ה א ו נ י ב ר ס י ט ה ה פ ת ו ח ה

20551

מבוא לבינה מלאכותית

חוברת הקורס סתיו 2019א

כתבה: אילנה בס

אוקטובר 2018 – סמסטר סתיו – תשע״ט

פנימי – לא להפצה.

תוכן העניינים

אל הסטודנט,

אנו מקדמים את פניך בברכה עם הצטרפותך אל הלומדים בקורס "מבוא לבינה מלאכותית".

בחוברת זו תמצא את לוח הזמנים של הקורס, תנאים לקבלת נקודות זכות וחלק מהמטלות.

לקורס קיים אתר באינטרנט בו תמצאו חומרי למידה נוספים, אותם מפרסם/מת מרכז/ת ההוראה.
בנוסף, האתר מהווה עבורכם ערוץ תקשורת עם צוות ההוראה ועם סטודנטים אחרים בקורס.
פרטים על למידה מתוקשבת ואתר הקורס, תמצאו באתר שה"ם בכתובת:
http://telem.openu.ac.il
מידע על שירותי ספרייה ומקורות מידע שהאוניברסיטה מעמידה לרשותכם, תמצאו באתר הספריה באינטרנט www.openu.ac.il/Library.

צוות הקורס ישמח לעמוד לרשותך בכל שאלה שתתעורר.
ניתן לפנות למנחים בשעות ההנחייה הטלפוניות שלהם או אלי בכל יום ד' בשעות 11:00-13:00 בטלפון 09-7781239. כמו כן ניתן לפנות אלי ב- e-mail כתובתי: ilana@openu.ac.il

לתשומת לב הסטודנטים הלומדים בחו"ל:

למרות הריחוק הפיסי הגדול, נשתדל לשמור אתכם על קשרים הדוקים ולעמוד לרשותכם ככל האפשר.
הפרטים החיוניים על הקורס נכללים בחוברת הקורס וכן באתר הקורס.
מומלץ מאד להשתמש באתר הקורס ובכל אמצעי העזר שבו וכמובן לפנות אלינו במידת הצורך.

אני מאחלת לך לימוד פורה ומהנה.

ב ב ר כ ה,

אילנה בס
מרכזת ההוראה בקורס

1. לוח זמנים ופעילויות (20551 /א2019)

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
1	19.10.2018-14.10.2018	פרקים 1,2		
2	26.10.2018-21.10.2018	פרק 3	מפגש 1	
3	2.11.2018-28.10.2018	פרק 4		ממ"ן 11 (להרצה) 30.10.2018
4	9.11.2018-4.11.2018	פרק 5	מפגש 2	
5	16.11.2018-11.11.2018	פרק 6		ממ"ן 12 (תיאורטי) 13.11.2018
6	23.11.2018-18.11.2018	פרק 7	מפגש 3	ממ"ן 13 (להרצה) 24.11.2018
7	30.11.2018-25.11.2018	פרק 8		
8	7.12.2018-2.12.2018 (ב-ו חנוכה)	פרק 9	מפגש 4	ממ"ן 14 (תיאורטי) 8.12.2018

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

לוח זמנים ופעילויות - המשך

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
9	14.12.2018-9.12.2018 (א-ב חנוכה)	פרק 10		
10	21.12.2018-16.12.2018	פרק 13	מפגש 5	ממ"ן 15 (להרצה) 22.12.2018
11	28.12.2018-23.12.2018	פרק 14		
12	4.1.2019-30.12.2018	פרק 17	מפגש 6	ממ"ן 16 (תיאורטי) 5.1.2019
13	11.1.2019-6.1.2019	פרק 18		
14	18.1.2019-13.1.2019	חזרה	מפגש 7	ממ"ן 17 (תיאורטי) 23.1.2019

תאריך אחרון למשלוח ממ"ן 18: 23.2.2019

מועדי בחינות הגמר יפורסמו בנפרד

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

2. תיאור המטלות

קרא היטב עמודים אלו לפני שתתחיל לענות על השאלות

בקורס זה 8 מטלות, 4 מטלות תיאורטיות ו-4 מטלות להרצה.
פתרון המטלות הוא חלק בלתי נפרד מלימוד הקורס, שכן הבנה מעמיקה של חומר הלימוד דורשת תרגול רב.
יש להגיש לפחות 2 מטלות מבין המטלות התיאורטיות (12,14,16,17) (במשקל כולל של 5 נק׳ לפחות) ו-2 מטלות לפחות מבין מטלות הרצה (11,13,15,18) במשקל כולל של 10 נק׳ לפחות .
אם שאלה מסוימת בממ״ן אינה ברורה לך, אל תהסס להתקשר אל המנחה (בשעות הייעוץ הטלפוני שלו) או להיעזר בקבוצת הדיון של הקורס.
להלן תמצא הסבר על אופן הפתרון הנדרש וכיצד לשלוח את המטלה למנחה.

2.1 מבנה המטלות וצורת הגשתן

בקורס ישנן כאמור מטלות משני סוגים:

מטלות רגילות:

מטלה כזו מורכבת מכמה שאלות. בראש כל שאלה מצוין משקלה היחסי בקביעת ציון המטלה. פתרון השאלות במטלה כזו אינו דורש הרצת תכניות במחשב. הן נועדו לבדוק את הבנתך בחומר הלימוד.
את הפתרונות למטלה כזו יש לכתוב בצורה ברורה ומסודרת.

מטלות הרצה:

במטלות אלה עליך לכתוב תכניות ולהריץ אותן במחשב.

את התכניות במטלות 11,13,15 יש לכתוב בשפת Python.

