**מבוא לבינה מלאכותית**

**תרגיל בית 2**

מגישים:

אביב כספי 311136691

יקיר יהודה 205710528

**חלק ב'**

**2.** השחקן GreedyAgent עובד בצורה הבאה:

השחקן עובר על כל אפשרויות ההתקדמות שלו, ומבצע בדיקה עבור כל פעולה. כאשר בהינתן הפעולה אותה בחר השחקן לבצע, השחקן בודק את כל האופציות לפעולות של אויביו, כאשר עבור כל אופציה של פעולות, הוא בונה מצב חדש ובודק בעזרת היוריסטיקה (שתוסבר בהמשך) , איזה מהמצבים מחזיר ערך מקסימלי. עבור מצב מקסימלי זה השחקן יבחר לבצע את הפעולה שהביאה אותו לאותו מצב. (עבור יותר משני אוייבים, השחקן יפתח רק מצב אחד עבור פעולות אויביו, בשביל לחסוך בזמן וזיכרון).

היוריסטיקה שהשחקן GreedyAgent משתמש:

אם הוא מת, מחזיר את האורך שלו.

אם הוא חי, בודק כמה פירות נשארו לאכול (פירות במסך + אורך הנחשים האויבים החיים) ומחזיר את אורך הנחש + תוספת עבור מספר הפירות שנשארו ביחס לכמות התורות שנשארו.

השחקן נמנע ממוות על ידי כך שנותן ציון יוריסטי נמוך למצב בו הוא מת ביחס למצב בו הוא חי.

**חלק ג'**

1. **יוריסטיקה :** 1- אורך הנחש, 2- מרחק לתפוח הקרוב ביותר, 3- עניין הקלאסטר , 4- שטח שניתן להגיע אליו לאחר ביצוע הפעולה.
2. אורך הנחש נועד על מנת למנוע מוות, וכדי לגרום לנחש לאכול את התפוח כאשר הוא מתקרב אליו.

מרחק לתפוח – יגרום לנחש להתקרב לתפוח הקרוב ביותר

קלאסטר – יגרום לנחש לבחור תפוח יותר רחוק כאשר יש לידו מספר גדול של תפוחים לעומת תפוח קרוב יותר עם סביבת תפוחים קטנה יותר

שטח שניתן להגיע – מונע מהנחש לסגור את עצמו כאשר יש תפוח במקום תקוע.

הבדל מהיוריסטיקה הקודמת , מרחק לתפוח מונע מהנחש להישאר בלולאה .

**חלק ד'**

2. תחילה אנחנו מצפים שיעבוד יותר טוב מGREEDY בגלל השימוש בהיוריסטיקה שלנו אשר ממפה מצבים בצורה טובה יותר וגם בגלל שכעת אנחנו מסתכלים לעומק עמוק יותר.

עבור מקרים בהם ייתכן כי נתקל במכשול (סוף הלוח או נחש אחר) במרחק שגדול מצעד יחיד, האלגוריתם יזהה את הבעיה (כל עוד המכשול בעומק בו מחפשים) .

1. כאשר כל היריבים בחיים, לכל צומת מינימום יהיה בנים, כאשר כל בן מסמל בחירה של פעולה עבור כל אחד מהאויבים (לכל אויב יש 3 אפשרויות פעולה).

עבור k=20 מספר הבנים לכל צומת מינימום יהיה  *בנים, כלומר על מנת לחשב את תוצאת האלגוריתם יהיה עלינו לעבור על כל הבנים של כל צומת מינימום ולכן אי אפשר.*

1. *כאשר יש יותר מיריב אחד, אנחנו מניחים כי כל אחד מהיריבים, יבצע פעולה שהכי פחות טובה לשחקן שלנו בנוסף אנחנו מניחים כי כל שחקן משחק בתורו ויחסית לפעולות של השחקנים לפניו. במשחק שלנו ההנחה הזאת אינה נכונה, מפני שאצלנו צומת מינימום יחיד שייך לכל היריבים יחדיו. כלומר במקרה שלנו ההנחה היא כי כל היריבים משחקים יחדיו נגדנו, וכי כל האויבים משחקים בתור יחיד יחסית לפעולה שאנחנו עשינו.*
2. *a. במקרה בו נרצה לממש שכבה נפרדת לכל אויב, נפרק את המצבים שהיו לנו עכשיו, כאשר לכל אויב תהיה שכבה בעץ המצבים.*

*חסרונות מימוש זה : זיכרון נוסף עבור כל מצב פנימי בין צומת המקסימום לבנים האחרונים של צמתי המינימום. בין כל שני צמתי מקסימום יהיו לנו עכשיו k צמתי מינימום עבור כל אחד מהיריבים לעומת צומת מינימום אחת במימוש הקודם, הוספה של .*

