מבוא לבינה מלאכותית - 236501

תרגיל בית 1

מרחבי חיפוש

מגישים:
מתן סולומון 209339894
מיכאל בלום 209291681

מבוא ורקע

התרגיל מתפרש על פני מסמך זה והמחברת המצורפת. מומלץ לענות על השאלות לפי הסדר במסמך זה.

במטלה זו נעסוק בהפעלת אלגוריתמי חיפוש על מרחבי מצבים לבעיות ניווט. מומלץ לחזור על שקפי ההרצאות והתרגולים הרלוונטיים לפני תחילת העבודה על התרגיל.

סיפור מסגרת

לקאקרוטו וגוהאן יש 5 כדורי דרקון וחסר להם שני כדורים, והם ממש צריכים אותם כדי להזמין הדרקון שן-ראן ולבקש ממנו להחזיר את החברים שלהם לחיים, לכן הם הלכו לכוכב לכת נאמיק כדי לחפש כדורי הדרקון, קאקרוטו הציע שיחפשו על הכדור דרך ה ג״י.פי.אס שלהם אבל גוהאן מסביר לקאקרוטו שיש לו חברים שלוקחים הסמסטר את קורס ״מבוא לבינה מלאכותית״. גוהאן מבקש ממכם לעזור לו לתכנן את המסלול הטוב ביותר כדי לאסוף כדורי הדרקון ולהגיע לקאקרוטו שמחקה לו.

שאלה 1 – מבוא (8 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת:

1. **רטוב**: עברו על המחברת עד שאתם מגיעים לחלק של BFS-G ועצרו שם.

2. יבש (1 נק׳): תחילה נרצה להגדיר את מרחב החיפוש כפי שנלמד בתרגול. הגדר את $(S, 0, I, G)$ עבור סביבת כדורי הדרקון. כאשר S זה מרחב המצבים, O, זה מרחב האופרטורים, I, זה המצב ההתחלתי וG הוא קבוצת מצבי המטרה. מה גודל מרחב המצבים S? הסבירו.

S – יהיה הלוח שלנו עם המיקום של גוקו עליו ואם הוא אסף את הכדור הראשון ואם הוא אסף את הכדור השני, בלוח יש 64 משבצות שבהן גוקו יכול להימצא ולכן מספר המצבים שלנו הוא: $64 \cdot 2 \cdot 2 = 256$.
O – ארבעת האופרטורים הם UP, DOWN, RIGHT, LEFT.
I – המצב ההתחלתי תמיד יהיה הנקודה S- (0,False,False).
G - כל משבצת בלוח שמסומנת ב-G ושגוקו אסף את שני הכדורים, כלומר: $\forall g \in G\ (g, True, True)$.

3. יבש (1 נק׳): מה תחזיר לנו הפונקציה Domain על אופרטור 2 (UP)?
הפונקציה תכיל את כל המצבים מלבד המצבים שבהם המשבצת מסומנת ב-G או ב- H, וזה כי האופרטור מוגדר על כל מצב שניתן לצאת ממנו כאשר אם המשבצת בשורה הראשונה ומעליה אין עוד משבצות גוקו ישאר במקום. המצבים ב-G הם סופיים ולא ניתן לצאת מהם, והמצבים שהם H מוגדרים גם הם כמצבים שלא ניתן לצאת מהם.

4. יבש (1 נק׳): מה תחזיר לנו הפונקציה Succ על המצב ההתחלתי 0?
הפונקציה תחזיר לנו את כל המצבים אליהן ניתן להגיע מהמצב ההתחלתי וכמה יעלה להגיע לשם:
נקבל את המצבים : (0, False, False) – אם ננסה ללכת למעלה או שמאלה נישאר באותו מקום, ויעלה לנו 1
(8, False, False) – אם נלך למטה ויעלה לנו 10
(1, False, False) – אם נלך ימינה ויעלה לנו 10

5. יבש (1 נק׳): האם קיימים מעגלים במרחב החיפוש שלנו?
כן, ניתן לבקר במשבצת שכבר היית ולכן ייתכנו מעגלים במרחב החיפוש, נשים לב כי במסלול האופטימלי לא יהיה מעגלים כי אין משבצות שהמעבר אליהן (קשתות) עם ערך שלילי.
נשים לב כי לאחר שאפסנו כדור נוסף כל משבצת נוספת שנגיע אליה היא מצב חדש.

6. יבש (1 נק׳): מה הוא מקדם הסיעוף בבעיה?
מקדם הסיעוף הוא 4 מכיוון שניתן להגיע מכל משבצת ל4 משבצות סמוכות (לפעמים לפחות).

7. יבש (1 נק׳): במקרה הגרוע ביותר, כמה פעולות ידרשו לסוכן כללי להגיע למצב הסופי?
במקרה הגרוע הוא לעולם לא יגיע, הוא יכול ללכת רק למעלה מהמצב ההתחלתי ולא לזוז בכלל או ליפול לבור ולא לצאת ממנו.

8. יבש (1 נק׳): במקרה הטוב ביותר, כמה פעולות ידרשו לסוכן כללי להגיע למצב הסופי?
14, עבור לוח כללי במקרה הטוב ביותר מספר הפעולות הוא מרחק מנהטן לנקודת G הקרובה ביותר ל-0 שעוברת דרך הצמתים עם הכדורים, בהנחה שיש דרך מינימלית שאינה עוברת בבור ולכן נקבל מסלול של 14 צעדים.

בהנחה שמדברים על הלוח שנתון למעלה הפתרון הקשר ביותר הוא: Down*6,Right*3,Up,Right*2,Down,Right*2,Down

סך הכל נקבל 16 צעדים.

9. יבש (1 נק׳): עבור לוח כללי, המסלול הקל ביותר הוא המסלול שמגיע למצב מטרה שהכי קרוב למצב ההתחלתי (במונחים של manhattan distance)? אם כן, הוכיחו. אם לא, ספקו דוגמא נגדית.
הטענה לא נכונה, וזה מכיוון שיש משבצות זולות יותר ממשבצות אחרות.
לדוגמה לוח 3x3: המסלול שמגיע למטרה הקרובה ביותר הוא הכחול ומשקלו לפחות 12.
לעומת זאת המסלול הירוק מגיע למטרה רחוקה יותר אך משקלו רק 4.

S	D,D	L
F	F	L
F	G	G

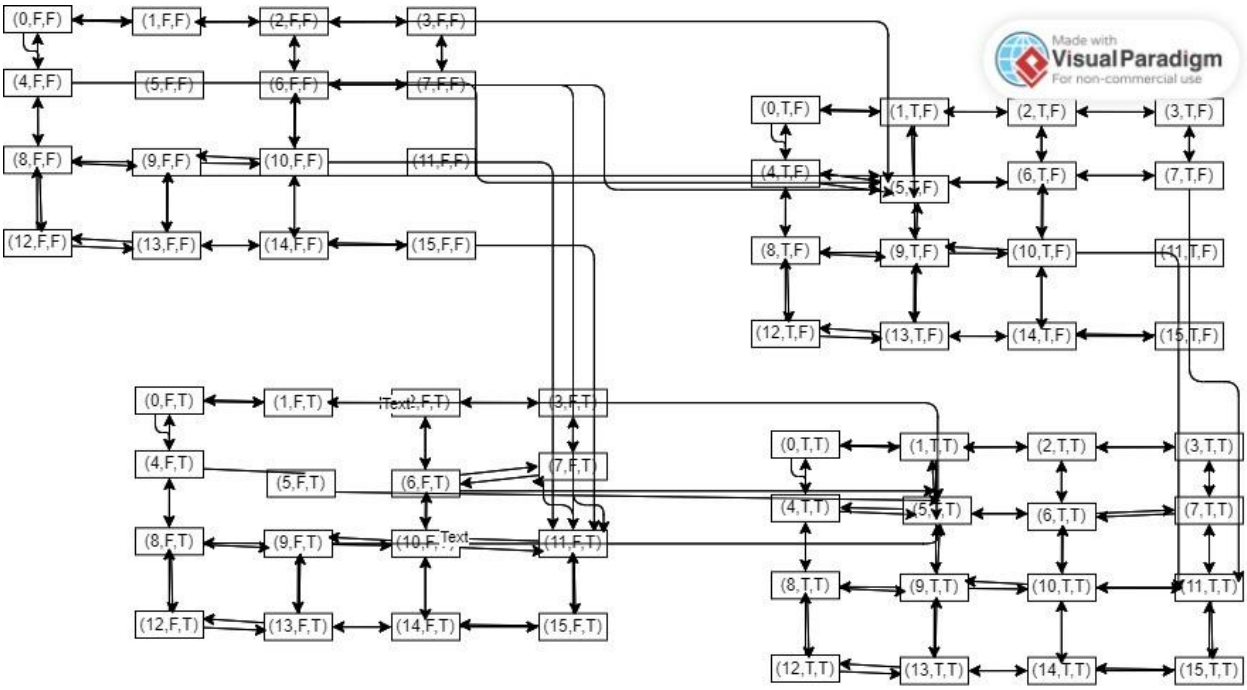
שאלה 2 – Breadth First Search-G (7 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

1. **רטוב**: ממשו את אלג׳ BFS-G (על גרף) במחברת ע״פ ההנחיות המופיעות שם.

