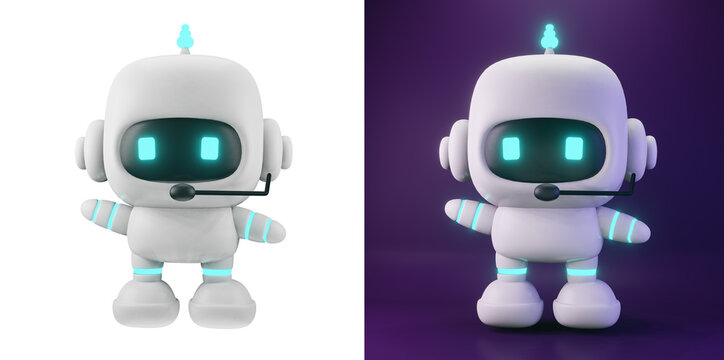
**תרגיל בית 2 מבוא לבינה מלאכותית**

**חיפוש רב סוכנים**

**בינה מחסנים**



**הקדמה ואדמיניסטרציה**

**הנחיות כלליות**

* תאריך הגשת התרגיל: **24.3.2024**
* את המטלה יש להגיש בזוגות בלבד – בקשות להגשה ביחידים באישור המתרגל האחראי בלבד (ספיר טובול).
* יש להגיש את המטלה מוקלדת בלבד – פתרון בכתב יד לא ייבדק.
* התשובות צריכות להיות כתובות בשפה העברית או באנגלית.
* אפשר לשלוח שאלות בנוגע לתרגיל דרך הפיאצה.
* המתרגלת האחראית על התרגיל: **אופק גוטליב**.
* בקשות דחיה מוצדקות יש לשלוח למתרגל האחראי בלבד.
* במהלך התרגיל ייתכן שיעלו עדכונים – הודעה תפורסם בהתאם במקרה זה.
* העדכונים מחייבים וזוהי אחריותכם להתעדכן לגביהם עד מועד הגשת התרגיל. עדכונים יופיעו בטופס בצבע צהוב.
* העתקות תטופלנה בחומרה.
* ציון המטלה כולל חלק יבש וחלק רטוב. בחלק היבש נבדוק שתשובתכם נכונה, מלאה, קריאה ומסודרת. בחלק הרטוב הקוד שלכם ייבדק הן על הגבלת הזמן שתפורט בהמשך ועל אחוזי ההצלחה של האלגוריתמים שלכם לעומת אלו שאנו נממש כבדיקה.
* מומלץ להסתכל בקוד בעצמכם. שאלות בסיסיות על פייתון שלא נוגעות לתרגיל כדאי לבדוק באינטרנט לפני שאתם שואלים בפיאצה. מומלץ לקרוא את הקוד הנתון על מנת להבין את אופן פעולתו – במקרה שישנם דברים לא מובנים . (לשם כך יש הערות רבות ואף הסבר מורחב על הסביבה!)
* מומלץ לא לדחות את התרגיל לרגע האחרון מאחר שהמימוש וכתיבת הדו"ח עלולים לקחת יותר זמן מהצפוי.
* התייחסות בלשון זכר, נקבה או רבים מתייחסים כלפי כלל המינים.
* אין צורך להשתמש בתהליכים ובמאפיינים מורכבים של מערכות הפעלה, התרגיל בית אמור לרוץ על כל מערכת הפעלה ולא למערכות הפעלה ספציפיות. אם ישנה בעייה פרטנית, שלחו מייל למתרגלת האחראית על התרגיל.

**הוראות הגשה**

בתוך קובץ זיפ עם השם : HW2\_AI\_id1\_id2.zip

את הדו"ח היבש בפורמט הבא : id1\_id2.pdf

ואת הקובץ: submission.py שבו אתם ממשים את האלגוריתמים

**הקדמה**

בינה מחסנים רוצים להפוך את המחסן לאוטונומי והם מתלבטים מיהו הרובוט אותו הם רוצים לשכור למשימה. על הרובוטים רובוט R1 ו-R2 להתחרות אחד בשני והמנצח יתקבל לעבודה.

בתרגיל זה תממשו ותחקרו אלגוריתמי משחק סכום אפס אדברסריאלים שלמדתם בהרצאות ובתרגולים.

**תיאור המשחק**

המשחק מתרחש בלוח משבצות בגודל 5X5 כאשר על הלוח: 2 רובוטים, 2 תחנות הטענה,   
2 חבילות ו-2 יעדים (אחד לכל חבילה).   
לכל רובוט יחידות הטענה (battery), וניקוד (credit).