*יתרונות : מבחינת מימוש, עבור כל צומת מינימום נעבור רק על אפשרויות הפעולה של יריב זה (כלומר 3 פעולות אפשריות) לעומת המימוש הנוכחי, שדורש לעבור על כל אפשרויות היריבים במקביל ולהחזיר מצב יחיד.*

*b. ההנחה במקרה זה היא כי כל יריב פועל בתורו, ביחס לפעולות הקודמות של השחקנים שלפניו. בנוסף בגלל מימוש MINIMAX , ההנחה היא כי כל יריב בוחר את הפעולה הרעה ביותר עבור השחקן שלנו. במימוש הקודם, כל היריבים שיחקו במקביל באותו תור, וכולם עבדו יחדיו כנגדנו. (במשחק אצלנו, אין באמת משמעות לתורות, בגלל שכל הצעדים של השחקנים נעשים במקביל בלי תלות).*

***חלק ה'***

*2. a. מבחינת זמן ריצה, הסוכן אלפא בטא ישתפר ברוב המקרים, בגלל קיצוץ של מסלולים שלא יביאו לשיפור.*

*b. הסוכן יפעל בצורה זהה לסוכן minimax , מפני שבחירת המהלכים זהה במהלך האלגוריתם וההבדל היחיד הוא ויתור על פיתוח מהלכים שלא ישפרו.*

***חלק ו'***

1. *מניחים שהמשחק קבוע ואינו משתנה מפני שהאלגוריתם שלנו מנסה לשפר את המצב הבא יחסית ללוח הנוכחי, כלומר אם כל איטרציה נשנה את הלוח, האלגוריתם ינסה להשתפר עבור הלוח הנוכחי, אך שיפור זה לא יהיה רלוונטי עבור הלוח הבא. (לוח הכוונה לכל הנתונים כולל מיקום הנחשים והתפוחים)****,*** *מימוש זה לא יניב תוצאות טובות.*
2. *המשחק אינו סכום אפס לפי הגדרת התרגיל, נשים לב כי פונקציית התועלת במשחק שלנו היא כך שכל שחקן מקבל ניקוד השווה לאורכו + 1 אם השחקן חי, כלומר ניקוד כל שחקן יהיה תמיד חיובי (מתחיל מגודל חיובי) ולכן אם נסכום את התועלת של כל השחקנים בהכרח נקבל ערך חיובי.*
3. *רשום בדף*
4. ***רשום בדף***
5. *מספר האיטרציות שנבצע בחיפוש זה הינו כגודל מערך הפעולות במקרה שלנו N פעולות.*
6. *6*
7. *בחרנו להתחיל ממצב בו כל הפעולות הן פעולות להתקדם ישר.*

*בחרנו במצב זה לאחר בדיקות של מצבים רבים, ובחירה במצב שהביא לנו תוצאות טובות ביותר.*

*עפ"י פונקציית התועלת שהוגדרה לתרגיל get\_fitness , מצב זה נותן תוצאות טובות מפני שבכל איטרציה, הנחש שלנו בודק באיזה כיוון יש מספר מקסימלי של תפוחים במסלול ישר מהנקודה הנוכחית שלו, ובוחר להתקדם לשם וממשיך באותו כיוון עד אשר יגיע לנק' בה מספר התפוחים שיוכל לאכול בצדדים יגדל ממספר התפוחים שנותרו לו במסלול הישר.*

*להוסיף 3 תמונות של נחש שפועל לפי האלגוריתם במצב זה*

1. *כאשר מבצעים שינוי בפעולה ה-i משפיעים רק על המצבים שיבדקו לאחר פעולה זו.*

*מצבים שהיו לפני הפעולה נשארו כמו שהם, מפני שהמעבר מהמצב ההתחלתי עד למצב ה-i אינו משתנה.*

*לעומת זאת כל המצבים שיופיעו אחרי הפעולה ה-i יושפעו, מהסיבה שכל מצב תלוי בכל הצעדים הקודמים לו. מספר המצבים שיושפעו בשינוי זה הוא לכל היותר (N-i)\*3 כאשר N הוא מספר הפעולות הכולל, מפני שבכל איטרציה נפתח 3 מצבים ולאחר שינוי i כל המצבים שיפותחו יהיו חדשים.*

*10.*

[<GameAction.STRAIGHT: 1>, <GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>, <GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>, <GameAction.LEFT: 0>,

<GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.LEFT: 0>,

<GameAction.LEFT: 0>, <GameAction.STRAIGHT: 1>,

<GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>,

<GameAction.LEFT: 0>, <GameAction.RIGHT: 2>,

<GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>,

<GameAction.LEFT: 0>, <GameAction.LEFT: 0>,

<GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.LEFT: 0>,

<GameAction.RIGHT: 2>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.STRAIGHT: 1>, <GameAction.RIGHT: 2>,

<GameAction.LEFT: 0>, <GameAction.LEFT: 0>]

Winner: WinnerAtTurn(player\_index=0, length=13)

הנחש שלנו הגיע לאורך 13, וניצח!