את התכנית במטלה 18 ניתן לכתוב ב- Python או ב-C/C++ או ב-Java.

<u>תיעוד:</u>
בכל תכנית הוסף תיעוד בגוף התכנית המסביר מהו תפקידו של כל משתנה, מה מבצעת כל שגרה וכל הסבר נוסף החשוב להבנת מהלך פעולתה של התכנית. יש לתת שמות משמעותיים למשתנים ולשגרות המופיעים בתכניות.
יש להקפיד על קריאות ובהירות תוך שימוש בהיסח (אינדטציה) מסודרת ואחידה.

<u>במטלת הרצה עליך לשלוח למנחה:</u>

א. את התכנית לאחר שבדקת שהיא מבצעת את הנדרש ממנה ללא טעויות.

ב. יש להגיש את קובץ המקור של התכנית (source code).

ג. יש לצרף מספר דוגמאות ריצה של התכנית. תכנית שתישלח ללא דוגמאות ריצה (דוגמאות קלט והפלט שהופק עבורן) לא תיבדק!

תכניות שתוגשנה בכתב-יד או ללא תיעוד או ללא קובץ המקור - לא תבדקנה!

2.2 חומר הלימוד הדרוש לפתרון המטלות

בטבלה שלהלן תמצא מהו חומר הלימוד הנדרש (לפי פרקי הספר) לפתרון כל אחת מהמטלות.

שים לב!
אין להשתמש לפתרון המטלות בידע הנרכש בפרקי לימוד מתקדמים יותר מהפרקים בהם עוסקת המטלה.

מטלה	חומר הלימוד הנדרש לפתרונה
ממ״ן 11	פרקים 1-4
ממ״ן 12	פרקים 1-5
ממ״ן 13	פרקים 1-5
ממ״ן 14	פרקים 6-9
ממ״ן 15	פרקים 1-10
ממ״ן 16	פרק 10, פרקים 13-14
ממ״ן 17	פרקים 17-18
ממ״ן 18	פרקים 1-18

2.3 ניקוד המטלות

המשקל הכולל של ממ״נים 11-18 הוא 30 נקודות. עליך לצבור לפחות 15 נקודות.

ללא עמידה בדרישות המטלות לא ניתן יהיה לגשת לבחינת הגמר

הכנת המטלות 11-18 חייבת להיעשות ע״י כל סטודנט בנפרד.
מטלות שלא יבוצעו באופן עצמאי – ייפסלו!!!

להלן פירוט הניקוד לכל מטלה:

ממ״ן	ניקוד
11	6
12	3
13	4
14	3
15	5
16	2
17	2
18	5

לתשומת לבכם:

מדיניות קורס זה היא לאשר הזנת ציון אפס במטלות שלא הוגשו כנדרש בקורס.
סטודנטים אשר לא הגישו את מכסת המטלות המינימאלית לעמידה בדרישות הקורס ולקבלת זכאות להיבחן, ומבקשים שמטלות חסרות יוזנו בציון אפס, יפנו למוקד הפניות והמידע בטלפון **09-7782222** או **יעדכנו בעצמם** באתר שאילתא http://www.openu.ac.il/sheilta

קורסים ⇦ ציוני מטלות ובחינות ⇦ הזנת ציון 0 למטלות רשות שלא הוגשו.

יש לקחת בחשבון כי מטלות אשר יוזן להן ציון אפס ישוקללו בחישוב הציון הסופי ובכך יורידו ציון זה ולא ניתן יהיה להמירן במטלות חלופיות במועד מאוחר יותר. על כן קיימת אפשרות שסטודנט אשר יעבור את הבחינה בהצלחה ייכשל בקורס (כשהממוצע המשוקלל של המטלות והבחינה יהיה נמוך מ- 60).

כלל זה איננו חל על מטלות חובה או על מטלות שנקבע עבורן ציון מינימום.

לתשומת לבכם!

כדי לעודדכם להגיש לבדיקה מספר רב של מטלות הנהגנו את ההקלה שלהלן:

אם הגשתם מטלות מעל למשקל המינימלי הנדרש בקורס, **המטלות** בציון הנמוך ביותר, שציוניהן נמוכים מציון הבחינה **(עד שתי מטלות)**, לא יילקחו בחשבון בעת שקלול הציון הסופי.

זאת בתנאי שמטלות אלה **אינן חלק מדרישות החובה בקורס** ושהמשקל הצבור של המטלות האחרות שהוגשו, מגיע למינימום הנדרש.

זכרו! ציון סופי מחושב רק לסטודנטים שעברו את בחינת הגמר בציון 60 ומעלה והגישו מטלות כנדרש באותו קורס.

3. התנאים לקבלת נקודות זכות בקורס

א. הגשת 2 מטלות לפחות מבין המטלות התיאורטיות (12,14,16,17) תוך צבירת 5 נק' לפחות.
ב. יש להגיש לפחות 2 מתוך המטלות להרצה (11,13,15,18) וצבירת 10 נק' לפחות.
ג. ציון 60 לפחות בכל מטלת הרצה.
ד. ציון 60 לפחות בבחינת הגמר.
ה. ציון סופי בקורס 60 לפחות.

מטלת מנחה (ממ"ן) 11 - להרצה

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: שפת Python ופרקים 1-3

מספר השאלות: 8 **משקל המטלה:** 6

סמסטר: 2019א **מועד אחרון להגשה:** 30.10.18

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

מטלות הריצה בקורס ייכתבו בשפת **Python(2.7)**, שפת תכנות מונחה עצמים.

אנו לא מניחים כלל שיש לכם נסיון בתכנות בשפה זו, אך מצפים שתלמדו את הבסיס שלה די מהר, (במיוחד לאור הכרותכם ונסיונכם עם שפת Java).