מטרת כל רובוט לצבור יותר ניקוד מהרובוט האחר עד סוף המשחק, המשחק נגמר כאשר לאחד מהרובוטים נגמרת הסוללה או כאשר נגמר מספר הצעדים המקסימלי לכל רובוט (ערך מוגדר מראש, דוגמה בהמשך).

ניקוד נצבר כאשר רובוט מביאה חבילה אל היעד שלה. כאשר רובוט מביאה חבילה ליעד הוא מקבל N יחידות ניקוד כאשר N הוא מרחק מנהטן בין מיקום החבילה המקורי ויעדה של החבילה כפול 2. לאחר שחבילה מגיעה אל היעד שלה היא והיעד שלה נעלמים ובמקומם מופיעים יעד וחבילה חדשים. כלומר, בכל עת על הלוח יש בדיוק שתי חבילות ושני יעדים, אחד לכל חבילה.

רובוטים מתחילים עם טעינה וללא חבילה. כל פעולת תנועה של הרובוט עולה לו יחידת סוללה אחת. רובוט יכול לנוע למעלה, למטה, ימינה, שמאלה. בנוסף, רובוט יכול להטעין את הסוללה בתחנת הטענה כאשר הוא נמצא במשבצת של תחנת הטענה, הוא עושה זאת ע"י המרה של כל יחידות הניקוד שלו ליחידות טעינה. רובוט יכול לאסוף ולהוריד חבילה. רובוט אוסף חבילה כאשר הוא עומד באותה משבצת כמוהה ומבצע פעולת אסיפה. רובוט מוריד חבילה כאשר הוא נמצא ביעד ומבצע פעולת הורדה. (אי אפשר להוריד חבילה שנאספה במשבצת שאינה היעד)

סה"כ לרובוט 7 פעולות אפשריות בכל עת:

move north, move south, move east, move west, pick up, drop off, charge

לא יתאפשר צעד למשבצת לא חוקית (משבצת לא חוקית היא משבצת מחוץ לגבולות הלוח או כזו שנמצא בה הרובוט השני). לא יתאפשר הטענה, אסיפה, הורדה במשבצות לא חוקיות. כל סוכן יכול להטעין בכל אחת מהתחנות הטענה.

**הסבר על הסביבה ו הרצת / דיבוג המשחק**

בשונה מתרגיל בית 1 הפעם תעבדו עם קבצי py ולא מחברות pynb. מוזמנים לעבוד על התרגיל בכל IDE מתאים שנוח לכם לעבודה בפייתון, אנו ממליצים על pycharm של חברת jetBrains אליה יש לכם מנוי מטעם הטכניון.

הסביבה שאיתה תעבדו ממומשת בקובץ WarehouseEnv.py מוזמנים לעיין בה.

בקובץ Agent.py ממומשים סוכנים מהם הסוכנים שאתם תממשו יורשים, מומלץ להסתכל על הסוכנים הממומשים בה בכדי להבין כיצד הם עובדים עם הסביבה.

מומלץ להסתכל על הפונקציות get\_legal\_operator ו - apply\_operator בכדי להבין את אופן ההתממשקות שלכם עם הסביבה.

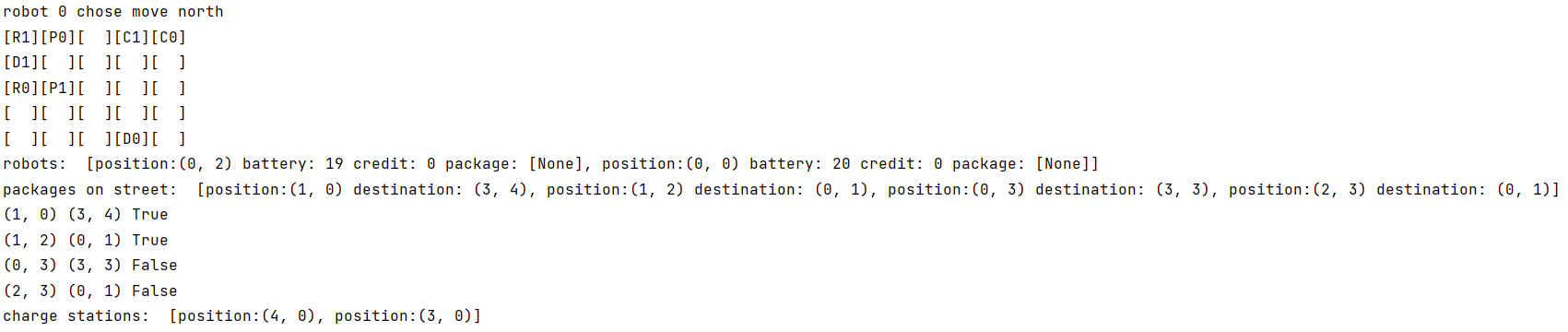
שימו לב שבקובץ submission יש סוכן שנקרא hardcoded, מטרתו לעזור לכם להבין את הסביבה. לפני שאתם שואלים לגבי כיצד הסוכן יתנהג אם יבצע פעולה בדקו בעצמכם בעזרת סוכן זה.