11. בחרנו לממש עבור סעיף זה, את האלגוריתם הגנטי אשר בכל איטרציה, מריץ דור של נחשים (וקטור באורך N של פעולות). האלגוריתם מדרג את הנחשים לפי פונקציית התועלת שהוגדרה לנו, ויוצר נחשים "ילדים" חדשים. האלגוריתם בוחר בצורה הסתברותית מבין הנחשים זוג הורים מהדור הנוכחי (זוג הורים יבחרו יחסית לfitness שלהם), ומבצע הצטלבות בין זוג ההורים (בהסתברות שנגדיר להיות 0.5, אחרת מעתיק את ההורים כמו שהם), לאחר מכן מבצע מוטציה בוקטור הפעולות של כל אחד מהילדים בהסתברות נמוכה (אשר נגדיר להיות 0.05). כאשר נגדיר ביצוע מוטציה לפעולה ה-i , כשינוי אקראי של הפעולה ה-i בוקטור הפעולות של הילדים, לפעולה אקראית אחרת.  
לסיום אנחנו מוסיפים את כל הנחשים הילדים לדור חדש, ומריצים את האלגוריתם מחדש.

בחרנו באלגוריתם זה, מפני שאנחנו חושבים שכאשר יש לנו זוג נחשים שפועל בצורה טובה, איחוד הפעולות שלהם + מוטציה לחלק מהפעולות, יכול להניב נחש חדש, שיתגבר על המכשולים שכל אחד מהוריו נתקל בריצתו.

שינויים שנבצע באלגוריתם :

במימוש ההצטלבות, בחרנו לבצע את ההצטלבות מנק' רנדומלית בוקטור הפעולות כאשר כל הפעולות לפני פעולה זו ילקחו מהורה אחד, וכל שאר הפעולות מההורה השני.

ביצענו שינוי זה, מפני שלדעתנו יש לשמור כמה שיותר על סדר הפעולות שביצע ההורה מבלי לערבב פעולות מההורה השני באמצע (פעולות שיופיעו באמצע הרצף יכולות לפגוע בהתקדמות הנכונה שהגיעה מההורה).

לשנות שההורה עם הFITNESS הגדול יותר יתרום חלק גדול יותר לילד ושלא יהיה באמצע

12. רשום בדף

14. [<GameAction.RIGHT: 2> <GameAction.LEFT: 0> <GameAction.LEFT: 0>

<GameAction.STRAIGHT: 1> <GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>

<GameAction.LEFT: 0> <GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>

<GameAction.STRAIGHT: 1> <GameAction.LEFT: 0> <GameAction.LEFT: 0>

<GameAction.STRAIGHT: 1> <GameAction.LEFT: 0> <GameAction.RIGHT: 2>

<GameAction.STRAIGHT: 1> <GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>

<GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2> <GameAction.RIGHT: 2>

<GameAction.LEFT: 0> <GameAction.RIGHT: 2> <GameAction.LEFT: 0>

<GameAction.STRAIGHT: 1> <GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>

<GameAction.RIGHT: 2> <GameAction.RIGHT: 2> <GameAction.STRAIGHT: 1>

<GameAction.STRAIGHT: 1> <GameAction.LEFT: 0> <GameAction.RIGHT: 2>

<GameAction.RIGHT: 2> <GameAction.STRAIGHT: 1> <GameAction.STRAIGHT: 1>

<GameAction.LEFT: 0> <GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>

<GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2> <GameAction.RIGHT: 2>

<GameAction.RIGHT: 2> <GameAction.RIGHT: 2> <GameAction.STRAIGHT: 1>

<GameAction.RIGHT: 2> <GameAction.RIGHT: 2> <GameAction.LEFT: 0>

<GameAction.STRAIGHT: 1> <GameAction.RIGHT: 2>]

הסוכן שלנו הגיע לאורך 4, ולא ניצח. ביצענו 30 איטרציות וראינו כי אין שיפור בפעולת הנחש שלנו.

לאחר בדיקה, הגענו למסקנה כי האלגוריתם הגנטי אינו מתאים לבעיה זו, מפני שלקיחת חלק מהפעולות של ההורה, מוציאה את הפעולות מהקשר. ייתכן כי אותן פעולות היו טובות לנחש ההורה לאחר שעשה פעולות שקדמו לפעולות שלקחנו, אך לאחר השינוי, פעולות אלה התחילו ממצב התחלתי שונה, ולכן לא הובילו לביצועים טובים כמו אצל ההורה.