2. יבש (1 נק׳): מה צריך להיות התנאי על גרף החיפוש (לא בהכרח בבעיית כדורי הדרקון) כך שBFS על גרף ו-BFS על עץ ייצרו ויפתחו צמתים זהים באותו הסדר?
על מנת שBFS על גרף ועל עץ ייצרו ויפתחו מצבים / צמתים זהים באותו הסדר – עלינו לדאוג למצב שבו אין מעגלים. כלומר שבגרף אין חזרה למצב שכבר הופיע. במקרה זה, האלגוריתם על הגרף לא יפתח צומת זה (לפי ההגדרה) ואילו האלגוריתם על העץ, כן. ופה ישתבש סדר הפיתוח. כמו כן עלינו לדאוג שהסידור של צמדי צומת בעץ ומצב בגרף, מזינים את האלגוריתם באותו האופן. (אם קיים סידור שונה של מצבים בגרף מאשר בעץ – זה בהכרח יפגע בסדר הפיתוח).

3. יבש (2 נק׳): עבור הלוח "4x4" שמופיע במחברת, ציירו את גרף המצבים.



4. יבש (2 נק׳): נתון לוח בגודל NxN. הציעו דרך להשתמש באלגוריתם BFS-G כך שיחזיר פתרון אופטימלי (עלות מינימלית) והסבירו.
- רמז: עליכם לספק פונקציה $T: G \to G'$ המקבלת את גרף המצבים G ויוצרת גרף חדש G' ובעזרתה למצוא את המסלול האופטימלי בגרף G.

BFS-G מפתח את כל הבנים של צומת בשכבה, מכניס לopen וכך הלאה. המטרה שלנו היא שהוא יעשה זאת לפי משקלים הצמתים.
נפתור את הבעיה באופן הבא:
הפונקציה T תבצע את הפעולה הבאה:
בגרף `G, הפונקציה תחליף כל מצב בשרוך שכמות המצבים בו שווה לעלות הכניסה לתא במצב המקורי. עבור מצב שהתא שלו הוא חור, היא תסיר מצב זה מהגרף.
וכך, כאשר BFS-G יפתח כל מצב, הוא יפתח קודם לכן מצבים שעלותם נמוכה יותר (כי הם יהיו פחות עמוקים), כיוון שכעת, עומק כל מצב בגרף יהיה שקול לעלות שלו.
נוכיח את הטענה. יהיה פתרון אופטימלי (הפתרון שעלותו מינימלית). מעצם היותו בעל עלות מינימלית, ומהגדרת הגרף `G, כל מצב בו יוחלף בשרוך בעל כמות מצבים שתואמת לעלותו. כיוון שעלותו מינימלית, אזי מצב המטרה בו הוא בעל העומק הקטן ביותר שניתן להגיע אליו. ולכן BFS-G יגיע אליו ראשון ויחזיר אותו ראשון.

נניח בשלילה שBfs-g החזיר פתרון שאינו אופטימלי. מאופן פעולת bfs-g עומק המסלול אל מצב המטרה בגרף G` היה הקטן ביותר, ולכן bfs-g הגיע אליו ראשון (למצב המטרה, במסלול שהוחזר). לכן, סכום עלויות המצבים בפתרון שbfs-g החזיר היה מינימלי, כי כל מצב בG מוחלף בשרוך בעל כמות מצבים שתואמת לעלותו. אם סכום העלויות היה הקטן ביותר, אזי מדובר בפתרון "הזול" ביותר כלומר המינימלי. ולכן מדובר בפתרון האופטימלי, בסתירה.

5. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל $N^2 - 2$ משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה. כמה צמתים <u>יפותחו</u> <u>וייווצרו</u> במהלך חיפוש BFS-G? הסבירו?

ניצור N2 מצבים ונפתח N2-2 מצבים.
הסבר: נבחין כי ישנן N2-2 משבצות רגילות, וכיוון שעוד 2 משבצות הינן משבצות ההתחלה והסיום (s,g) אין כדורים על גבי הלוח.
לכן, יש לנו סה"כ N2 מצבים (כי אין בוליאני עבור אחזקה בכדור). המרחק ממצב ההתחלה למצב המטרה הינו N2 (מנהטן), וזהו המרחק הגדול ביותר בין משבצת כלשהי למשבצת ההתחלה בלוח. כיוון שאלגוריתם הBFS-G לא יפתח אותו מצב פעמיים, וכיוון שהוא מפתח ב"שכבות", כלומר טרם פיתוח מצב בעומק עמוק יותר, הוא יפתח את כל המצבים בעומק רדוד יותר. וכך, נפתח את כל המצבים עד לעומק N-22. (מדובר על N2-3 מצבים), לאחר מכן, אנחנו נפתח מצב בודד בעומק N-12 (הבן של המצב הזה הוא מצב המטרה). בשלב זה, ניצור את בנו- מצב בעומק N2, נבדוק אם הבן של מצב זה הוא מצב המטרה, והוא אכן מצב המטרה. הגענו ליעד, ולא נפתח צמתים נוספים. סה"כ יצרנו את כל N2 המצבים בלוח, ופיתחנו N2-2 מצבים.

שאלה 3 – Depth First Search-G (6 נק׳):

1. יבש (1 נק׳): עבור בעיית כדורי הדרקון עם לוח NxN, האם האלגוריתם שלם? האם הוא קביל?
האלגוריתם שלם אך לא קביל וזה מכיוון שהוא יכול להחזיר מסלול ארוך יותר, כפי שראינו בהרצאה.
דוגמה על לוח 3X3:
הפתרון האופטימלי הוא $0 \to 1 \to 2 \to 5 \to 8$, כלומר מסלול באורך 4
לעומת זאת *DFS* יעבור ב$0 \to 3 \to 6 \to 7 \to 8 \to 4 \to \cdots$ ארוך יותר מ4.

0	1	B-2
3	4	B-5
6	7	G-8

2. יבש (1 נק׳): האם אלגוריתם DFS (על עץ), עבור בעיית כדורי הדרקון על לוח NxN, היה מוצא פתרון כלשהו? אם כן, מה המסלול שיתקבל? אם לא, כיצד האלגוריתם היה פועל?
האלגוריתם לא היה מוצא, מכיוון שהוא רץ על עץ הוא לא בודק אם הוא כבר היה בצומת לכן יתכן שיחזור על צומת מספר פעמים. מהגדרות הסביבה נובע כי כאשר מנסים לצאת מחוץ לגבולות הלוח זה כאילו נכנסנו למשבצת שאנו עומדים עליה, ולכן גוקו ילך לכיוון מטה עד הגעתו לקצה התחתון של הלוח וימשיך לנסות לרדת וישאר במקום.
אם במקרה קיימת נקודת סיום בדרך בה גוקו הולך (ואין נקודת בור והכדורים נמצאים בדרך זו) האלגוריתם יצליח, אבל גוקו יצטרך הרבה מזל.

3. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל $N^2 - 2$ משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה (תניחו כי שני כדורי הדרקון הם בפינה ימינית תחתונה) . כמה צמתים <u>יפותחו</u> <u>וייווצרו</u> במהלך חיפוש DFS-G? הסבירו?

מספר הצמתים שיווצרו הוא 4N-4 ושיפותחו הוא 2N-2, וזה מכיוון שגוקו ירד עד למטה (N) ואז כאשר יגיע לקצה ילך ימינה עד המטרה (N-1 צמתים), בכל צעד הוא ייפתח צומת אחד וייצור 2 צמתים.
אנחנו מניחים כי אופן בחירת הכיוון של האלגוריתם הוא למטה, ימינה, למעלה, שמאלה.

4. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל $N^2 - 2$ משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה (תניחו כי שני כדורי הדרקון הם בפינה ימינית תחתונה). כמה צמתים <u>יפותחו</u> <u>וייווצרו</u> במהלך חיפוש backtracking DFS-G? הסבירו?

זהה לסעיף קודם רק שבאלגוריתם זה אנחנו מייצרים מצבים באופן אצל כלומר ברגע שייצרנו מצב אנחנו נלך עליו ונייצר את המצב הבא רק לאחר שנחזור, במקרה שלנו המסלול הראשון של האלגוריתם מוצא את הפתרון ולכן לא ייוצרו צמתים מחוץ למסלול שהוא למטה עד הסוף (N) ואז ימינה עד הסוף (N-1).
מספר הצמתים שייווצרו הוא 2N-1 ושיפותחו הוא 2N-2, כי אין צורך לפתח את צומת המטרה.