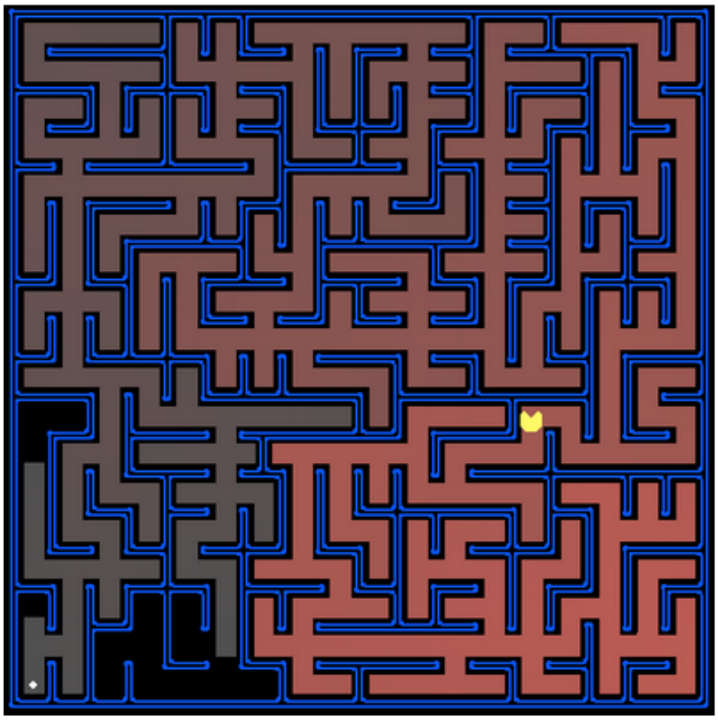
מצורף כאן קישור למדריך בסיסי ב-Python(2.7) ובו משולבים תרגילים שעליכם לפתור כדי להכיר השפה. אין צורך להגישם. הקובץ שבקישור מתייחס גם לעבודה בסביבת Unix וניתן להתעלם מההתייחסויות ל-Unix.

שתי מטלות הריצה הראשונות פותחו באוניברסיטת ברקלי על-ידי John DeNero ו-Dan Klein.

מדריך לPython בעברית ניתן למצוא כאן.

מדריך מפורט ניתן למצוא כאן.

חיפוש ב-Pac-Man



All those colored walls,
Mazes give Pacman the blues,
So teach him to search.

הקדמה

במטלה זו, סוכן פקמן שלכם ימצא מסלולים בעולם המבוך שלו, הן כדי להגיע למיקום מסויים והן כדי ללקט מזון בצורה יעילה. עליכם לכתוב אלגוריתמי חיפוש כלליים ולממשם לתרחישים של פקמן.

הקוד עבור מטלה זו מורכב ממספר קבצי Python. לצורך ביצוע המטלה יהיה עליכם לקרוא ולהבין את חלקם ומהאחרים תוכלו להתעלם. את כל הקוד והקבצים הנלווים ניתן להוריד כקובץ zip.

הקבצים שתערכו ולתוכם תוסיפו את הקוד שלכם:

search.py	כל אלגוריתמי החיפוש שכם יימצאו שם
searchAgents.py	כל סוכני החיפוש שלכם יימצאו שם

קבצים שכדאי לקרוא:

pacman.py	הקובץ העיקרי שמריץ משחקי פקמן. הקובץ מתאר את הטיפוס GameState שתשתמשו בו במטלה זו
game.py	הלוגיקה שמאחורי אופן פעולת עולם הפקמן. קובץ זה מתאר מספר טיפוסי עזר כגון AgentState , Agent, Direction, Grid .
util.py	מבני נתונים שימושיים למימוש אלגוריתמי חיפוש

<u>קבצים שאין צורך להתעמק בקוד שלהם</u>:

GraphicsDisplay.py	גרפיקה עבור פקמן
GraphicsUtils.py	תמיכה לגרפיקה של פקמן
TextDisplay.py	גרפיקת אסקי לפקמן
GhostAgents.py	סוכנים לבקרה על רוחות רפאים
KeyboardAgents.py	ממשקי מקלדת לבקרה על פקמן
Layout.py	קוד לקריאת קבצי layout ואחסון תוכנם

<u>עליכם להגיש</u>

יש למלא חלקים חסרים בקוד search.py ו-searchAgents.py במהלך ביצוע המטלה. עליכם לשלוח למנחה את שני הקבצים הללו (בלבד) וכן את README.txt (שם, ת.ז ותיעוד) כקובץ zip. כלומר יש לשלוח בדיוק קובץ אחד.
הנכם מתבקשים לא לשנות שמות של פונקציות או מחלקות הנתונות בקוד.

ברוכים הבאים לפקמן

לאחר שהורדתם את הקוד (search.zip) , עשיתם לו unzip, תוכלו לשחק משחק פקמן. לצורך כך הקלידו בשורת הפקודה (command line):

```
python  pacman.py
```

שימו לב: אם קיבלתם הודעת שגיאה לגבי python-tk, התקינו python-tk או ראו הוראות מפורטות יותר בעמוד זה.

פקמן חי בעולם כחול בוהק של מסדרונות פתלתלים וממתקים עגולים טעימים. ניווט בעולם זה ביעילות תיהיה המשימה הראשונה של פקמן.
הסוכן הפשוט ביותר בקובץ searchAgents.py נקרא GoWestAgent, שתמיד זז מערבה (סוכן של תגובה פשוטה). סוכן זה יכול לעתים לנצח:

```
python  pacman.py --layout  testMaze --pacman  GoWestAgent
```

אבל, העניינים מסתבכים עבור סוכן זה כאשר עליו להסתובב:

```
python  pacman.py --layout tinyMaze --pacman GoWestAgent
```

אם פקמן נתקע, תוכלו לצאת מהמשחק על ידי הקשת CTRL-c.

