים הכולל (כלומר נתקע בשלב מסויים ללא מטושים אך עם דירות שלא ביקרנו בהן)

8. יבש (1 נק'):

האורך המינימלי של מסלול: נניח שנתחיל עם מספר מטושים המספיק לבקר בכל הדירות, נשאר לבקר במעבדה האחרונה, לכן סה"כ: K+1

האורך המקסימלי: נניח שנתחיל בביקור בכל המעבדות ללקיחת מטושים ולאחר מכן נבקר בכל הדירות כשאחרי כל דירה נבקר במעבדה, לכן סה"כ: M + 2*K

9. יבש (1 נק'):

$$Succ_{MDA}(s) =$$

$$\left\{(di.loc, taken, transferd, matoshim, s.visited) \middle| \begin{matrix} CanVisit(s, di) \\ taken = s.taken \cup \{di\} \end{matrix} \right\} \cup$$

$$\left\{(li.loc, taken, transferd, matoshim, s.visited) \middle| \begin{matrix} li \in visited \\ s.taken \neq \emptyset \\ transferd = s.transferd \cup s.taken \end{matrix} \right\} \cup$$

$$\left\{(li.loc, taken, transferd, matoshim, s.visited) \middle| \begin{matrix} li \notin visited \\ visited = s.visited \cup \{li\} \\ matoshim = s.matoshim \cup li.matoshim \\ transferd = s.transferd \cup s.taken \end{matrix} \right\}$$

כש-

$$CanVisit(s, d_i) \triangleq \begin{bmatrix} d_i \notin s.Taken \cup s.Transferred \\ \wedge \\ d_i.roommates \leq s.Matoshim \\ \wedge \\ d_i.roommates \leq AmbulaceTestsCapacity - \sum_{d \in s.Taken} d.roommates \end{bmatrix}$$

חלק ד' – מתחילים לתכנת

תרגילים

10. רטוב

חלק ה' – אלגוריתם A* (2 נק' יבש)

תרגילים

11. רטוב
12. רטוב
13. רטוב

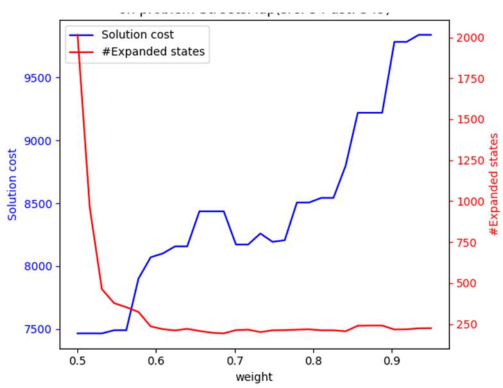
14. יבש (1 נק')

מספר מצבי פיתוח (nullhuristic) : 17354

מספר מצבי פיתוח (AirDistHeuristic) : 2015

החיסכון: 88 אחוז

15. רטוב
16. יבש (1 נק'):



<u>הסבר</u>: הגרף מתאר את מספר הצמתים שפותחו<אדום> ועלות הפתרון <כחול> (של ריצת A* עם AirDistHeuristic) כתלות בערכי w שונים בתחום [0.5,0.95].

<u>ערך w</u>: נרצה ערכי w עבורם נפתח כמה שפחות מצבים ונשלם כמה שפחות על הפתרון, מהגרף ניתן לראות כי נקודה זו מתקבלת באזור שבין 0.58~ לבין 0.52~, כלומר זה האזור העדיף בשבילנו.
נשים לב שהכלל אכן בא לידי ביטוי בגרף באופן כללי אך יש מקטעים בהם הוא אינו נכון, למשל בקטע 0.68~ עד 0.71~ שם יש ירידה בעלות הפתרון אך עלייה בערכי w.

חלק ו' – מימוש בעיית מד"א (15 נק' יבש)

17. רטוב
18. רטוב
19. שאלה יבש (2 נק'):

במרחב "העל" יהיו הרבה יותר מצבים אפשריים, מסלולים יתנפחו והcache יהיה מנופח בהתאמה. כיוון שאנחנו בודקים עלויות בcache בפתרון בעיית המרחק, cache גדול יותר אומר יותר זמן לחפש בו, ודבר זה יפגע לנו ביעילות זמן ריצה של הפתרון.