**הרצה**

הרצת משחק נעשית ע"י שורת הפקודה הבאה שמריצים מהטרמינל:

python main.py greedy random -t 0.5 -s 1234 -c 200 --console\_print --screen\_print

* הארגומנט הראשון (במקרה הזה greedy) מציין את האלגוריתם שלפיו ישחק agent0
* הארגומנט השני (במקרה הזה random) מציין את האלגוריתם שלפיו ישחק agent1
* הגבלת זמן לצעד t- מקבלת ערך שמייצג את מספר השניות המקסימילי לצעד
* גרעין לקבלת ערך רנדומלי s- מקבל ערך שעוזר לחולל סביבה באופן רנדומלי, כאשר מועבר אותו ערך seed תחולל אותה סביבה
* מספר הצעדים המקסימלי עבור סוכן c-
* ערך console\_print-- דגל אופציונלי, כאשר מועבר מתבצעת הדפסה לקונסולה של המשחק שנראת כך :

מודפס מספר הסוכן יכול להיות 0 או 1 והאופרטור בו בחר. לאחר מכן מודפס הלוח לאחר שהסוכן הפעיל את האופרטור. הלוח כולל 25 משבצות כאשר בכל אחת יכולים להיות רובוט, חבילה המחכה לרובוט או תחנת הטענה. 

האותיות אשר מופיעות בלוח מסמלות:

R – Robot

C – Charge station

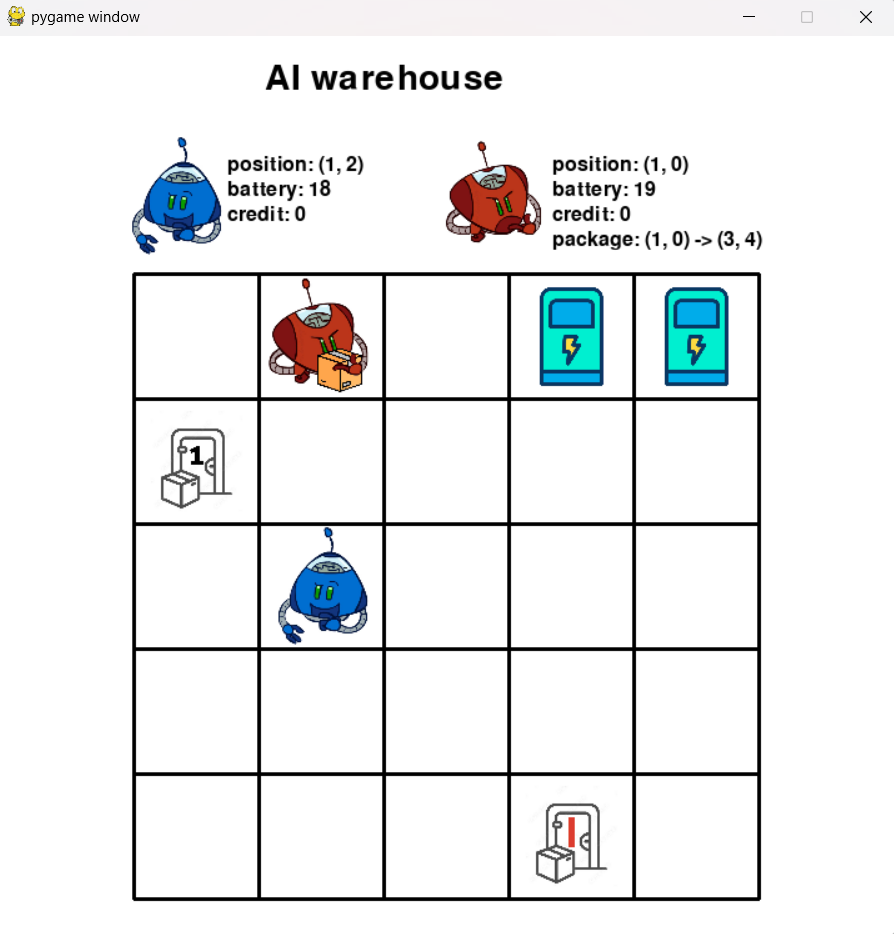
P – Package on street

D – Destination of Package on street

X – Destination of Package taken by Robot

המספר המופיע ליד כל אחת מהאותיות הוא המזהה של האובייקט אליו הוא משתייך – לדוגמא: R1 עבור הרובוט הראשון. עבור חבילה שנאספה ע"י רובוט, המספר ליד X הוא המזהה של הרובוט שאסף אותה. עבור חבילה במרחב, המספר ליד D הוא המזהה של החבילה במרחב. לעיתים מספר אובייקטים מופיעים באותו המיקום בתמונה ואז מודפס רק אחד מהם. במקרה זה, אפשר להשתמש ברשימות המפורשות שמופיעות אחרי הלוח. בנוסף שימו לב! מופיע לכם ברשימות את המקור והיעד של שתי החבילות הבאות שיופיעו על הלוח, מוזמנים ומומלץ להשתמש במידע זה בהמשך התרגיל.

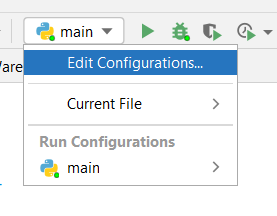
* ערך screen\_print-- דגל אופציונלי, כאשר מועבר יודפס לכם הנפשה של המשחק בחלון pygame שנראה כך:

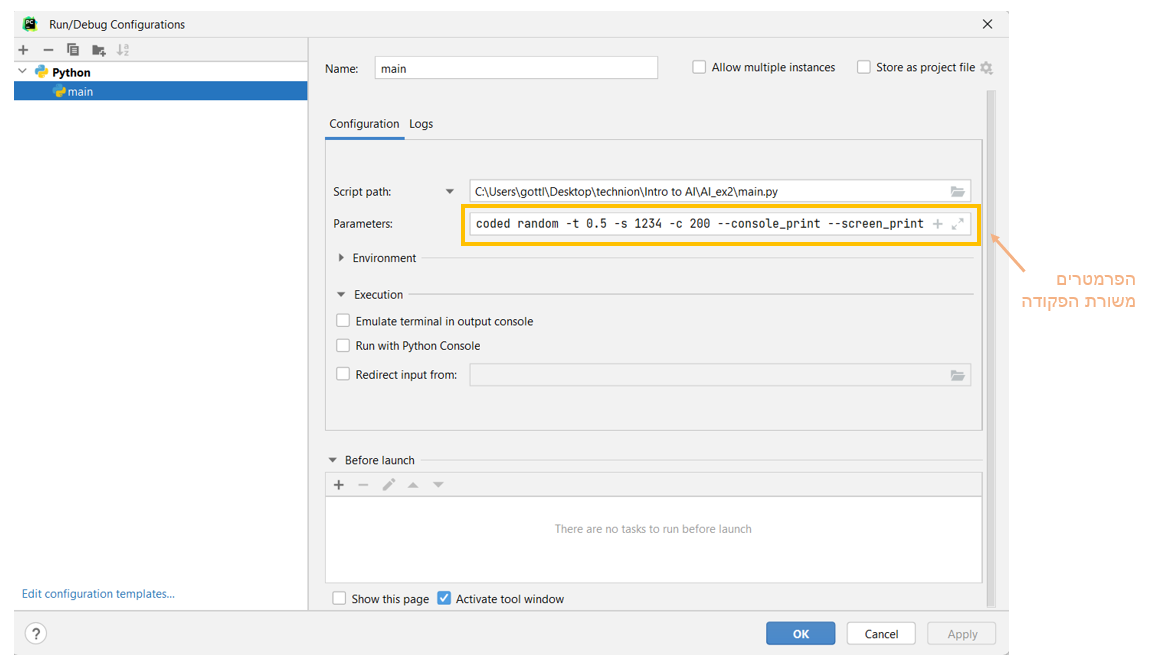


בנוסף בכדי לבדוק את הסוכנים שלכם על מספר רב של משחקים, אתם יכולים להוסיף את הדגל tournament-- וכך תקבלו הרצה רצופה של n משחקים. מוזמנים לשחק עם המשתנה של מספר המשחקים ולראות האם מקבלים התנהגויות רצויות עבור האלגוריתמים.

**דיבוג**

בכדי לדבג את המשחק עליכם לעקוב אחר השלבים הבאים:

2.



**מתחילים לכתוב !**

**חלק א - ImprovedGreedy**

1. (יבש: 4 נק') כפי שלמדתם, אנו מגדירים בעייה במרחב בתור רבעייה (S, O, I, G). הגדירו פורמלית (הסבירו כיצד נראים ערכי הרבעייה) את המשחק המתואר לכם ע"פ הנתונים שאתם מקבלים מהסביבה.