במהרה הסוכן שלכם יפתור לא רק tinyMaze אלא כל מבוך שתרצו. שימו לב ש- pacman.py תומך במספר אופציות. כל אחת מהן ניתנת לביטוי באריכות (למשל, layout--) או בקצרה (למשל, 1-). תוכלו לראות את רשימת כל האפשרויות וערכי ברירת המחדל שלהן על ידי:

```
python  pacman.py  -h
```

כמו כן, כל הפקודות המופיעות במטלה זו, מופיעות גם ב-commands.txt, לצורך נוחיות העתקתן.

שימוש באלגוריתמי חיפוש למציאת נקודת מזון קבועה

בקובץ searchAgents.py, תמצאו searchAgent ממומש במלואו, המתכנן מסלול דרך עולם פקמן ולאחר מכן מבצע מסלול זה צעד אחר צעד. אלגוריתמי החיפוש עבור התכנית (plan) אינם ממומשים. משימה זו מוטלת עליכם. במהלך פתרון השאלות שיופיעו להלן, תצטרכו כנראה להתייחס לרשימת תיאור סוגי האובייקטים שבקוד המופיעה בסוף מטלה זו.

תחילה, בידקו שה-searchAgent פועל היטב על ידי כך שתריצו:

```
python  pacman.py  -l  tinyMaze  -p  SearchAgent  -a  fn=tinyMazeSearch
```

הפקודה שלעיל מורה ל-searchAgent להשתמש ב-tinyMazeSearch (הממומש בקובץ search.py) כאלגוריתם החיפוש שלו. פקמן אמור לנווט במבוך בהצלחה.

כעת ניתן לכתוב פונקציות חיפוש כדי לעזור לפקמן לתכנן מסלולים. הפסאודו-קוד לאלגוריתמי החיפוש נמצא בספר הלימוד. זכרו כי צומת במרחב החיפוש צריך להכיל בנוסף למצב עצמו, את כל המידע הדרוש לבנייה מחדש של המסלול (plan) המוביל למצב זה.

הערה חשובה: כל פונקציות החיפוש שלכם צריכות להחזיר רשימת *פעולות* שתוביל את הסוכן מהמצב ההתחלתי אל מצב המטרה. פעולות אלה צריכות להיות (כולן) מהלכים חוקיים (כיוונים מותרים, לא להיכנס בתוך קירות).

רמזים:

1. האלגוריתמים דומים אחד לשני. אלגוריתמים עבור DFS, BFS, A*,Uniform-Cost search נבדלים זה מזה רק בפרטי אופן ניהול החזית. כלומר, לאחר שתכתבו את DFS כנדרש, יהיה לכם קל מאד לכתוב את יתר האלגוריתמים. אכן, אחד המימושים האפשריים (אינו חובה) דורש יצירת מתודה לחיפוש כללי שמקבלת את מבנה הנתונים לאחזקת המצבים הממתינים להיסרק.
2. בדקו את טיפוסי Queue, Stack ו-PriorityQueue הנתונים לכם בקובץ util.py!

שאלה 1 (10%)

ממשו את אלגוריתם חיפוש לעומק (DFS) בפונקציה depthFirstSearch אשר בקובץ search.py.
כדי שהאלגוריתם שלכם יהיה שלם, כתבו את גירסת חיפוש-גרף של חיפוש לעומק, המונעת פיתוח של מצבים שנסרקו כבר. (ראו סעיף 3.3 בספר הלימוד).
הקוד שתכתבו אמור למצוא פתרון עבור:

```
python pacman.py -l tinyMaze -p SearchAgent

python pacman.py -l mediumMaze -p SearchAgent

python pacman.py -l bigMaze -z .5 -p SearchAgent
```

הלוח של פקמן יראה את המצבים שנסרקו ואת הסדר על פיו נסרקו (אדום בהיר יותר פירושו שנסרקו מוקדם יותר). האם סדר הסריקה תואם לסדר שהייתם מצפים?
רמז: אם תשתמשו ב-Stack כבמבנה הנתונים שלכם, הפתרון שימצא אלגוריתם חיפוש לעומק שכתבתם עבור mediumMaze יהיה באורך 130 (בהנחה שהוספתם עוקבים לחזית על פי הסדר הנתון על ידי getSuccessors ; אתם אמורים לקבל 244 אם הוספתם אותם בסדר ההפוך).

האם זה פתרון זול ביותר? נמקו.

שאלה 2 (10%)

ממשו אלגוריתם חיפוש לרוחב (BFS) בפונקציית breadthFirstSearch שבקובץ search.py. שוב, עליכם לכתוב אלגוריתם חיפוש-גרף המונע פיתוח של מצבים שנסרקו כבר. בדקו את הקוד שלכם באותה הדרך שבדקתם עבור חיפוש לעומק.

```
python pacman.py -l mediumMaze -p SearchAgent -a fn=bfs

python pacman.py -l bigMaze -p SearchAgent -a fn=bfs -z .5
```

האם אלגוריתם חיפוש לרוחב מוצא פתרון זול ביותר? נמקו.

רמז: אם נראה לכם שפקמן זז לאט מדי, נסו את האופציה --frameTime 0.
שימו לב: אם כתבתם את קוד החיפוש באופן כללי, הוא אמור לעבוד באותה מידה גם עבור בעיית 8-puzzle, ללא כל שינוי.

```
python eightpuzzle.py
```

שינוי פונקצית המחיר

BFS ימצא את המסלול למטרה בעל מספר פעולות מינימלי, אך יתכן שנרצה למצוא מסלולים שהם "טובים יותר" במובנים אחרים.