20. שאלה יבש (3 נק'):

(i)

```
21    @dataclass(frozen=True)
```

(ii)

שורה זו לא מספיקה -היא מונעת שינוי של שדות שניגשים אליהם by value אבל לא מונעת שינוי של שדות שניגשים אליהם by reference. אל הסטים ניגשים by reference ולכן הם מוגדרים להיות FrozenSet, וזהו הדבר הנוסף שמבטיח שלא יהיה ניתן לשנות בטעות את האובייקט.

(iii)

כן יכול להווצר מצב כזה. במקרה בו בעת פיתוח צומת נגלה מצב שכבר פיתחנו אשר ערך הפונקציית g שלו "זולה" יותר, ואז נכניס את הצומת לתור ה OPEN שוב למרות שכבר פיתחנו את המצב הזה , רק שהפעם נעדכן את ערכי g ו f לערכים החדשים שגילינו . זה קורה פה:

```
if old_node:                      ; A node with state s exists in CLOSED
    if new_g < old_node[g] :      ; New parent is better
        old_node[g] ← new_g
        old_node[parent] ← next
        old_node[f] ← old_node[g]+old_node[h]
        CLOSED ← CLOSED\{old_node}
        OPEN ← OPEN ∪ {old_node}; Move old node from CLOSED to OPEN
```

(iv)

הצורך בהגדרת MDAState כ- frozen נובע מזה שבעת פיתוח צמתים עוקבים, כאשר סורקים צומת מסוים לא נרצה לאפשר למצב העוקב שמפותח לשנות את הערכים של מצב האבא שלו, מכוון שכל צומת עוקב הוא חלק ממסלול שונה בסריקה. שינויים כאלה עלולים לגרום להגדרת מצבים שגויים ולשגיאות במהלך החיפוש. לדוגמא במתודה cost_with_state_expand של המחלקה MDAProblem :

המתודה מחזירה **צמתים עקבים** עבור צומת מסוים עם שימוש ב yield. כאשר מעצבעים פיתוח של עוקב הנוצר מביקור במעבדה L כלשהי (הפעלת אופרטור ביקור) ניתן לשנות ישירות את הרשימה visited_labs באופן הבא : להוסיף את המעבדה L לרשימה visited_lbas ואז לבצע השמה של המעבדה L ל visited_labs במצב החדש שפיתחנו.

```
state_to_expand.visited_labs.append(L)
new_state.visited_lab=L
```

כתוצאה מכך גם האבא וגם המצב העוקב חולקים מצביע לאותה רשימה visited_labs בעוד שהרשימה לא נכונה עבור האבא. בפעם הבאה שנפתח עוקב לאותו מצב השגיאה תגרום לפיתוח מצבים לא נכונים

עתה, כדי להריץ את A* על הבעיה, יש ראשית להגדיר (ולממש) היוריסטיקות עבור הבעיה.

21. רטוב
22. רטוב
23. יבש (4 נק'): הוכח/הפרך:

היוריסטיקה קבילה. הוכחה:

ע"פ הגדרה h יוריסטיקה קבילה אם מתקיים:

$$\forall s \in S : 0 \leq h(s) \leq h(s^*)$$

נסמן h . h'= MDAMaxAirDistHeuristic נותנת לכל צומת ערך שמציין את המרחק האווירי המקסימלי בין 2 נקודות בהן המסלול צריך לעבור (שעדיין לא עבר בהן). כלומר הערך מציין את המרחק המקסימלי בין 2 דירות שעדיין לא ביקרו בהן. נבחן את המקרה בו נשארה דירה אחת בלבד שעדיין לא ביקרו בה. מתקיים: h'(s)≤ 〚h'〛 ^* (s) . מאחר והמרחק האווירי הוא יוריסטיקה

קבילה (הוכח בהרצאה). לפי א"ש המשולש כל הוספה של דירה תאריך את המסלול ולכן מתקיים לכל s :

h'(s)≤h^('*) (s)

ולכן קיבלנו כי h' קבילה ע"פ הגדרה.