שאלה 4 – ID-DFS (6 נק׳):

.1

a. (1 נק׳) האם האלגוריתם שלם? אם כן, הוכיחו. אם לא, ספקו דוגמה נגדית.

כן. האלגוריתם שלם. אנחנו עוסקים בבעיות להן קיים פתרון סופי כלשהו ומקדם סיעוף סופי (אם לא קיים פתרון או מקדם הסיעוף אינסוף, האלגוריתם לא ידע להחזיר דבר). מאופן פעולת האלגוריתם, בכל שלב (איטרציה) הוא מפתח את כל הצמתים עד לעומק מסוים (לא בשכבות כמו BFS, אבל במידה והוא לא מוצא פתרון, הוא בפועל יפתח את כל השכבה). כאשר הוא יגיע לעומק בו נמצא הפתרון, הוא יגיע אליו ויחזיר אותו.

b. (1 נק׳) נניח כי עלות כל פעולה היא 1, האם האלגוריתם קביל? אם כן, הוכיחו. אם לא, הסבירו.

כן! כאשר לקשתות מחיר אחיד, המסלול הקצר ביותר הוא גם המסלול האופטימלי. יהיה d אורך פתרון אופטימלי. אזי האלגוריתם ימצא אותו באיטרציה ה-d. אם קיים פתרון שימצא באיטרציה קודמת, אזי הוא באורך קטן מd, בסתירה לנתון לגבי אורך הפתרון האופטימלי. אם הוא יחזיר פתרון באורך ארוך יותר, אזי הוא לא מצא את הפתרון באורך d, בסתירה לכך שהאלגוריתם באיטרציה ה-dית יעבור על כל הצמתים בעומק קטן שווה לd.

.2 הניחו כי יש לנו ידע מקדים על חסם עליון למרחק למצב מטרה, נסמנן D. בת (Beth) הציעה את האלגוריתם חיפוש הבא:

```
function ReverseDFS (problem, D):
        L ← D
        result ← failure
        While Not Interrupted:
                new_result ← DFS-L (problem, L)
                if new_result = failure:
                        break
                L ← L - 1
                result ← new_result

        return result
```

.3 בשאלות הבאות הניחו כי יש מספיק זמן לסיום האיטרציה הראשונה.

a. (1 נק׳) ספקו דוגמה בה ReverseDFS עדיף על ID-DFS ודוגמה בה ID-DFS עדיף על ReverseDFS. הדוגמאות יכולות להיות כלליות ולא בהכרח מסביבת התרגיל.

דוגמה בה ReverseDFS עדיף על ID-DFS- מקרה בו מצב המטרה נמצא בעומק D (החסם העליון הוא גם הפתרון האופטימלי) ו- D>2. במקרה זה ReverseDFS ירוץ פעם אחת לעומק D, ימצא פתרון, פעם נוספת לעומק D-1 לא ימצא פתרון ויחזור. לעומת זאת Id-DFS ינסה את כל העומקים עד לעומק D, וימצא אותו רק בעומק D. כלומר ההבדל יהיה ש-IDDFS יריץ בנוסף את DFS-L על כל העומקים: d-2,d-3,…

מקרה בו ID-DFS עדיף הוא כאשר החסם העליון הוא מאוד מאוד גס והפתרון הרבה יותר קרוב להתחלה מאשר לסוף (אנחנו לא נעשה אריתמטיקה מדויקת של מתי כל מצב עדיף בהנתן מקדם סיעוף כלשהו) כלומר בהנתן וD>>1. והפתרון הוא בעומק 2, Id-DFS ימצא אותו תוך 2 איטרציות, בעוד שReverseDfs ינסה את כל העומקים עד שיגיע ל2.

b. (2 נק׳) הציעו כיצד ניתן לייעל את האלגוריתם. רמז: האם אתם יכולים לחשוב על צעד עדכון עדיף לL?

בהחלט! נבקש מdfs-L להחזיר לנו לא רק את new_result (כלומר האם מצאנו פתרון), אלא גם באיזה עומק הפתרון שהוא מצא. ואז נעדכן את L להיות Solution_depth-1. לא מובטח לנו שהוא בהכרח ימצא פתרון קצר יותר בכל ריצה (יכול להיות ביש מזל והוא יחזיר כל פעם פתרון ארוך ביותר). אבל בהנתן האלגוריתם הנ"ל, הוא כבר כן יכול למצוא פתרון קצר בהרבה (ואולי אפילו את הפתרון האופטימלי), ואז לא נעשה הרבה מאוד ריצות מיותרות. כלומר אנחנו נפיק יותר מידע וחיסכון מכל ריצה שעשינו, כי בתרחיש הנוכחי המידע נאבד, גם אם הוא מוצא את הפתרון האופטימלי כל פעם, הוא ימשיך להריץ שוב ושוב ושוב על פתרונות באורכים פחות טובים.

או פתרון נוסף – לנסות להפחית את L לא ב1, אלא בחצי, ואז כמו במיון בינארי – אם לא מצאנו פתרון בעלות של L/2, אז נחפש פתרון באמצע (כלומר בעלות של L*3/4, ואז החיפוש יהיה לוגריתמי בL).

שאלה 6 - UCS (4 נק׳):

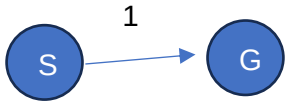
השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

1. יבש (1 נק׳): עבור אילו בעיות חיפוש אלגוריתם UCS ואלגוריתם BFS יפעלו באותו האופן? הסבירו.
במידה ומשקל כל הקשתות שווה האלגוריתמים יהיו זהים, מכיוון שUCS בוחר את הקשת הקלה ביותר בכל פעם ומכיוון שכל הקשתות באותו גודל ניתן לגרום לו לבחור כמו BFS.

2. יבש (1 נק׳): האם בבעיית החיפוש שלנו, עבור לוח NxN, האלגוריתם הוא שלם? האם הוא קביל?
האלגוריתם שלם וקביל וזאת מכיוון שהוא ידע לא להיכנס לבורות כי העלות שלהם היא אינסוף, בנוסף משקל הקשתות חסום מלמטה על ידי 1 ולכן כפי שראינו האלגוריתם תמיד ימצא את הפתרון האופטימלי.

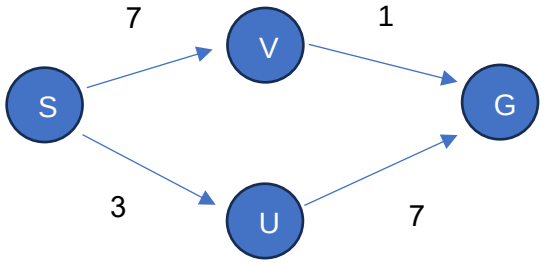
3. יבש (2 נק׳): שאדי טעה במימוש של אלגוריתם UCS ובטעות בדק בעת יצירת הצומת האם היא צומת מטרה במקום בפיתוח שלה. הביאו דוגמה לגרף חיפוש שעבורו שאדי יחזיר בכל זאת את המסלול הקל ביותר, ודוגמה לגרף חיפוש שעבורו שאדי לא יחזיר את המסלול הקל ביותר. עבור כל דוגמה הסבירו מה המסלול והעלות ש-UCS השגוי החזיר, ומה המסלול והעלות שהאלגוריתם הנכון היה מחזיר. נדגיש שגרף החיפוש לא בהכרח צריך לייצג את בעיית כדור הדרקון. אתם יכולים לתת דוגמה לגרף שמייצג בעיית חיפוש אחרת. הגרף צריך להכיל קשתות מכוונות ואת העלות של כל קשת.

האלגוריתם של שאדי יחזיר תשובה נכונה על הגרף הפשוט הבא שבו יש רק פתרון אחד והוא 1:



לעומת זאת בגרף חיפוש הבא האלגוריתם של שאדי יטעה ויחזיר את המסלול S-U-G במשקל 10

במקום את המסלול S-V-G במשקל 8.



שאלה 7 - יוריסטיקות (8 נק׳):

יהי מרחב חיפוש (S,O,I,G) , נסתכל על בעיית הניווט לכדור דרקון יחיד. . המטרה היא למצוא מסלול זול ביותר מהמוצא I ליעד יחיד G . פונק׳ העלות מוגדרת כאורך הכביש המחבר בין שתי נקודות. ניתן להניח כי העולם שטוח . מלבד זאת, לא ניתן להניח דבר נוסף על מרחב החיפוש.