נתייחס ל-mediumDottedMaze ול-mediumScaryMaze. על ידי שינוי פונקצית המחיר, אנו יכולים לעודד את פקמן למצוא מסלולים אחרים.

למשל, אנו יכולים לתת מחיר גבוה לצעדים באיזורים בהם יש הרבה רוחות רפאים או לתת מחיר נמוך לצעדים באיזורים עתירי מזון וסוכן פקמן רציונלי אמור להתאים את התנהגותו לשינויים אלה.

שאלה 3 (10%)

ממשו אלגוריתם לחיפוש-גרף מונחה-מחיר בפונקציה uniformCostSearch אשר בקובץ search.py. כדאי לעבור על util.py כדי לראות מבני נתונים שיכולים להיות לכם לעזר במימוש האלגוריתם. כל הסוכנים שלהלן הינם סוכני חיפוש מונחה מחיר וההבדל ביניהם הוא בפונקציית המחיר בה כל אחד משתמש (הסוכנים ופונקציות המחיר נתונים לכם).

```
python pacman.py -l mediumMaze -p SearchAgent -a fn=ucs
python pacman.py -l mediumDottedMaze -p StayEastSearchAgent
python pacman.py -l mediumScaryMaze -p StayWestSearchAgent
```

שימו לב: אתם אמורים לקבל מחירי מסלול מאד נמוכים ומאד גבוהים עבור StayEastSearchAgent ו-StayWestSearchAgent בהתאמה, עקב פונקציות המחיר האקספוננציאליות שלהם (פרטים בקובץ searchAgents.py).

חיפוש A*

שאלה 4 (15%)

ממשו אלגוריתם חיפוש-גרף A* בפונקציה aStarSearch (הריקה) אשר בקובץ search.py. A* מקבל כפרמטר פונקציה יוריסטית. יוריסטיקה מקבלת שני פרמטרים: מצב בבעיית חיפוש (הפרמטר הראשי) והבעיה עצמה. דוגמה טריוויאלית היא הפונקציה היוריסטית nullHeuristic שבקובץ search.py. אתם יכולים לבדוק את מימוש A* על הבעיה המקורית של מציאת מסלול למיקום קבוע במבוך תוך שימוש ביוריסטיקת מרחק מנהטן (הממומשת כבר כ-manhattanHeuristic בקובץ searchAgents.py).

```
python pacman.py -l bigMaze -z .5 -p SearchAgent -a fn=astar,
heuristic=manhattanHeuristic
```

תוכלו לראות ש*A מוצא את הפתרון האופטימלי קצת מהר יותר מחיפוש מונחה-מחיר (בסביבות 549 לעומת 629 צמתי חיפוש מפותחים במימוש שלנו אבל כשיש עדיפויות זהות, הבחירות שהאלגוריתם יבצע עלולות לשנות מעט את מספרם של המצבים.

מה קורה ב-openMaze לאסטרטגיות החיפוש השונות?

מציאת כל הפינות

היתרון האמיתי של *A יבוא לידי ביטוי בבעית חיפוש מאתגרת יותר. ננסח כעת בעיה חדשה ונתכנן יוריסטיקה עבורה.
נתון מבוך עם ארבע פינות ובכל אחת מהן יש נקודה. בבעית החיפוש החדשה יש למצוא את המסלול הקצר ביותר במבוך העובר דרך ארבע הפינות. (בין אם במבוך יש שם מזון או לא).שימו לב לכך שעבור מבוכים מסויימים כגון tiny Corners, המסלול הקצר ביותר לא תמיד עובר תחילה דרך נקודת המזון הקרובה ביותר! *רמז*: אורך המסלול הקצר ביותר דרך tinyCorners הוא 28 צעדים.

שאלה 5 (10%)

ממשו את בעיית החיפוש CornersProblem בקובץ searchAgents.py. עליכם לבחור יצוג למצבים אשר יקודד את כל המידע הדרוש כדי לדעת האם נסרקו כל ארבע הפינות. כעת סוכן החיפוש שלכם יפתור:

```
python pacman.py -l tinyCorners -p SearchAgent -a fn=bfs,prob=CornersProblem
```

```
python pacman.py -l mediumCorners -p SearchAgent -a fn=bfs,prob=CornersProblem
```

כדי לקבל ניקוד מלא בשאלה זו, עליכם להגדיר את ייצוג המצב כך שלא יקודד מידע לא רלבנטי (כגון מיקום רוחות הרפאים, היכן יש מזון נוסף וכד'). במיוחד, אל תשתמשו ב-GameState של פקמן כבמצב של מרחב החיפוש. אחרת, הקוד שלכם יהיה איטי מאד מאד וגם שגוי).

רמז: החלקים היחידים במצב במשחק שאליהם עליכם להתייחס במימושכם הם המיקום ההתחלתי של פקמן ומיקומן של ארבע הפינות.

המימוש של breadthFirstSearch מפתח פחות מ-2000 צמתים ב- medium Corners. ואולם, יוריסטיקה (שמשתמשים בה ב-*A) יכולה להפחית את החיפוש הנדרש.

שאלה 6 (15%)

ממשו יוריסטיקה עבור בעיית החיפוש CornersProblem ב- CornersHeuristic.