המשחק יוגדר באופן הבא:

S = (R,P,C,num\_steps)

כאשר:

R מייצג את שני הרובוטים: מיקום הרובוט, אחוז הסוללה שנותר, נקודות זכות, חבילה שנמצאת בידיהם (אם קיימת)

P מייצג את החבילות: מיקום החבילה, מיקום היעד

C מייצג את מיקום תחנות ההטענה: מיקום התחנה

Num\_steps מייצג את מספר הצעדים שנותרו עד לסיום המשחק

O = { North, South, West, East, Pick up, Charge, Drop off }

I מייצג את המצב ההתחלתי של המשחק:

כל רובוט מתחיל עם 20 אחוזי סוללה, 0 נקודות זכות ונקודת התחלה אקראית.

2 חבילות ממוקמות באופן אקראי על הלוח. 2 תחנות ההטענה ממוקמות באופן אקראי על הלוח.

G מכיל את המצבים שבהם נגמרו מספר הצעדים לביצוע או שנגמרו אחוזי הסוללה של אחד הרובוטים

1. (יבש: 4 נק') הגדירו היוריסטיקה משלכם להערכת מצבי המשחק. עליכם לתעד אותה בנוסחה מפורשת ועלייה לכלול לפחות שלושה מאפיינים של הסביבה. בחרו שמות ברורים בנוסחה שלכם.

נפריד למקרים:

אם הרובוט אסף חבילה, אז ההיוריסטיקה תהיה:

H(s,k) = k.credit + manhatan(k.position, package.destination)

כלומר, ההיוריסטיקה תחזיר את הסכום של נקודות הזכות של הרובוט ומרחק מנהטן עד ליעד החבילה

אם הרובוט לא אסף חבילה, אז ההיוריסטיקה תהיה:

H(s,k) = k.credit + min(manhatan(k.position, package0.position),

manhatan(k.position,package1.position)

כלומר, ההיוריסטיקה תחזיר את הסכום של נקודות הזכות של הרובוט ומרחק מנהטן מהחבילה הקרובה לרובוט( במידה ולא נאספה)

1. (רטוב: 10 נק') ממשו בקובץ submission.py את הפונקציה **smart\_heuristic**

שבה משתמש הסוכן AgentGreedyImproved

1. (יבש: 2 נק') מהו החיסרון העיקרי של האלגוריתם? (לעומת minimax)

אלגוריתם זה מנסה למקסם את התועלת של סוכן יחיד, ללא התחשבות בפעולות של הסוכנים האחרים.

**חלק ב - RB-Minimax**

1. (יבש 3 נק') מה היתרונות והחסרונות של שימוש בהיוריסטיקה קלה לחישוב לעומת היוריסטיקה קשה לחישוב בהינתן שהיוריסטיקה הקשה לחישוב יותר מיודעת מהקלה לחישוב ? בהינתן שאנו בmin-max מוגבל משאבים.

בעזרת היוריסטיקה קשה לחישוב, נוכל לקבל הערכה טובה יותר של מצבי המשחק, מה שמגביר את הסיכויים לבצע פעולות שיועילו לסוכן.

לעומת זאת, בעזרת היוריסטיקה קלה לחישוב, נוכל לבצע מספר גדול יותר של צעדים באלגוריתם, להגיע לעומק גדול יותר ולבחון מספר רב יותר של פעולות אפשריות. בנוסף, היוריסטיקה קלה לחישוב דורשת פחות משאבים.

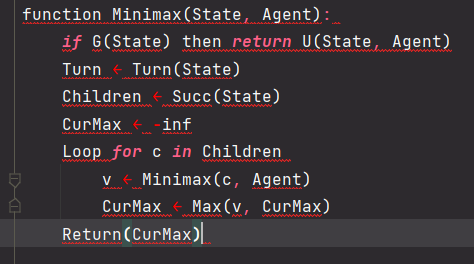
1. (יבש: 4 נק') חברתכם לקורס דנה מימשה סוכן minimax, היא שמה לב כי לעיתים הסוכן יכול לנצח בצעד אחד אך הוא בוחר בצעד אחר. האם יש לה באג באלגוריתם? אם אין באג הסבירו מה באלגוריתם גורם להתנהגות שכזו. אם יש באג מה הוא יכול להיות?

לא קיים באג באלגוריתם עקב העובדה כי ייתכן שהאלגוריתם הצליח להבטיח בעזרת minimax ניצחון בעל תועלת גדולה או שווה לזו המושגת בצעד אחד. במקרה זה, יבחר האלגוריתם בצעד שיוביל לתועלת גבוהה יותר או ייבחר באקראי בצעד אחר שיוביל לתועלת דומה(אם 'ניצחון' הוא מצב בעל התועלת המקסימלית, ייתכן כי יש באג באלגוריתם כתלות באופן שבו בוחרים במקרה של שוויון תועלות).