הגדרה : יוריסטיקה h היא ε-קבילה אם קיים $\varepsilon \geq 1$ כך שלכל מצב $s \in S$ מתקיים $h(s) \leq \varepsilon \times h^*(s)$.
נזכיר כי $h^*(s)$ הינה פונקציית המחיר המסלול האופטימאלי מ-s לצומת היעד .

עבור כל אחת מהיוריסטיקות הבאות קבעו האם קיים $\varepsilon \geq 1$ כך שהיוריסטיקה תהיה ε-קבילה . אם כן מצאו את ה-ε ההדוק ביותר המקיים את זאת. נמקו היטב .

1. יבש (1 נק׳): מרחק מנהטן : $h_{MD}(p) = |P - G|_1 = |G_x - P_x| + |G_y - P_y|$

קיים, מכיוון שאורך הכביש הקצר ביותר בין שתי נקודות הוא קו ישר כלומר $\sqrt{(G_x - P_x)^2 + (G_y - P_y)^2}$ ועבור $\epsilon = \sqrt{2}$ נקבל

כי לכל צומת $h(p) = |G_x - P_x| + |G_y - P_y| \leq \sqrt{2} \cdot \sqrt{(G_x - P_x)^2 + (G_y - P_y)^2} = h^*(p)$

2. יבש (1 נק׳): $h(p) = |P - G| = \min\{G_x - P_x \ , \ G_y - P_y\}$

קיים והוא שווה ל1.

מתכונות המשולש נקבל כי המרחק בין שתי נקודות גדול גם מהפרש הx וגם מהפרש הy ולכן גדול מהמינימום ביניהם.

3. יבש (1 נק׳): L^3 : $h(p) = |P - G|_3 = \sqrt[3]{|G_x - P_x|^3 + |G_y - P_y|^3}$

כן, כמו בסעיף קודם עבור $\epsilon = 1$ נקבל שהיוריסטיקה היא ϵ -קבילה, מכיוון שלכל שני מספרים x,y מתקיים כי

$$h(p) = \sqrt[3]{x^3 + y^3} \leq \sqrt{x^2 + y^2} = h^*(p)$$

4. יבש (1 נק׳): נתונות יוריסטיקות h_1 , h_2 שהן ε_1 , ε_2 קבילות בהתאמה וכי ε_1 , ε_2 הם האפסילונים ההדוקים ביותר.
הראו כי $h_3 = h_1 + h_2$ היא ε_3 – קבילה , מצאו את ε_3 ההדוק ביותר והוכיחו .

מתקיים לכל $s \in S$: $h_1(s) \leq \epsilon_1 h^*(s), \ \ h_2(s) \leq \epsilon_2 h^*(s)$

$$h_3(s) = h_1(s) + h_2(s) \leq \epsilon_1 h^*(s) + \epsilon_2 h^*(s) = (\epsilon_1 + \epsilon_2) h^*(s)$$

בגלל שϵ_1 הוא החסם ההדוק ביותר על h_1 וכנ"ל עבור h_2 נקבל כי החסם ההדוק ביותר על h_3 הוא $\epsilon_3 = \epsilon_1 + \epsilon_2$.

נגדיר יוריסטיקה חדשה :

- D היא קבוצת כדורי הדרקון , $D = \{d1, d2\}$.

$$h_{MSAP}(s) = \min\{h_{Manhatan}(s, g) | g \in G \cup D\}$$

הערה: בנוסחת המרחק מתייחסים למיקום של צומת.
שימו לב שבמקרה זה אנחנו לוקחים את המינימום על פני כל צמתי היעד.

5. יבש (1 נק׳): האם היוריסטיקה h_{MSAP} קבילה על כל לוח? אם כן הסבר, אם לא הבא דוגמה נגדית.

היוריסטיקה קבילה על כל לוח, וזה כי בעולם שלנו בו לא ניתן ללכת באלכסון מרחק מנהטן הוא המסלול המינימלי מבין כל המסלולים בין כל שתי נקודות על הלוח, כלומר $h_{Manhatan}(s) \leq h^*(s)$. היוריסטיקה שבסעיף לוקחת את המינימום מבין מרחקי מנהטן לנקודה הסופית ולכל כדור ולכן בהכרח מקיים $h_{MSAP} \leq h_{Manhatan}(s) \leq h^*(s)$ והיוריסטיקה קבילה.

6. יבש (1 נק׳): האם היוריסטיקה h_{MSAP} עקבית על כל לוח? אם כן הסבר, אם לא הבא דוגמה נגדית.(לחשוב אם היא עקבית ולתקן בהתאם)

היוריסטיקה אכן עקבית,
וזאת מכיוון שעלות כל צעד בכיוון כלשהו על הלוח היא לפחות 1 כלומר משקל כל קשת היא לפחות 1.
מבחינת מרחקי מנהטן צעד בכיוון כלשהו יכול לשנות את המרחק במקסימום 1 ולכן משקל היוריסטיקה בין שני צמתים סמוכים הוא לכל היותר 1, מכך נקבל כי בין כל שני צמתים שיש ביניהם קשת – צמתים סמוכים מתקיים
$|h_{MSAP}(s_1) - h_{MSAP}(s_2)| \leq 1 \leq e = (s_1, s_2)$.

נגדיר יוריסטיקה חדשה :

- D היא קבוצת כדורי הדרקון , $D = \{d1, d2\}$.

$$h_{new}(s) = \max\{h_{Manhatan}(s, g) | g \in G \cup D\}$$

7. יבש (1 נק׳): האם היוריסטיקה h_{new} קבילה על כל לוח? אם כן הסבר, אם לא הבא קודמה נגדית.
היוריסטיקה אינה קבילה.
דוגמה נגדית:
נניח כי אנו נמצאים בנקודה המסומנת בסגול (מספר 14) וכבר אספנו את שני הכדורים, המצב הוא (14,T,T)
ניתן להגיע למצב הסופי על ידי צעד יחיד בכיוון ימינה שמשקלו הוא 1 ולכן $h^*(s) = 1$
לעומת זאת היוריסטיקה שלנו מחזירה את מרחק מנהטן הגדול ביותר שבמקרה זה הוא 4 ולכן נקבל $h_{new}(s) > h^*(s)$.

S	F	F	F
D	F	F	F
F	F	F	F
D	F	F	G

8. יבש (1 נק׳): האם היוריסטיקה h_{new} עקבית על כל לוח? אם כן הסבר, אם לא הבא דוגמה נגדית.

היוריסטיקה אינה עקבית וזאת מכיוון שהראינו שהיא לא קבילה בסעיף קודם.

שאלה 8 – Greedy Best First Search (3 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

1. יבש (1 נק׳): האם האלגוריתם שלם? האם הוא קביל?
האלגוריתם שלם, כי הלוח סופי, וכמות המצבים סופית, ולכן בסוף האלגוריתם יעבור על כל המצבים האפשריים ויחזיר פתרון (וקיים פתרון, כי ישנו מסלול כלשהו שלוקח את הכדורים ומגיע למצב המטרה, ללא קשר ליוריסטיקה במקרה זה).
לגבי הקבילות, האלגוריתם לא קביל. ראשית, נבחין כי הפתרון האופטימלי הוא: 6 צעדים למטה, 3 צעדים ימינה, צעד למעלה, 2 צעדים ימינה, צעד למטה, 2 צעדים ימינה וצעד למטה. Greedy, יבחר בדיוק באותו מסלול עד הקטע המודגש. ושם- כיוון שהיוריסטיקה מייחסת אותו משקל לתא 61 (שורה 7 עמודה 6) ו62 כמו לתא 54 ו55 (כפי לפי היוריסטיקה hsmap יש להם אכן משקל זהה). היא תעדיף ללכת למטה (פשוט כי צעד למטה הוא קודם בניסיונות לצעד ימינה). ונקבל פתרון שאינו אופטימלי (T ו – L עולים פחות מפעמיים F) ולכן greedy במקרה זה יהיה לא קביל.

2. יבש (2 נק׳): תנו יתרון וחיסרון של אלגוריתם Greedy Best first Search לעומת Beam Search.

יתרון של greedy best:

למרות שGBFS אינו קביל ואינו בהכרח מחזיר את המסלול האופטימלי במצבים בהם יש לנו הרבה צמתים ב-open Beam Search יזרוק חלק מהצמתים (הוא מחזיק מקסימום k צמתים בכל רגע נתון), ובכל עלול לאבד פתרונות טובים יותר. ולכן במצבים אלו GBFS ימצא מסלולים טובים יותר.

חשוב להדגיש כי GBFS אינו קביל ולכן אינו בהכרח מחזיר את המסלול האופטימלי אבל הוא כנראה יחזיר מסלול טוב יותר מ-Beam Search.