אופן מתן הציון לשאלה זו: לא ינתן ניקוד עבור פונקציה יוריסטית שאינה קבילה. תינתן נקודה אחת לכל פונקציה יוריסטית קבילה. נקודה אחת אם יפותחו פחות מ-1200 צמתים. נקודה אחת אם יפותחו פחות מ-900 צמתים. ניקוד מלא אם יפותחו פחות מ-800 צמתים.

```
python pacman.py -l mediumCorners -p AStarCornersAgent -z 0.5
```

רמז: זכרו, פונקציות יוריסטיות מחזירות רק מספרים, אשר צריכים להוות חסמים תחתונים על מחיר המסלול (האמיתי) הקצר ביותר למטרה הקרובה ביותר, כדי שהיוריסטיקות תהיינה קבילות.

שימו לב: AStarCornerAgent הוא קיצור עבור

```
-p SearchAgent -a fn=aStarSearch,prob=
Cornersproblem,

heuristic=CornersHeuristic
```

"אכילת" כל הנקודות

נפתור כעת בעית חיפוש קשה: "אכילת" כל המזון של פקמן במספר צעדים קטן ככל האפשר. לצורך כך נזדקק להגדרה חדשה עבור בעית החיפוש המבטאת את בעית "אכילת" כל המזון.

FoodSearchProblem בקובץ searchAgents.py (הממומשת עבורכם). נגדיר פתרון כמסלול האוסף את כל המזון שבעולם הפקמן. לצורך ביצוע מטלה זו, הפתרונות אינם מביאים בחשבון רוחות רפאים כלשהן או חטיפי (טבליות) אנרגיה ; הפתרונות תלויים רק במיקום הקירות, מזון רגיל ובפקמן. (כמובן שרוחות רפאים יכולות להרוס את ביצוע הפתרון! בכך נטפל במטלת הריצה הבאה). אם כתבתם נכון את מתודות החיפוש הכלליות, *A ללא יוריסטיקה (השקול לחיפוש מונחה-מחיר) יוכל בקלות למצוא פתרון אופטימלי ל-test Search ללא שינוי בקוד מצדכם (מחיר כולל של 7).

```
python pacman.py -l testSearch -p AStarFoodSearchAgent
```

שימו לב: AStarFoodSearchAgent הוא קיצור עבור

```
-p SearchAgent -a fn=aStar,prob=FoodSearchProblem, heuristic=FoodHeuristic
```

אתם אמורים לגלות שחיפוש מונחה מחיר מתחיל להיות איטי אפילו ה-tiny Search הפשוט. לצורך השוואה, המימוש שלנו מוצא מסלול באורך 27 ב-2.5 שניות, לאחר פיתוח של 4902 צמתים במרחב החיפוש.

שאלה 7 (20% + בונוס 5%*)

השלימו את foodHeuristic בקובץ searchAgents.py עם פונקציה יוריסטית עבור FoodSearchProblem. נסו את הסוכן שלכם על לוח trickySearch:

```
python pacman.py -l trickySearch -p AStarFoodSearchAgent
```

סוכן החיפוש המונחה-מחיר שלנו מוצא את הפתרון האופטימלי ב-13 שניות לערך, תוך שהוא סורק 16,000 צמתים. אם היוריסטיקה שלכם קבילה, תקבלו את הניקוד הבא, התלוי במספר הצמתים שפותחו בעזרתה.

מספר צמתים קטן מ-	ניקוד
15000	5%
12000	10%
9000	15%
7000	5% - בונוס*

שימו לב: אם היוריסטיקה שלכם אינה קבילה, לא תקבלו ניקוד כלל. חשבו על ענין הקבילות בזהירות, שכן יוריסטיקה לא קבילה יכולה להצליח ליצור חיפושים מהירים ואף מסלולים אופטימליים. האם תוכלו לפתור medium Search בזמן קצר? אם כן, הדבר מרשים ביותר או... שהיוריסטיקה שלכם אינה קבילה.

*הציון הכולל במטלה לא יעלה על 100.

קבילות לעומת עקביות:

טכנית, קבילות אינה מספקת כדי להבטיח נכונות בחיפוש-גרף. יש צורך בקיום תנאי חזק יותר, של עקביות. כדי שיוריסטיקה תהיה עקבית, צריך להתקיים שאם מחירה של פעולה הוא c, אזי ביצוע פעולה זו יכול לגרום רק לירידה ביוריסטיקה של c לכל היותר. אם היוריסטיקה שלכם אינה רק קבילה אלא גם עקבית, תקבלו 5% נוספים על תשובתכם לשאלה 7.

כמעט תמיד, יוריסטיקות קבילות הן גם עקביות, במיוחד אם הן נגזרו מבעיות מוחלשות. לכן קל יותר, כפי הנראה, להתחיל במחשבה על יוריסטיקות קבילות. לאחר שתהיה לכם יוריסטיקה קבילה שעובדת היטב, תוכלו לבדוק האם היא אכן גם עקבית. לעתים ניתן לגלות חוסר עקביות על ידי בדיקה האם ערכי ה-f של הפתרונות המוחזרים שלכם אינם יורדים. יתר על כן, אם חיפוש מונחה מחיר ו-A* מחזירים מסלולים באורך שונה, היוריסטיקה שלכם אינה עקבית.

חיפוש תת-מיטבי (Suboptimal Search)

לעתים, גם עם A* ופונקציה יוריסטית טובה, מציאת מסלול אופטימלי דרך כל הנקודות הינה משימה קשה. במקרים כאלה, נרצה עדיין למצוא מסלול טוב, בזמן קצר.

עליכם לכתוב כעת קוד לסוכן פקמן אשר תמיד אוכל באופן חמדני את הנקודה הקרובה ביותר.

