תרגיל 2

**Introduction to Artificial Intelligence (236501)**

מגישים: אייל אמדור, בארי זיטלני

318849270, 209351626ת.ז:

ImprovedGreedy

שאלה 1

נפרמל את הבעיה:

* S – מרחב המצבים, כאשר נגדיר כך:  
  כאשר נגדיר מושגי עזר:  
  ונגדיר:
  + – המיקום הנוכחי שלי בלוח, יוגדר להיות tupple באופן הבא:
  + battery – כמה סוללה נשארה לרובוט, ערך גדול/שווה 0.
  + credit – כמה נקודות צבר הרובוט, ערך גדול/שווה 0.
  + package – חבילה שיש ברשות הרובוט, יכול לקבל כערך:
* O – פונקציית מעברים, כאשר נגדיר:  
  כאשר:  
  כלומר בכל תור משחק בדיוק סוכן אחד.
* I – קבוצת המצבים ההתחלתיים, כאשר נגדיר:  
  כאשר,
* G – קבוצת מצבים הסופיים:  
  כלומר או שלאחד הרובוטים נגמרה הסוללה או שקצבת הצעדים הכוללת למשחק נגמרה.

שאלה 2

נגדיר את היוריסטיקה הבאה עבור :

שאלה 3

מימוש בקוד.

RB-Minimax

שאלה 1

בהינתן מינמקס מוגבל במשאבים, שימוש ביורסטיקה קלה לחישוב ייתן עץ פעולות עמוק יותר ולכן ייצג בחירה על מרחב אפשרויות רחב יותר – כלומר ניבוי שחוזה בצורה מדויקת יותר את "עתיד" המשחק. לעומת זאת, היורסטיקה הינה קלה ולכן מתארת פחות נאמנה את המציאות, כלומר מביאה לידי ביטוי פחות פרמטרים או מורכבויות שצעדים מסוימים עלול ליצור.

מנגד, שימוש ביורסטיקה כבדה לחישוב ייתן עץ פעולות עמוק פחות ולכן ייצג בחירה על מרחב אפשרויות מוגבל יותר כלומר ינבא פחות מהשלכות הצעד במשחק ("רואה פחות רחוק"). לעומת זאת, היורסטיקה הכבדה מתארת בצורה מהימנה יותר את המציאות ולכן השערוך היורסטי טובה יותר.

שאלה 2

ייתכן ולדנה אין באג באלגוריתם. כתלות במימושה לאלגוריתם מינימקס, אין התחייבות של האלגוריתם לבחירת הניצחון המהיר ביותר, לכן אם יש לסוכן של דנה אפשרות לנצח בתור הנוכחי והיא לא נבחרת ייתכן וקיימת אפשרות לניצחון בהמשך המשחק והאלגוריתם בוחר בה.

שאלה 3

אלגוריתמי any-time הם אלגוריתמים שיכולים לשפר את ביצועיהם בהינתן משך ריצה ארוך יותר. במקרה הנוכחי, הגבלת זמן במקום הגבלת עומק תתבטא בהרצת מינימקס עד לעומק L=1 ובסיום כל איטרציה של האלגוריתם נגדיל את L ב-1 ונריץ מחדש את RB-Minimax עם L גדול 1.

אלגוריתם any-time נוסף שנלמד בקרוס הינו ID-DFS שמריץ DFS לעומק מוגבל לזמן ריצה מוגבל.

שאלה 4

מימוש בקוד.

שאלה 5

במשחק מרובה משתתפים (שהוא משחק סכום אפס) יש יותר שחקנים שרוצים להרע לי, כלומר יותר צמתי OR בעץ, כתלות במספר השחקנים (NUM\_OF\_PLAYERS) ולכן נעשה התאמה לפסודו-קוד מהתרגול:

Function RB-Minimax-Multiple-Players (State, Agent, depth, turn\_count = 0):

If G(State) OR depth=0 then return h(State,Agent)

ChildrenSucc(State)

if turn\_count % NUM\_OF\_PLAYERS == 0: // the turn is mine, choose max

CurMax

Loop for c in Children:

v RB-Minimax-Multiple-Players(c, Agent, depth-1, turn\_count+1)

CurMaxMax(v, CurMax)

Return CurMax

else: //the turn is not mine, choose min

CurMin

Loop for c in children:

v RB-Minimax-Multiple-Players(c, Agent, depth-1, turn\_count+1)

CurMinMin(c, CurMin)

return CurMin