חסרון:

במידה ויש המון מצבים בעלי יוריסטיקה זהה, אנחנו נחזיק המון מהם בזיכרון, וכמו כן אנחנו גם עלולים לפתח אותם, זה עלול לעלות לנו בזמן וזיכרון, בעוד שייתכן ומה שאותם מצבים מציעים לנו הוא "more of the same" ואין טעם לפתח את כולם.

במצבים שכאלה, greedy יבזבז לנו זמן וזיכרון, בעוד שbeam search יחזיק לכל היותר k מצבים בתור הopen שלו, ובכך יאיץ את החיפוש (לא יעבור על מצבים שלא תורמים דבר).

נבחין כי היתרונות של כל אחד באים לידי ביטוי במצבים ספציפיים, לא ניתן לטעון בכללי כי אחד עדיף על השני.

שאלה 9 – W-A* (2 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת.

1. **רטוב**: ממשו את החלקים החסרים באלג׳ W-A* בקובץ ע״פ ההנחיות המופיעות שם. עליכם להשתמש ביוריסטיקה h_{MSAP}.

2. (יבש 2 נק׳) בהינתן $w_1 < w_2 \leq 1$, נסמן את המסלולים המחוזרים על ידי W-A* תחת הפורמולציה $f = g + w \cdot h$ בp_1, p_2 עבור w_1, w_2 בהתאמה. אזי $cost(p_1) < cost(p_2)$ עבור:

   a. יוריסטיקה קבילה h. אם כן הסבירו. אם לא, ספקו דוגמה נגדית.

   לא בהכרח, לדוגמה ניקח את h להיות יוריסטיקת האפס שהיא קבילה, ואז נקבל $f = g$ ושני המסלולים שיוחזרו יהיה בעלי אותו משקל ולכן $cost(p_1) = cost(p_2)$.

   b. יוריסטיקה כללית (לא בהכרח קבילה) h. אם כן הסבירו. אם לא, ספקו דוגמה נגדית.

   לא, כמו בדוגמה הקודמת, ניקח את יוריסטיקת ה0.

שאלה 10 – IDA* (2 נק׳):

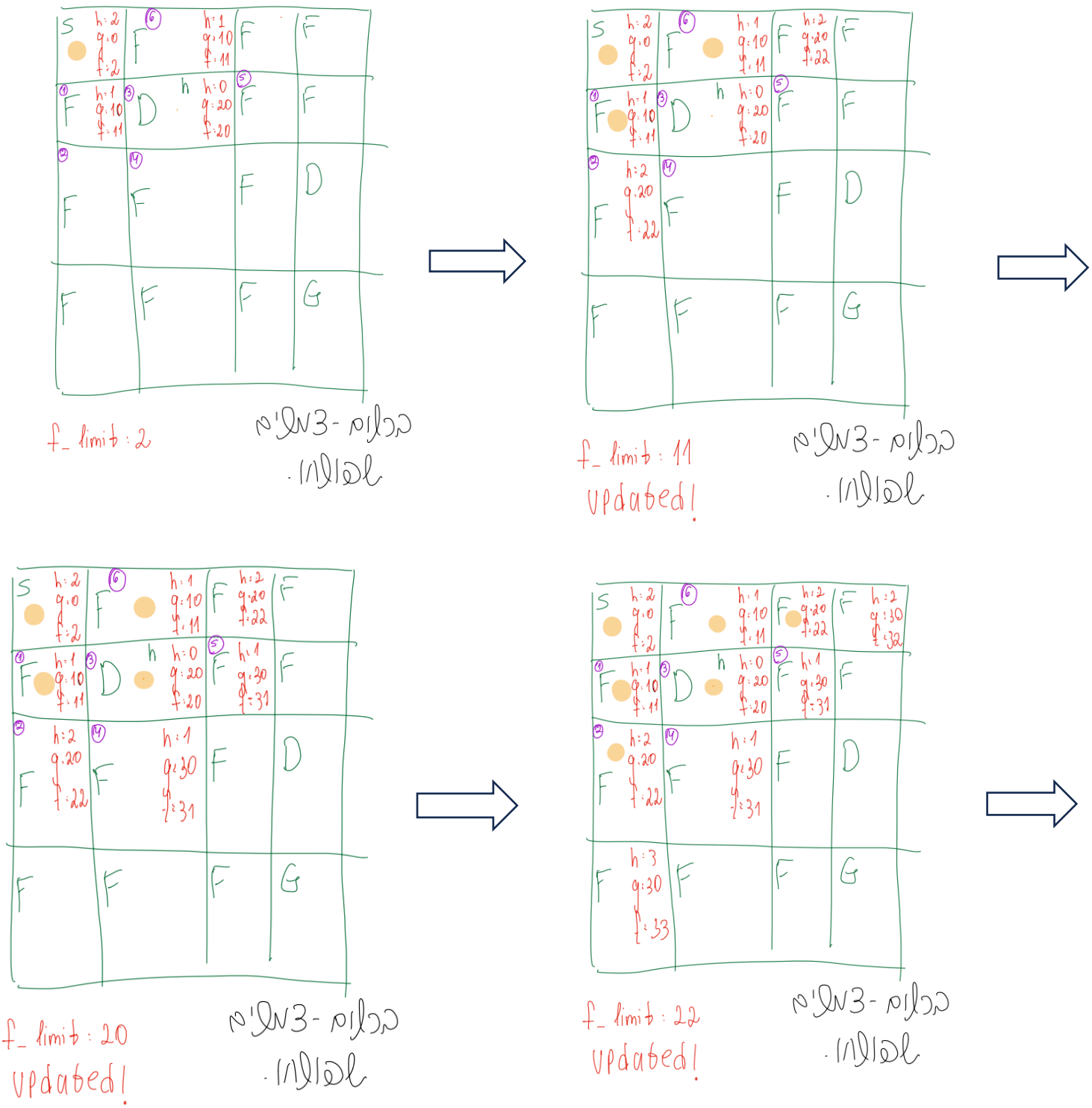
1. יבש (1 נק׳): ספקו יתרון וחסרון של IDA* ביחס ל*A. באילו מקרים הייתם מעדיפים להשתמש בכל אחד מהם?

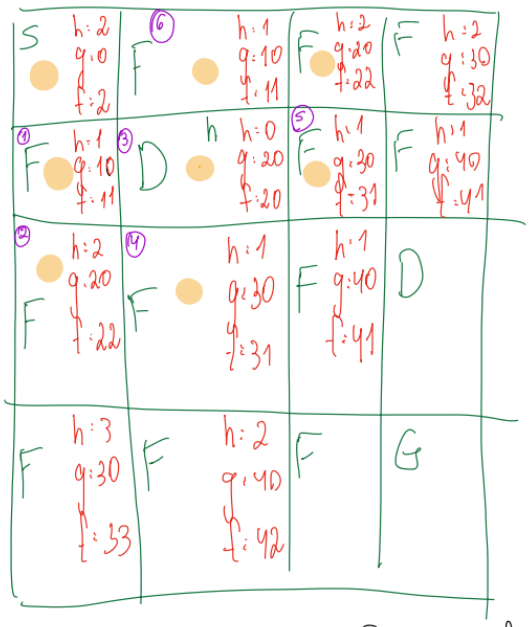
החידוד החשוב: *IDA הוא dfs שבעצם מתבסס על g+h=f, ולא רק על משקלים או נטו על ללכת לעומק. בניגוד לA* שבכל איטרציה מפתח את כל הבנים של צומת מסוים. שים לב – A* יצור את כל הצמתים אבל יפתח רק את האחד עם ערך הf האופטימלי מבחינתו. *IDA פשוט קודח לעומק עד שהוא נתקל בבעיה, כאשר המגבלה שלו בניגוד לIDfs היא לא העומק אלא הf, הוא לא מנסה לאפטם את ערך הf או לבחור את הקטן ביותר, אלא משתמש בהם כקו מנחה בלבד.

יתרון: *ida מפתח כמות צמתים פרופורציונלית באורך המסלול, לעומת *A שמפתח כמות צמתים פרופורציונלית בכמות הצמתים בopen&close. במידה ויש לנו הרבה מאוד צמתים עם f זהה (ששווה ל f_limit לדוגמא), *A עלול לפתח את כולם! בעוד ש*ida בהיותו אלגוריתם מבוסס DFS יתקדם לעומק. וכך, *A עלול לפתח כמות צמתים ענקית, בעוד ש*ida יפתח מסלול אחד בלבד. מובטח לנו שאם היוריסטיקה קבילה, 2 האלגוריתמים קבילים.