ClosestDotSearchAgent ממומש עבורכם בקובץ searchAgents.py, אך חסרה לו הפונקציה העיקרית המוצאת מסלול לנקודה הקרובה ביותר.

שאלה 8 (10%)

ממשו את הפונקציה findPathToClosestDot שבקובץ searchAgents.py. הסוכן שלנו פותר מבוך זה (באופן תת-מיטבי) בפחות משניה אחת ובמחיר מסלול של 350:

```
python pacman.py -l bigSearch -p ClosestDotSearchAgent -z .5
```

רמז: הדרך המהירה ביותר להשלים את findPathToClosestDot היא להשלים את החסר ב-AnyFoodSearchProblem (שחסר בו מבחן המטרה). לאחר מכן, פתרו בעיה זו עם פונקציית חיפוש מתאימה. הפתרון אמור להיות מאד קצר!

ה-ClosestDotSearchAgent שלכם לא תמיד ימצא את המסלול האפשרי הקצר ביותר דרך המבוך. אך אם תנסו, תוכלו לשפרו.

לנוחיותכם מרוכזים ברשימה שלהלן **סוגי האובייקטים העיקריים המופיעים בקוד הבסיסי הקשורים לבעיות חיפוש:**

`SearchProblem (search.py)`
A SearchProblem is an abstract object that represents the state space, successor function, costs, and goal state of a problem. You will interact with any SearchProblem only through the methods defined at the top of `search.py`

`PositionSearchProblem (searchAgents.py)`
A specific type of SearchProblem that you will be working with --- it corresponds to searching for single pellet in a maze.

`CornersProblem (searchAgents.py)`
A specific type of SearchProblem that you will define --- it corresponds to searching for a path through all four corners of a maze.

`FoodSearchProblem (searchAgents.py)`
A specific type of SearchProblem that you will be working with --- it corresponds to searching for way to eat all the pellets in a maze.

Search Function
A search function is a function which takes an instance of SearchProblem as a parameter, runs some algorithm, and returns a sequence of actions that lead to a goal. Example of search functions are `depthFirstSearch` and `breadthFirstSearch`, which you have to write. You are provided `tinyMazeSearch` which is a very bad search function that only works correctly on `tinyMaze`

`SearchAgent`
`SearchAgent` is a class which implements an Agent (an object that interacts with the world) and does its planning through a search function. The `SearchAgent` first uses the search function provided to make a plan of actions to take to reach the goal state, and then executes the actions one at a time.

מטלת מנחה (ממ"ן) 12

הקורס: **20551 – מבוא לבינה מלאכותית**

חומר הלימוד למטלה: פרקים 1-5

מספר השאלות: 5 **משקל המטלה:** 3 נקודות

סמסטר: 2019א **מועד אחרון להגשה:** 13.11.2018

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

השאלות בעמודים הבאים

שאלה 1 (15 נק': 5 נק' לכל סעיף)

בעית המיסיונרים והקניבלים:
שלושה מיסיונרים ושלושה קניבלים נמצאים בגדה השמאלית של נהר.
בגדה הזו מצויה סירה, שיכולה לשאת אדם אחד או שני אנשים.
מעונינים להעביר את ששת האנשים לגדה הימנית של הנהר בעזרת הסירה.
מסיבות מובנות, אין לאפשר, אפילו לרגע אחד, מצב שבו מספר הקניבלים גדול ממספר המיסיונרים באחת הגדות של הנהר.
העברת הסירה מגדה לגדה איננה יכולה להתבצע בלי שיהיה בה לפחות אדם אחד.

ננסח את הבעיה כבעית חיפוש עם מרחב מצבים $S=\{(c,m,b)\}$ כך ש:

- C מציין את מספר הקניבלים בגדה השמאלית של הנהר
- M מציין את מספר המיסיונרים בגדה השמאלית של הנהר
- B מציין את מיקום הסירה (0 עבור הגדה השמאלית, 1 עבור הגדה הימנית)

המצב ההתחלתי: $\{(3,3,0)\}$

מצב המטרה: $\{(0,0,B)\}$

נתונה הגדרה חלקית של מודל המעברים:

- $(C,M,0 \mid (M\geq 2) \wedge (M-2\geq C \vee M=2)) \rightarrow (C,M-2,1)$
- $(C,M,1 \mid (C\leq 2) \wedge (M\leq 2)) \rightarrow (C+1,M+1,0)$

א. השלימו את ניסוח בעית החיפוש.

ב. מצאו פתרון אופטימלי לבעיה (אופטימלי במובן של מספר העברות הסירה מגדה לגדה) תוך שימוש באלגוריתם חיפוש חסר ידע.
הציגו סדרת פעולות, מן הפעולות שהצעתם בסעיף א', שמעבירה מן המצב ההתחלתי למצב מטרה.
נסחו את הפעולות והמצבים במינוחים של תשובתכם לסעיף א'. (אינכם צריכים להסביר כיצד הגעתם לפתרון).

ג. הציעו יוריסטיקה שתגרום לחיפוש best-first חמדני למצוא את הפתרון הטוב ביותר.
שימו לב, מדובר על חיפוש שלא בודק האם כבר ביקרנו במצב שאליו עוברים (אפילו לא במקרה שמחזיר אותנו למצב שממנו הגענו למצב שבו אנו נמצאים כעת). לכן עליכם להקפיד על כך שהיוריסטיקה שאתם מציעים לא תגרום ללולאות אינסופיות. (זכרו שפונקציה יוריסטית היא פונקציה ממצב למספר.)