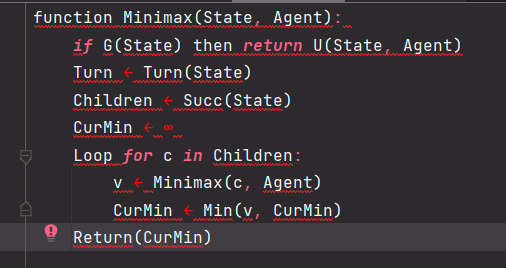
נשים לב כי האלגוריתם מנסה למקסם את התועלת של הסוכן ולא להביא למינימום את מספר הצעדים.

1. (רטוב: 10 נק') עליכם לממש את המחלקה AgentMinimax בקובץ submission.py. שימו לב! הסוכן מוגבל משאבים, כאשר המשתנה time\_limit מגביל את מספר השניות שהסוכן יכול לרוץ לפני שיחזיר תשובה. (הגבלת הזמן עלייה אתם נבדקים הינה שנייה כלומר 1 t-).
2. (יבש: 3 נק') נניח שבסביבה היו K שחקנים במקום 2. אילו שינויים יהיה צריך לעשות במימוש סוכן Minimax? **כתבו פסאודו קוד בדומה לזה שראינו בתרגול.**
   1. בהינתן שכל סוכן רוצה לנצח ולא אכפת לו רק ממכם.
   2. בהנחה והדבר היחיד שכל סוכן רוצה הוא שלא תנצחו.
   3. בהנחה שכל סוכן רוצה שהסוכן שאחריו בתור ינצח.

a. במקרה זה, כל סוכן יבצע max על התועלות שיקבל מכל פעולה אפשרית מאחר והוא חותר לניצחונו האישי בלבד, ולכן ינסה למקסם את תועלתו.

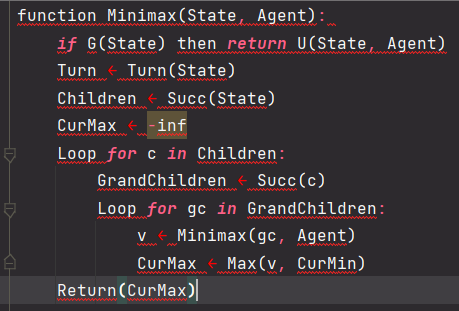


b. במקרה זה, כל סוכן יבצע min על התועלות שיקבל מכל פעולה אפשרית מאחר והוא וחותר להפסדנו האישי בלבד, ולכן ינסה למזער את התועלת שלנו.



c. במקרה זה, כל סוכן יבצע max על התועלות, אך הפעם, ינסה לבחון 2 צעדים קדימה - צעד שלו וצעד של הסוכן שאחריו, וינסה למקסם את התועלת של הסוכן שאחריו.

למעשה, הסוכן ינסה למקסם את התועלת של ה"נכדים" שלו.



**חלק ג - Alpha-Beta**

1. (רטוב: 10 נק') ממשו שחקן אלפא - בטא מוגבל משאבים במחלקה AgentAlphaBeta בקובץ submission.py, כך שיתבצע גיזום כפי שנלמד בהרצאות ובתרגולים.
2. (יבש: 3 נק') האם הסוכן שמימשתם בחלק זה יתנהג שונה מהסוכן שמימשתם בחלק ב מבחינת זמן ריצה ובחירת מהלכים ? הסבירו.

הסוכן שמימשנו בחלק זה עשוי להתנהג שונה מהסוכן שמימשנו בחלק ב.

באמצעות אלגוריתם Alpha-Beat, ניתן לגזום צמתים מהעץ שגם ככה לא נצטרך לפתח על מנת להגיע לפתרון. כלומר, בחירת המהלכים לא תשתנה, שכן נקבל את אותם ערכי יוריסטיקה בכל צומת.

אולם, עקב העובדה כי עשוי להתבצע גיזום והעובדה כי האלגוריתם מוגבל משאבים, ייתכן כי ייחסך זמן ריצה על פיתוח צמתים מיותרים, וכך האלגוריתם את מהלך המשחק בעץ בעומקים גדולים יותר, ואולי לבחור פעולה אחרת.

**חלק ד - Expectimax**

1. (יבש: 3 נק') בהנחה ואתם משתמשים באלגרותים Expectimax נגד סוכן שמשחק באופן רנדומלי לחלוטין באיזה הסתברות תשתמשו? ומדעו?