חסרון: במידה ויש לנו מצב שכל פעם אנחנו נתקלים בערך f ייחודי, אנחנו נאלץ כל פעם לבצע ריצה, לגלות שהf_limit הנוכחי "תוקע" אותנו, ולנסות שוב עם f גדול יותר, וניתן לגרום למצב שכל פעם נפתח צומת אחד נוסף בלבד. אבל! בכל איטרציה אנחנו יוצרים ומפתחים צמתים מ0, גם צמתים שכבר יצרנו ופיתחנו באיטרציות קודמות, וכך *IDA עלול לפתח אותם צמתים פעמים רבות מאוד, בניגוד לA* שיעשה זאת פעם אחת בלבד.

2. יבש (1 נק'): ספק המחשה שלב אחר שלב של אלגוריתם IDA* על הלוח (4x4) שמופיע במחברת, המראה כיצד החיפוש מתקדם באמצעות העמקה איטרטיבית ?



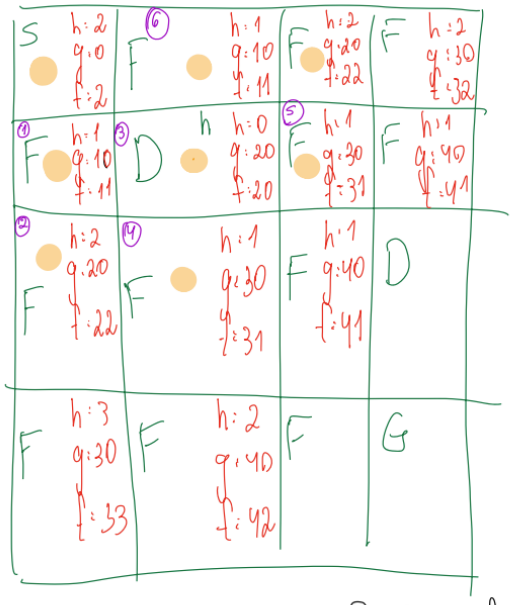


f_limit: 31

updated!

בכלים - 3 מלי'ם

לפעולה.

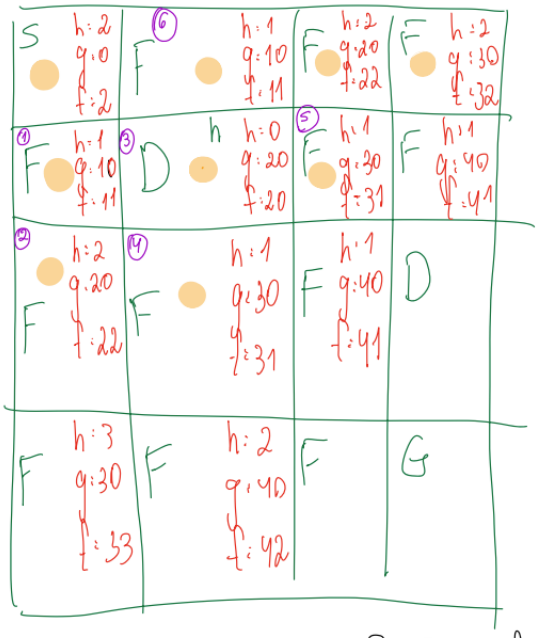


f_limit: 31

updated!

בכלים - 3 מלי'ם

לפעולה.

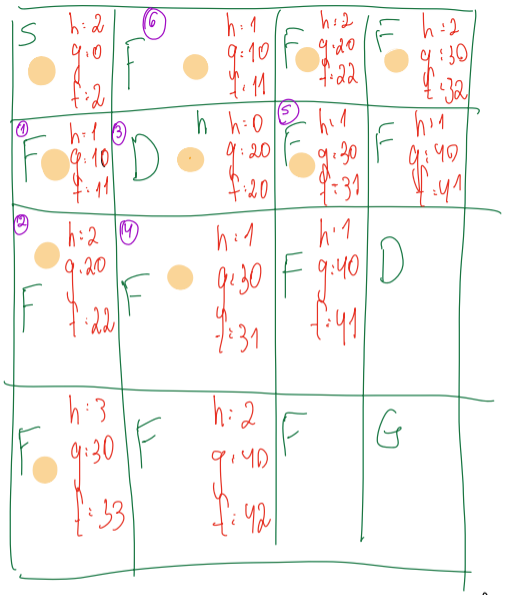


f_limit: 32

updated!

בכלים - 3 מלי'ם

לפעולה.

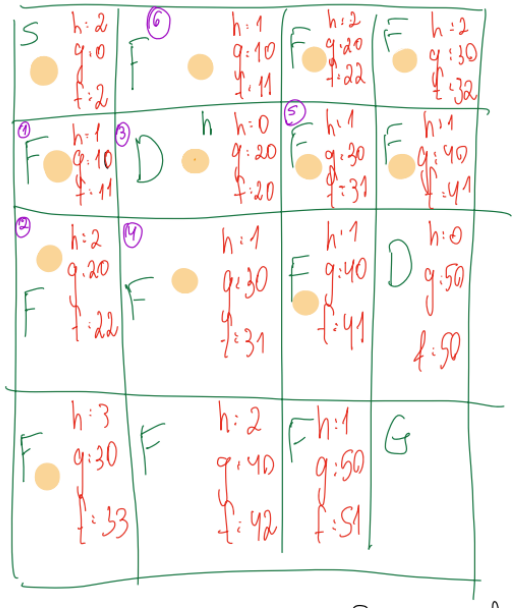


f_limit: 33

updated!

בכלים - 3 מלי'ם

לפעולה.

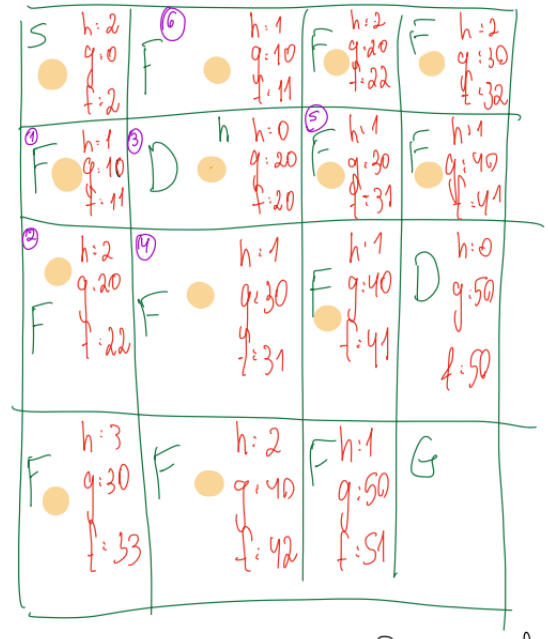


f_limit: 41

updated!

בכיוון - 3 מלמ'

לפולמו.

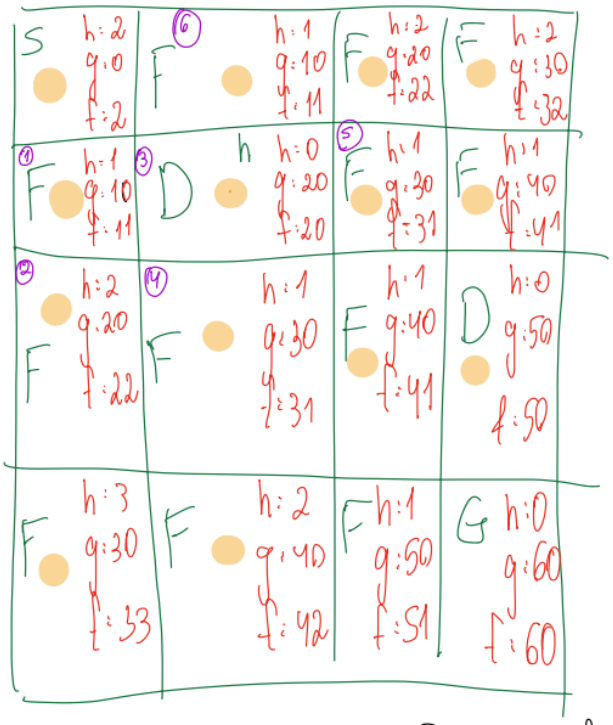


f_limit: 42

updated!

בכיוון - 3 מלמ'

לפולמו.

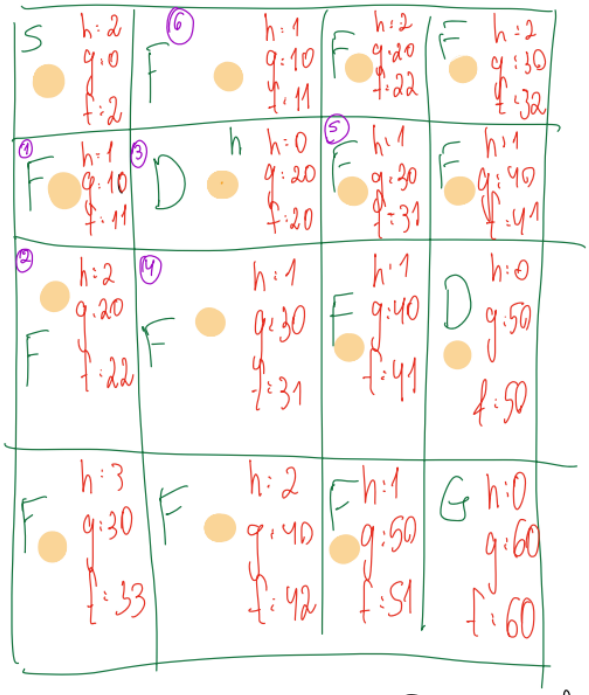


f_limit: 50

updated!