24. רטוב
25. רטוב
26. יבש (4 נק'): הוכח/הפרך:

לא קבילה. דוגמא נגדית: עבור k=3 ומיקומי הדירות הם : $(1,0), (3,0)$, $(-2,0)$ והמיקום ההתחלתי של האמבולנס הוא (0,0) (יש מספיק מטושים התחלתיים ובכל דירה דייר אחד) נקבל:

H(s)=sum((0,0)=>(1,0)=>(3,0)=>(-2,0))=1+2+5=8

אבל קיים מסלול קצר יותר:

H(s*)=sum((0,0)=>(-2,0)=>(1,0)=>(3,0))=2+3+2=7

כלומר מצאנו מסלול s* עבורו $H(s) > H(S^*)$. ולכן זו לא יוריסטיקה קבילה.

27. רטוב
28. רטוב
29. יבש (4 נק'): הוכח/הפרך:

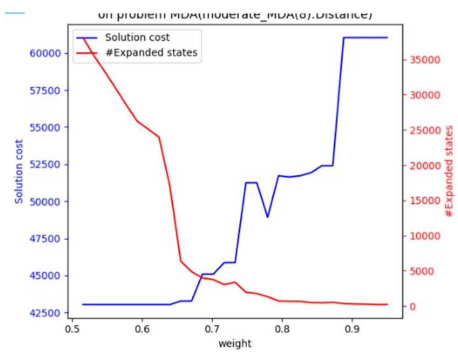
יוריסטיקה קבילה:

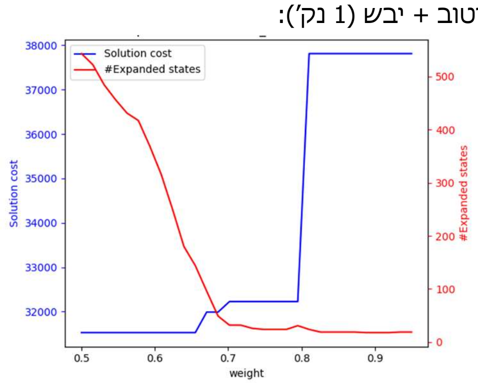
ע"פ הגדרת הפונקציה הערך של מצב s הוא סכום קשתות ה MST של גרף G , כאשר הצמתים בו מוגדרים להיות דירות שעדיין לא בוקרו בהם ממצב s. משקל כל קשת הוא המרחק האווירי בין שני צמתים. על מנת להגיע ממצב s למצב מטרה צריך לעבור בכל הצמתים בגרף G. , כלומר ניתן להסתכל על הבעיה של מסלול בין דירות כמסלול של עץ פורש בגרף G. נתון שהערך היוריסטי של s הוא ערך העץ הפורש המינימאלי של גרף זה , בהכרח כל מסלול אחר בין הדירות יהיה **לפחות** באותו המחיר. מחיר של כל קשת מוגדר לפי המרחק האווירי בין שתי דירות שהיא מחברת, זאת אומרת ש**מקבילות יוריסטיקת המרחק האווירי** ומכך שמדובר ב **עץ פורש** נקבל

$$\forall s \in S : 0 \le h(s) \le h(s^*)$$

ע"פ הגדרה קיבלנו ש MDAMSTAirDistHeuristic קבילה.

30. רטוב + יבש (1 נק'):





הסבר:

הגרפים כמו קודם מתארים מס' צמתים שפותחו ועלות הפתרון עם היוריסטיקה ASTAR עבור ערכי שונים בריצת האלג'.

ערכים כדאיים: אנחנו רוצים לשמור על פתרון קביל, לכן בשתי הבעיות נבחר w=~0.66 . אם נוכל לוותר על הקבילות, נפתח משמעותית פחות מצבים ונבחר ב
w=~0.7 for moderate
w=~0.67 for small

חלק ז' – מימוש והשוואת פונק' עלות שונות (21 נק' יבש)

31. יבש (2 נק'): סמן בכל אחד מהתאים כן/לא. האם כל אחת מההיוריסטיקות הנקובות הינה קבילה ביחס לפונק' המחיר הנקובה? (אין צורך בנימוק).