מאחר והסוכן משחק באופן רנדומלי לחלוטין, נשתמש בהסתברות יוניפורמית, שכן לכל צעד יש סיכוי זהה להתבצע

1. (יבש: 4 נק') עבור משחקים הסתברותיים כמו שש בש, בהם יש מגבלת משאבים, משתמשים באלגוריתם .RB-Expectimax הניחו כי ידוע שהפונקציה היוריסטית ℎ באלגוריתם Expectimax-RB מקיימת

איך ניתן לבצע גיזום לאלגוריתם זה? תארו בצורה מפורטת את תנאי הגזימה, והסבירו את הרעיון מאחוריו.

הגיזום יתבצע כאשר בצומת נגלה שאחד הבנים הוא בעל ערך 1 או צומת min שאחד הבנים הוא בעל ערך של 1-. מכיוון שהפונקציה היוריסטית חסומה, לא נקבל ערך גדול יותר מ1 או קטן מ- 1-, ולכן נוכל לגזום בנים, כולל כאלה שעושים חישוב של תוחלת בין 1- ל1 ( ולכן לא יקבלו ערך גבוה יותר).

1. (רטוב: 10 נק') הסוכן של minimax ו-alpha-beta מניח שהסוכן היריב יבחר באופרטור שיוביל לתוצאה האופטימלית בתורו, אולם זה לא תמיד מתרחש.

לדוגמה, כאשר אנו מתחרים עם סוכן חמדן, סביר להניח שהוא לא יבחר בפעולה האופטימלית בכל צעד. אפשר להתחשב באפשרות שהיריב יבחר בפעולה שאינה אופטימלית בתורו באמצעות סוכן Expectimax.

גילינו מידע סודי על הרובוט המתחרה, הוא בוחר בין כל הפעולות בצורה יוניפורמית (בצורה אחידה) אבל לתזוזה ימינה ולאסיפת חבילה (כאשר פעולות אלו אפשריות) יש הסתברות גדולה פי 2 מלשאר הפעולות.

ממשו אלגוריתם Expectimax המשתמש במידע הסודי שקיבלתם.

**חלק ה - משחק עם פקטור סיעוף גדול**

1. (יבש: 6 נק') להלן שינויים אפשריים ששוקלים בינה מחסנים לעשות במשחק בכדי לבחון יכולות נוספות של הרובוטים. עבור כל שינוי ציינו מה ההשפעה שלו על מקדם הסיעוף וחשבו את מקדם הסיעוף החדש המתקבל.
   1. הגדלת לוח המשחק להיות 8X8 והוספת מחסומים בסביבה. (מחסומים משמע משבצות שהסוכן לא יכול לעבור בהן)
   2. הוספת היכולת של רובוט בכל תור לבחור משבצת על הלוח ולהניח עלייה בלוק, משמע בכל תור יכול הרובוט לנוע למעלה, למטה,ימינה, שמאלה, לאסוף חבילה, להוריד חבילה, להטעין, ולהניח בלוק על הלוח, בלוק יכול להיות מונח על כל משבצת ריקה.

a. מקדם הסיעוף לא ישתנה. הגדלת הלוח אינה מוסיפה פעולות אפשריות עבור כל סוכן, אלא פשוט תגדיל את מרחב האפשרויות למיקום רובוט/חבילה/תחנה.

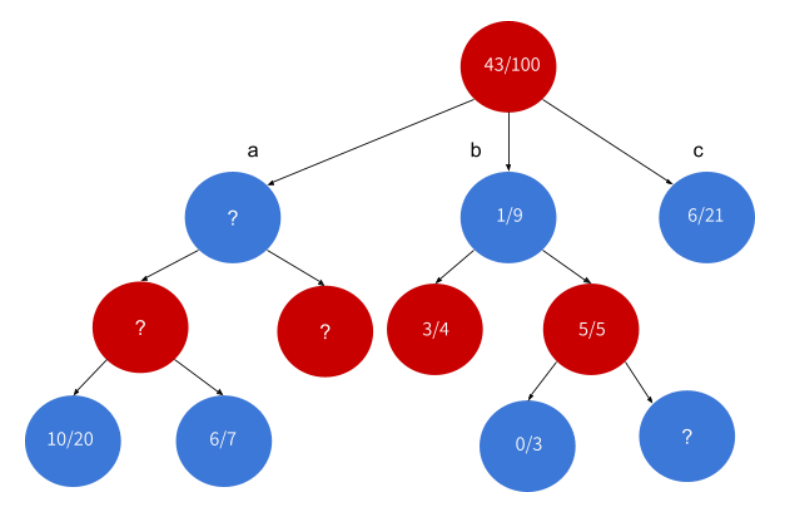
הוספת מחסומים עשויה להקטין את מקדם הסיעוף במשבצות ספציפיות עקב ביטול פעולה אחת או יותר מאותה המשבצת. למרות זאת, ייתכן כי יהיו משבצות שלא יפגעו מהוספת המחסומים, ולכן מקדם הסיעוף לא יושפע.(5)

b. מקדם הסיעוף יגדל ב-1. הנחת מחסום על אחת המשבצות מוסיפה עוד פעולה שהרובוט יכול לבצע, ולכן יש יתווסף מקרה נוסף בעץ בו הרובוט יוכל להניח מחסון על משבצת כלשהי.(6)