בכיוון - 3 מלמ'

לפולמו.



f_limit: 51

updated!

בכיוון - 3 מלמ'

לפולמו.

S h:2 g:0 f:2	6 F h:1 g:10 f:11	F h:2 g:20 f:22	F h:2 g:30 f:32
1 F h:1 g:10 f:11	3 D h h:0 g:20 f:20	5 F h:1 g:30 f:31	F h:1 g:40 f:41
2 F h:2 g:20 f:22	4 F h:1 g:30 f:31	F h:1 g:40 f:41	D h:0 g:50 f:50
F h:3 g:30 f:33	F h:2 g:40 f:42	F h:1 g:50 f:51	G h:0 g:60 f:60

f_limit: 60

updated!

בכלום - 3 שלבים
לפעולה.

שאלה 11 – A* epsilon (6 נק׳):

1. **רטוב**: ממשו את החלקים החסרים באלג׳ W-A* בקובץ ע״פ ההנחיות המופיעות שם. עליכם להשתמש ביוריסטיקה h_{MSAP}.

2. יבש (2 נק׳): תנו יתרון וחיסרון של A*-epsilon לעומת A*.

יתרון: A*-epsilon מהיר יותר כי במציאת המסלול למטרה וזה כי הוא מאפשר הליכה במסלולים שאינם אופטימליים אך יכולים לקרב מאוד אל המטרה .
חסרון: A*-epsilon עלול להחזיר מסלול שאינו אופטימלי.

3. יבש (3 נק׳): תנו הצעה ליוריסטיקה כדי לבחור את הצומת הבאה לפיתוח מתוך FOCAL. תארו את היוריסטיקה והציגו השוואה בין השימוש ביוריסטיקה זו לעומת השימוש ב-$g(v)$, מבחינת מספר פיתוחים, מסלול שנבחר ועלות המסלול שנבחר.

היוריסטיקה שבחרנו היא היוריסטיקה הבאה:

$$\boldsymbol{h(v) = \frac{1}{index(v) + 1}}$$

הגיון היוריסטיקה הוא שהיא מתעדפת משבצות עם ערכים גדולים יותר, ובהינתן שהמטרה הסופית במשבצת עם האינדקס הכי גדול היוריסטיקה תקרב אותנו אל המטרה.

ביוריסטיקה הנ"ל קיבלנו את התוצאות הבאות:
מסלול:
{ 0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 1, 1, 0, 1, 0, 1 }
עלות המסלול: 112
מס' צמתים שפותחו: 85.

נבחין שהיוריסטיקה שבחרנו, מפתחת יותר צמתים, ומחזירה לנו פתרון פחות טוב. הסיבה לכך היא כפי שאמרנו היוריסטיקה שלנו מתעדפת צמתים עם אינדקס גדול יותר, כלומר היא תעדיף ללכת "למטה" אל עבר שורות נמוכות יותר, כי הדבר מגדיל את האינדקס בצורה הגדולה ביותר.
נבחין כי בצעד הלפני אחרון שלה, היוריסטיקה בחרה ללכת למטה. הדבר מייקר את הפתרון, כי צעד ימינה הוא זול יותר, אבל כפי שאמרנו, היוריסטיקה הנ"ל מתעדפת צמתים עם אינדקס גדול יותר.
אבחנה מעניינת נוספת שנתקלנו בה, היא שכשאנחנו מגדילים את אפסילון (לערך של 6), היוריסטיקה בוחרת ללכת למטה בצורה עיוורת (גם ב2 צעדים לפני), דבר זה מקטין בצורה דרמטית את מס' הצמתים המפותחים (ל21).
נוכל להסביר זאת באופן הבא: ברגע שהגדלנו מאוד את אפסילון, היוריסטיקה מסתכלת בfocal על מס' גדול של צמתים שהיו לנו בopen. וכך, בגלל אופי הלוח, היא מהר מאוד מוצאת מסלול שרץ כמה שיותר ימינה ולמטה. במקרה הזה, זה גם לא רחוק מהפתרון האופטימלי, והיא מוצאת אותו די מהר, הדבר חוסך לה פיתוח של צמתים אחרים. אם היינו במצב בו מצב המטרה הוא בפינה השמאלית העליונה, היא הייתה מפתחת הרבה יותר צמתים.
לסיכום, היוריסטיקה החדשה שאנחנו מגדירים מאפשרת לנו לתעדף פתרונות שהם לא "גרועים מדי" לפי היוריסטיקה הקודמת, והיא בוחרת בצומת שהוא הכי טוב לפיה. במקרה והיא תתעדף פתרונות שקרובים לפתרון, זה יהיה טוב, אך היא עלולה לגרום להעדפה של פתרונות שרחוקים מאוד מהפתרון האופטימלי.

4. יבש (1נק׳): אם נגדיר שאפסילון שווה לאינסוף איך תהיה ההתנהגות של האלגוריתם עם סביבת כדורי הדרקון.

האלגוריתם יתנהג זהה לזה של A^* וזה מכיוון שכל שכן עומד בדרישה של להיות קטן מ– $\epsilon h^*(s)$ ולכן נוסיף אותו לרשימת FOCAL, מלבד מצבים שכבר היינו בהם ואז אין צורך.

שאלה 12 – Benchmarking (2 נק׳):

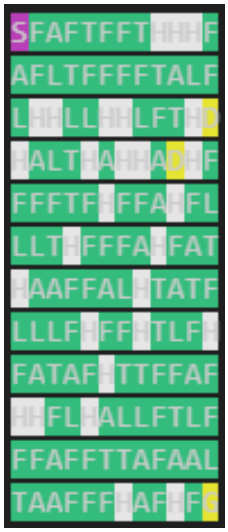
בשאלה זאת נשווה בין אלגוריתמי חיפוש שונים על בעיות שונות. הריצו את החלק הרלוונטי במחברת ותיראו שנוצר קובץ csv. (ניתן לפתוח עם Excel).

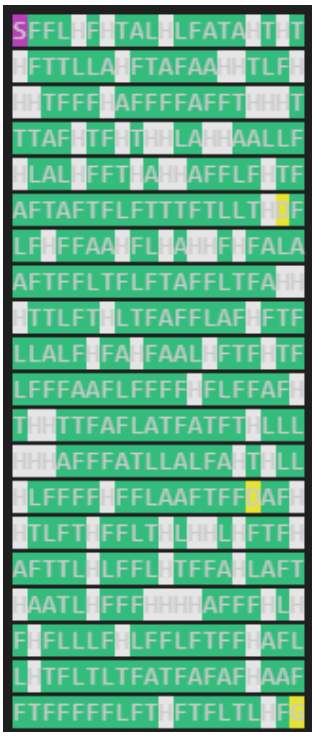
1. **רטוב**: הריצו את החלק הרלוונטי במחברת ותיראו שנוצר קובץ csv. (ניתן לפתוח עם Excel).