	MDAMaxAirDistHeuristic	MDASumAirDistHeuristic	MDAMSTAirDistHeuristic
$cost_{MDA}^{test\ travel}$	לא	לא	לא
$cost_{MDA}^{monetary}$	כן	לא	כן

32. רטוב + יבש (0.5 נק' יבש):

מס' נשווה את שתי סוגי הבעיות שבדקנו, mda small ו mda moderate, עם monetary cost לעומת distance cost:

MDA(small_MDA(5):Distance): total_cost: MDACost(dist= 31528.659m, money= 49.717NIS, tests-travel= 52112.429m)

MDA(moderate_MDA(8):Distance): total_cost: MDACost(dist= 43034.794m, money= 95.847NIS, tests-travel= 176505.013m)

MDA(small_MDA(5):Monetary): total_cost: MDACost(dist= 31923.809m, money= 42.050NIS, tests-travel= 53317.118m)

MDA(moderate_MDA(8):Monetary): total_cost: MDACost(dist= 54951.037m, money= 77.201NIS, tests-travel= 172922.318m)

SMALL: כשנלקח distance cost, קיבלנו מחיר: 49.717NIS

לעומת בmonetray cost, שקיבלנו: 42

MODERATE: במקרה עםdistance cost קיבלנו מחיר: 95.847NIS

עם monetary: 77

33. רטוב

34. יבש (2 נק'):

היוריסטיקה קבילה עבור testtravel cost:

ניקח מצב s כלשהו. נניח h*(s) = 0. אזי אין יותר דירות לבקר ולכן גם h(s) = 0.

אחרת, h*(s) > 0, נותרו דירות לביקור. לכל דירה k נסמן מרחק למעבדה הקרובה ביותר k_closet_lab ומרחק שהאמבולנס נסע בפועל בפתרון האופטימלי למעבדה k_real_lab.

כיוון שלכל דירה, k_closet_lab < k_real_lab, אז גם h(s) = sum(k_closet_lab) < sum(k_real_lab) = h*(s)

והדבר נכון לכל s, לכן היוריסטיקה אופטימית תמיד, כלומר קבילה.

35. רטוב + יבש (0.5 נק' יבש):

הפלט שיצא:

MDA(moderate_MDA(8):TestsTravelDistance):

total_cost: MDACost(dist= 93226.428m, money= 59.969NIS, tests-travel= 131265.153m)

ניתן לראות שהמרחק שהבדיקות עברו בפתרון זה: tests-travel= 131265.153m

קטן מהמרחק שהן עברו בפתרונות הקודמים: tests-travel= 176505.013m

שילוב בין 2 מדדים

36. יבש (3 נק'): הוכח/הפרך:

הוכחה:

נניח שקיים פתרון במרחב המקורי, ונסמן את מסלול הפתרון האופטימלי - $q=\langle s_0 \rightarrow s_1 \rightarrow \dots \rightarrow s_t \rangle$.

מסלול זה הוא מצב מקבל במרחב המסלולים, נסמן את המסלול המשוייך אליו

ב- $p = \langle s_0 \rangle \rightarrow \dots \rightarrow \langle s_0 \rightarrow s_1 \rightarrow \dots \rightarrow s_t \rangle$. כיוון שעלות המצב הכי יקר היא C_{dist}^* מתקבל שלכל צומת-

$$\sum_{i=0}^{t-1} cost_{MDA}^{dist}(s_i, o_i) \leq (1+\varepsilon) \cdot C_{dist}^*$$

לכן עבור העלות שקיבלנו כל צומת יקבל ערך קטן מאינסוף ויוכלו להפתח אם נרוץ מספיק זמן.

בסופו של דבר גם יפתח מצב מטרה, ולכן אם קיים פתרון במרחב המקורי האלגוריתם בהכרח יחזיר פתרון.

37. יבש (3 נק'): הוכח/הפרך:

הוכחה:

מהגדרת C_{dist}^*, יש מסלולים (לפחות אחד, נסמנו p) שכל צמתיו מקיימים את התנאי בגרף ש$cost_{MDA}^{dist}(p) \leq (1+\varepsilon) \cdot C_{dist}^*$ לכל אפסילון. מהגדרת פונ' המחיר, המחיר של מסלולים אלו