1. (יבש: 6 נק') בהנחה ומימשו את השינוי השני עבור הסביבה (סעיף 1b)
   1. האם יש אלגוריתם מהסעיפים הקודמים שנוכל להשתמש בו שזמן הריצה שלו סביר? (סביר משמע לא גדול מהותית מהזמן שלוקח לו להחזיר צעד עבור המשחק בלי השינוי).
   2. הציעו אלגוריתם שונה מאלו שממשתם בסעיפים הקודמים שנלמד בקורס שירוץ בזמן סביר. הסבירו מדוע בחרתם בו ולמה הוא טוב להתמודדות עם האתגר שנוצר משינוי הסביבה.

**חלק ו - יבש - שאלה פתוחה - MCTS**

שחקן אדום ושחקן כחול שיחקו משחק. להלן עץ המשחק שמתאר את העץ שנוצר בשלב ביניים בהרצת MCTS עם פיתוח צמתים לפי UCB1 על משחק סכום אפס בין שניהם, נתון .



9/43

50/70

0/2

11/27

דגש על האלגוריתם: הערך בצומת הכחול מייצג את כמות הניצחונות של השחקן האדום מתוך כמות המשחקים שבוצעו עם הצעד הזה (ולהיפך). למשל צומת b מייצג את כמות הניצחונות של השחקן האדום אם בחר בפעולה שגרמה לו להגיע למצב b מתוך סך כל המשחקים ששוחקו עם הצעד.

1. (5 נק') חלק מהערכים בצמתים נמחקו, השלימו את החלקים החסרים, אין צורך לנמק.
2. (5 נק') הצומת הבא שייבחר בשלב ה - selection יהיה (הוסיפו חישובים לנמק את בחירתכם):
   1. צאצא של a
   2. צאצא של b
   3. צאצא של c

ע"פ החישוב שהוצג בהרצאה:

עבור צאצא a נקבל :

עבור צאצא b נקבל :

עבור צאצא c נקבל :

לכן, נבחר בצאצא b

1. (5 נק')בהנחה שכל סימולציה מכאן והלאה מסתיימת בניצחון של השחקן הכחול מה מספר הניצחונות המינימלי שנדרש כדי שצאצא אחר של השורש ייבחר בשלב ה-selection (הוסיפו חישובים לנמק את תשובתכם)?
2. (3 נק') כעת רוצים לבצע שינוי כך שנעדיף exploration יותר מ - exploitation. הגישה לנוסחה שמחשבת את ה - UCB1 חסומה לכם, אך הנוסחה משתמשת ב - אשר אליו יש לכם גישה ואתם יכולים לשנותו. כיצד תשנו אותו בכדי שהנוסחה החדשה שתיווצר תעדיף יותר exploration מ - exploitation לעומת הנוסחה הקודמת.

להגביל מינימקס בעומק (של 4).

לשים לב שלפעמים לא שווה להטעין.

ז"א לתעדף הטענה פחות מאשר להתאבד.

לנסות להתחשב במספר הצעדים שנותרו.

לגבי הגבלת הזמן – איטרטיביות או THREADS. לבדוק כמה עומקים לוקחים וכל עוד נשאר אפסילון זמן אתה יכול להמשיך לרוץ. לבדוק כמה זמן נשאר מאז שהתחלתי, ואז לוודא אם זה קטן מאפסילון (לא להשתמש ב THREADS). לבדוק כמה זמן לקח לי עד סוף האיטרציה ה i ואז להשוות מול אפסילון.

Start\_time

Curr\_time

ידוע כמה זמן האלגוריתם רץ. נעשה פונקציה שעוטפת ולפני שניכנס למינמקס נבדוק שההפרש קטן יותר מאפסילון זמן. אם לא, אז נצא מהלולאה.

Alpha = c.time – s.time

מינימקס לא אמור לשפר יותר מדי מול הגרידי.

לא ללכת להטעין אם לך יש יותר קרדיט ולאויב אין סוללב= .