התוצאות שקיבלנו:

map	BFS-G cos	BFS-G exp	WA* (0.5)	WA* (0.5)	WA* (0.7)	WA* (0.7)	WA* (0.9)	WA* (0.9) expanded
map12x12	140	445	118	224	118	200	118	240
map15x15	215	858	178	651	178	604	195	707
map20x20	203	1045	188	684	188	587	188	1002

נדפיס גם את הלוחות על מנת להבין יותר טוב את התוצאות





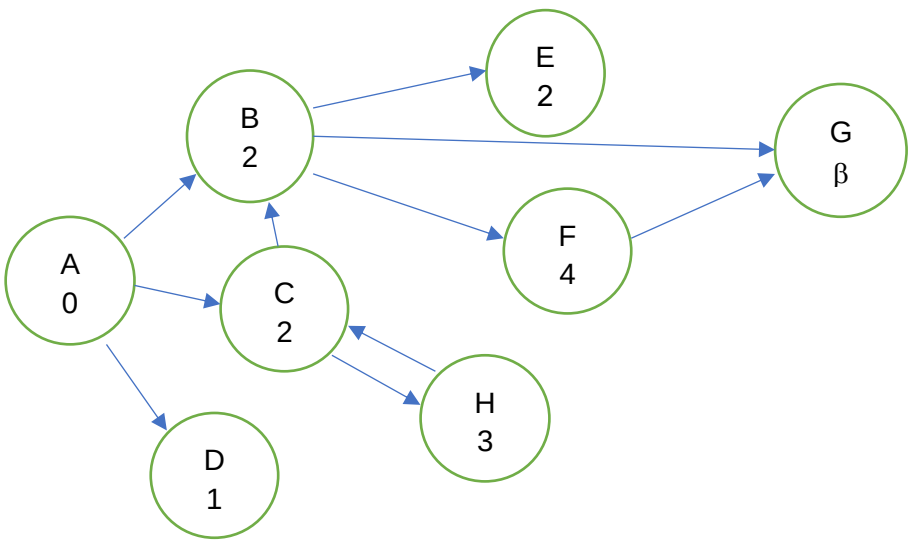
2. יבש (2 נק׳): הסבירו את התוצאות. האם הן תואמות לציפיות שלכם? האם התוצאות היו משתנות עם יוריסטיקה יותר מיודעת? נתחו והסבירו את התוצאות במונחים של מספר פיתוחים, מסלול מוחזר ומחיר הפתרון. שימו לב שבסעיף זה אין תשובה נכונה או לא נכונה אבל נדרש ממכם לספק הסבר מפורט ומבוסס.

ניתן לראות כי בכל המפות אלגוריתם הBFS פיתח יותר מצבים משאר האלגוריתמים וזה מכיוון שהוא מפתח בשכבות, הוא בהכרח יפתח כמות מצבים קבועה (האפשרית בהינתן הבורות) לפי אורך הפתרון. כיוון שאין לו יוריסטיקה שתכוון אותו לכיוון המטרה הבאה הוא מפתח גם מצבים שמרחיקים אותו מהפתרון.
מכיוון שBFS מוצא את המרחק הקצר ביותר מבלי להתחשב במשקל הקשתות הוא החזיר את המסלולים הכבדים ביותר בכל מפה.
עבור אלגוריתם *wA התוארה אומרת שככל ש-w גדל האלגוריתם יפתח פחות מצבים וזאת מכיוון שניתן יותר משקל ליוריסטיקה (אצלינו מרחק מנהטן לכדור/נקודת סיום הקרוב ביותר), ולכן ככל ש-w יותר גדול ניתן פחות משקל ל-g ויותר

ליוריסטיקה שכל הזמן מקרבת לעבר המטרה ולכן יפותחו פחות מסלולים, אך אנו עלולים להחזיר מסלול פחות אופטימלי - כבד יותר.
התוצאות שקיבלנו תואמות באופן חלקי את התאוריה וזאת מכיוון שלמרות שעבור w=0.5 מפותחים יותר מצבים מאשר w=0.7 (תואם את התאוריה) קיבלנו שעבור w=0.9 פותחו יותר מצבים. הסבר אפשר לכך הוא שבמפות ישנם הרבה בורות בסביבת נקודת הסיום, ומכיוון שהיוריסטיקה שלנו עיוורת לבורות היא מנסה לגרום לאלגוריתם שלנו ללכת ישירות לעבר הנקודת סיום אך מכיוון שלא ניתן לעבור דרך הבור נקבל שיפותחו הרבה מסלולים שקרובים מבחינת היוריסטיקה אך לא ניתן לעבור בהם בגלל הבורות.
מבחינת משקל המסלול קיבלנו כי כמעט כל האלגוריתם החזירו את אותו מסלול (מלבד מפה 15x15 בה w=0.9 החזיר מסלול כבד יותר).
אנחנו מניחים כי האלגוריתמים אכן מצאו את המסלול האופטימלי (הרצנו את האלגוריתם עם משקל w נמוך 0.01 על מנת למצוא את הפתרון האופטימלי ואכן התוצאה לא השתנתה), בגלל ריבוי החורים שבמפות וכך שהיוריסטיקה מתעלמת מהבורות קיבלנו שאלגוריתמים מפתחים הרבה מצבים ובכך בודקים הרבה מסלולים שונים דבר שהביא אותם למצוא את המסלול האופטימלי.

שאלה 13 – Local Search (5 נק׳):

בהינתן מרחב המצבים הבא, כאשר a הינו המצב ההתחלתי, $U: S \to \mathbb{R}^+$ הינה פונקציית ערך והערך עבור כל מצב מצוין בצומת. המטרה שלנו היא למצוא מצב שממקסם את ערך U.



נשתמש באלגוריתם <u>Stochastic Hill Climbing</u>.
כמו כן ידוע כי $\boldsymbol{\beta} > \mathbf{3}$.

1. יבש (1 נק׳): מה ההסתברויות למעבר מהמצב ההתחלתי לכל אחד מהמצבים b, c, d. רשמו את $p(d|a), p(b|a), p(c|a)$.

$$p(B|A) = \frac{2}{5},\ \ p(C|A) = \frac{2}{5},\ \ p(D|A) = \frac{1}{5}$$

2. יבש (1 נק׳): מה הוא מספר הצעדים המקסימלי שהאלגוריתם יכול לבצע? צעד מוגדר כמעבר בין מצבים.

אם $\beta > 4$ מספר הצעדים המקסימלי הוא 3 אחרת 2, וזה מכיוון שאנו יכולים ללכת רק למצב שערך הU שלו גדול משלנו, ולכן המסלול הכי ארוך יהיה:

$$A \to B \to F \to^{*} \beta$$

(*) במידה ו- $\beta > 4$.

3. יבש (1 נק׳): בהיתן שבצעד הראשון האלגוריתם עבר למצב c. האם האלגוריתם יתכנס למקסימום הגלובלי?

לא, וזה מכיוון שבשביל להגיע למקסימום הגלובלי שזה F או β האלגוריתם חייב לעבור דרך מצב B, ומתקיים: $p(B|C) = \frac{U_B - U_C}{\sum \Delta U} = \frac{2-2}{\sum \Delta U} = 0$, כלומר מB ניתן להגיע רק לH שהוא מקסימום לוקאלי כלומר לא ניתן להתקדם ממנו.

4. יבש (1 נק׳): מה ההסתברות שהאלגוריתם יתכנס לפתרון לא אופטימלי (שאינו מקסימום גלובלי)?

יש שני מקרים :

נחלק למקרים לפי הערכים של β:

$\beta \geq 4$:

ניתן לראות כי אם האלגוריתם מגיע לB הוא בהכרח יגיע למקסימום מוחלט וזה כי לא ניתן להגיע לE ולכן נעבור או לG שהוא מקסימום מוחלט או לF שאם הוא לא מקסימום מוחלט משם נמשיך לG.

כלומר האלגוריתם לא יגיע למקסימום מוחלט אם אחד מהמקרים הבאים

1) האלגוריתם יתכנס ל-H, ראינו בסעיף קודם שאם האלגוריתם מגיע לB הוא מגיע בהכרח לH ולכן

$$p(A|H) = p(A|B) = \frac{2}{5}$$

2) האלגוריתם יתכנס ל-D, כמו בסעיף הראשון:

$$p(A|D) = \frac{1}{5}$$

סך הכל הסתברות של $\frac{3}{5} = \mathbf{0.6}$.

$\beta < 4$:

במקרה זה β אינו מקסימום גלובלי ולכן נקבל עוד מסלול שאינו מסתיים במקסימום גלובלי, נחשב את ההסתברות שלו:

$$p(\beta|A) = p(B|A) * p(\beta|B) = \frac{2}{5} \cdot \frac{\beta - 2}{2 + \beta - 2} = \frac{2(\beta - 2)}{5\beta}$$

נוסיף את המסלולים מסעיף קודם ונקבל:

$$\boldsymbol{\frac{2(\beta - 2)}{5\beta} + \frac{3}{5}}$$

5. יבש (1 נק׳): עבור אילו ערכים של β ההסתברות להגיע מהמצב ההתחלתי למקסימום הגלובלי תוך <u>בדיוק 3 צעדים</u> גדול מ $\frac{1}{5}$?

אין ערך שכזה.

דרוש כי נגיע אל המקסימום הגלובלי ב3 צעדים כלומר β חייב להיות המקסימום הגלובלי כי אל F ניתן להגיע רק ב2 צעדים.

נשים לב כי אם $\beta \leq 4$ לא נגיע מ-F ל-β כדרוש, ולכן $\beta > 4$.

$$p(A|\beta) = p(A|B)p(B|F)p(F|\beta) = \frac{2}{5} \cdot \frac{2}{2 + (\beta - 2)} \cdot 1 > \frac{1}{5}$$

$$\Rightarrow \frac{2}{\beta} > \frac{1}{2} \Rightarrow \beta < 4$$

קיבלנו שאין ערך של β כנדרש.