נמוך משל כאלה שאינם מקיימים תנאי זה, ובפרט לא ייתכן שנפתח מצב מטרה שאינו מקיים

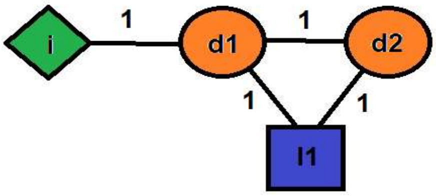
תנאי זה (בגלל מינימליות UCS). יתר על כן, מצב מטרה שנפתח הוא המצב המינימלי מכל אלו המקיימים את התנאי (מאותה סיבה) ולכן גם בעל ערך $cost_{MDA}^{test\ travel}$ המינימלי.
סה"כ מצב מטרה שנפתח עם האלגוריתם יהיה המינימלי תחת שתי פונ' העלות ולכן המסלול שיתקבל אופטימלי ע"פ הקריטריון המשולב

38. רטוב + יבש (0.5 נק'):
הפלט שהתקבל:
MDA(moderate_MDA(8):TestsTravelDistance):
total_cost: MDACost(dist= 65686.522m, money= 99.486NIS, tests-travel= 132209.981m)
לעומת סעיפים קודמים (מסודר בסדר עולה של dist):

	Test-travel	dist
A* , testtravel cost	131265.153m	43034.794m
A2	132209.981m	65686.522m
A*, dist cost	176505.013m	93226.428m

ניתן לראות ש A2 אכן מאזן בין המדדים(נמצא בין שניהם), אך מוותר לשם כך על הdist האופטימלי.
חישוב הערך הנ"ל: 0.6 ~ 0.53 = 1- (65686.522/43034.794) , ערך ה-epsilon אכן נשמר!

39. יבש (4 נק'): הוכח/הפרך:
הפרכה:
דוגמא נגדית-



ניקח גרף כזה: i=(0) , d1=(1), d2=(2), l1=(3) , עם קשתות המחברות וממושקלות כמו בציור.
נניח שיש מספיק מטושים כדי לא לדאוג, בכל דירה דייר יחיד. נניח שהאמבולנס בנק התחלה.
אחרי ריצת ASTAR, נקבל שהמרחק האופטימלי הוא 3 (i->d1->d2->l1) = C_{dist}^*
ניקח $\varepsilon = 0.1$. (צמתים עם מרחק גדול מ 1.1*3 לא ניקח)
נראה ריצה של הUCS על הגרף: [ייצוג צומת- (v:(g_testtravel,g_dist)]

Iter.	1	2	3	4	5
Next	i	D1	L1	D2	----
Open	{i}	{d1:(1,0)}	{d2:(2,1),l1:(2,1)}	{d1:(0,3),d2:(0,3)}	{l1:(4,1),d1:(4,1)}
Close	{}	{i}	{I,d1}	{I,d1,l1}	{I,d1,l1}

נשים לב ש'נתקענו' באיטרציה החמישית כיוון שאין צמתים העונים על תנאי הdist. לכן אין פתרון

40. יבש (4 נק'):
הוכחה:
לפי אופן פעולת האלגוריתם, הוא מחפש פתרון לפי פונ' העלות $cost_{MDA}^{test\ travel}$, לכן אם מוחזר פתרון הוא יהיה אופטימלי ביחס לפונ' זו(מאופטימליות UCS). בנוסף, הפתרון מקיים שלכל צומת העלות שלו קטנה מ$(1+\varepsilon)\cdot C_{dist}^*$,(אחרת היה נמחק ולא מפותח),ומאופטימליות C_{dist}^* נקבל כי הפתרון המוחזר אופטימלי ע"פ הקריטריון המשולב.

41. יבש (1.5 נק'):
$\mathcal{A}_1$ פשוט רץ על מרחב הרבה יותר גדול מ$\mathcal{A}_2$(מרחב המסלולים של $\mathcal{S}_{MDA}$), דבר המייקר את זמן הריצה (למשל במעבר על מס' הצמתים לפיתוח): במקרה גרוע מרחב המסלולים הוא בגודל 2 בחזקת המרחב הרגיל (קבוצת החזקה).

חלק ח' – מימוש האלג' $A^*\varepsilon$ והרצתו (1 נק' יבש)

42. רטוב
43. רטוב
44. יבש (1 נק'):

MDA(small_MDA(5):Distance)   A* (h=MDA-MST-AirDist, w=0.500)   time: 0.60 #dev: 543 |space|: 877   total_g_cost: 31528.65909 total_cost: MDACost(dist= 31528.659m, money= 49.717NIS, tests-travel= 52112.429m) |path|: 8

MDA(small_MDA(5):Distance)   A*eps (h=MDA-MST-AirDist, w=0.500) time: 38.12 #dev: 492 |space|: 821   total_g_cost: 31528.65909 total_cost: MDACost(dist= 31528.659m, money= 49.717NIS, tests-travel= 52112.429m) |path|: 8

חסכנו בפיתוחים : ניתן לראות כי חסכנו 51 פיתוחים בריצה עם A*eps. השוני בין האלגוריתמים הוא בבחירת הצומת הבא לפיתוח.
רשימת ה focal מחזיקה צמתים שערך ה f שלהם מינימלי עד כדי כפל ב 1 + eps כלומר ערך ה f שלהם אינו המינימלי, אבל הוא טוב מספיק בשביל לבחון אפשרות שמחיר המסלול שנפתח ממנו יהיה זול. מתוך הרשימה הזאת, בוחרים את הצומת בעלת הערך היוריסטי הנמוך ביותר, כלומר הצומת שמוערך על ידי h כצומת הכי קרוב למטרה.
מכיוון ש f מושפעת מ g ומ- h, גם אם f מקבלת ערך מינימאלי מכל הצמתים בתור ה OPEN, לא מובטח שאכן זה הצומת שיתן את מסלול האופטימלי, למשל יכול להיות מצב שהיוריסטיקה עבור צומת זה הייתה אופטימית ועבור צמת אחר הייתה מחמירה. מכיוון ש**כן** נותנים משקל להיוריסטיקה אכן מגבילים את f של הצמתים ב OPEN להיות מינימאלית (עד כדי 1+ epx), וכך גם השתמשנו במידע שהיוריסטיקה נותנת לנו. בנוסף, אפשרנו גמישות כדי לכפר על מקרים שהיא לא מדויקת. העקרון מאחורי זה הוא שנותנים הזדמנות גם לצמתים שלא נראו מבטיחים בהתחלה אבל כן הובילו למסלולים אופטימליים, להיות מפותחים בשלב מוקדם באיטרציות (מישהו רק היה צריך להאמין בהם)

חלק ט' – מימוש האלג' Anytime A* והרצתו

45. רטוב
46. רטוב

חלק י' – שאלה תאורטית (12 נק' יבש)

סעיף (א) – 1 נק' יבש

IDA* תופס פחות זכרון בזמן הריצה. ליתר דיוק O(bd) כאשר b הוא branching factor ו d הוא עומק העץ.
בעוד שבA* צריכת הזכרון פרופורצינלית למס הצמתים שנוצרו.

סעיף (ב) – 5 נק' יבש

(i) (1 נק' יבש)

זמן ריצה

(ii) (2 נק' יבש)

כי IDA* לא שומר מבנה נתונים של צמתים שכבר ביקרו בהם , ולכן יכול להיווצר מצב שאלגוריתם מבקר שוב ושוב באותה צומת מבלי לדעת שביקר בו בעבר.

(iii) (2 נק' יבש)

לא. ראשית מכיוון ש IDDFS לא קביל בעוד ש BFS כן קביל. לעומת A* וגם IDA* ששניהם קבילים. בנוסף IDDFS מגדיל את עומק החיפוש כל פעם ב 1 מה שלא מובטח ב IDA*.

בהמשך השאלה נבחן וריאציה לאלג' IDA* שמטרתה להתמודד עם הבעיה עליה נשאלתם בסעיף ב', תוך הקרבה של איכות הפתרון.

סעיף (ג) – 6 נק' יבש

(i) (3 נק')

תשובה: $K * Q_k(C_s^*) + 1 - Q_k(h(i))$
הסבר: כל עוד מוצאים צמתי עם ערך f קטן מ f-limit , ממשיכים לפתח את הצמתים התואמים.אחרת , מגדילים את f-limit ב $\frac{1}{k}$ כל עוד אין פתרון , לכן בסה"כ נקבל כי הערך המקסימלי של f-limit הוא $K * Q_k(C_s^*) + 1$

(ii) (3 נק')

חסם עליון כנ"ל הוא $\frac{1}{k}$ כיוון שכל מסלול נבדק בקפיצה של $\frac{1}{k}$ ולכן ערך אפסילון המקסימלי הוא $\frac{1}{k}$