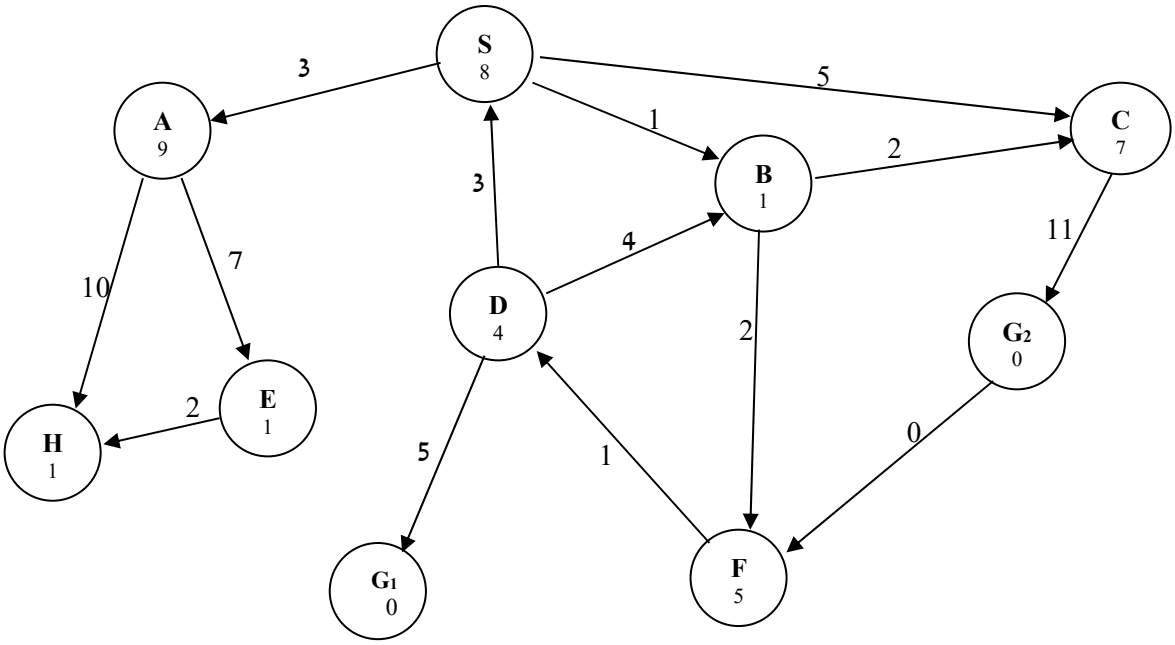
המשך המטלה בעמודים הבאים

שאלה 2 (21 נק' : 15 נק' לסעיף א'; 3 נק' לכ"א מסעיפים ב'-ג')

א. נתון גרף מרחב מצבים שלהן.

S הוא המצב (צומת) ההתחלתי ו- G1, G2 הם מצבי סיום (מקיימים את מבחן המטרה).

מחירי המעברים בין המצבים רשומים על הקשתות; הערכים היוריסטיים (ערכי הפונקציה h=עלות משוערת למטרה) רשומים בצמתים.



עבור כל אחת מאסטרטגיות החיפוש שלהלן, כתבו לאילו מצבי מטרה ניתן להגיע (אם בכלל) על-ידי האלגוריתם ורשמו על-פי הסדר את הצמתים המוצאים מהחזית (frontier) במהלך ריצת האלגוריתם.

הניחו כי כל האלגוריתמים משתמשים ב-explored set (למעט עבור Iterative Deepeninig) ובמידה ולשני צמתים או יותר יש עדיפות שווה, יש לבחור בסדר אלפביתי (A עדיף על B).

הניחו שלא מתבצעות בדיקות למניעת מופעים כפולים של צמתים במסלול.

1. BFS
2. Iterative Deepening
3. Uniform Cost Search
4. Greedy Best First Search
5. A*

ב. האם הפונקציה היוריסטית הנתונה h קבילה (admissible)? הסבירו את תשובתכם.

ג. האם הפונקציה היוריסטית הנתונה h עקבית (consistent)? הסבירו את תשובתכם.

שאלה 3 (15 נק': 5 נק' לכל סעיף)

נתבונן בבעיה הבאה:

מעונינים להציב פרשים, במספר גדול ככל האפשר, על לוח משבצות מסדר $n\times n$ (לוח כמו לוח שח, אלא שהגודל של הלוח איננו דווקא 8×8), באופן שכל זוג פרשים על הלוח לא יאיימו זה על זה.

א. נסחו את הבעיה כבעיית חיפוש מקומי:
כיצד נראה מצב של הבעיה.
(זכרו שבחיפוש מקומי הניסוח הוא של מצב-שלם (complete-state formulation)).
מהו מודל המעברים?
מהי פונקצית המטרה?

ב. הוכיחו: לכל n טבעי, אפשר להציב לפחות $\left\lceil \frac{n^2}{2} \right\rceil$ פרשים על לוח מסדר $n\times n$.
($\lceil x \rceil$ הוא המספר השלם הקטן ביותר שאינו קטן מ-x. למשל, $\lceil 5/2 \rceil = 3$, $\lceil 7 \rceil = 7$, $\lceil -3/2 \rceil = -1$).

ג. תנו דוגמה למקסימום מקומי שאיננו מקסימום גלובלי בלוח מסדר 5×5.
(עליכם להראות מצב s שבו כל צעד (לפי מודל המעברים) מעביר למצב (שכן) שערך פונקצית המטרה שלו איננו גדול מערך פונקצית המטרה של s, וערך פונקצית המטרה של s איננו הגדול ביותר האפשרי).

שאלה 4 (8 נק')

נתונים 10 קלפים הממוספרים $1,\ldots,10$.
מצאו בעזרת אלגוריתם גנטי חלוקה שלהם לשתי תת-קבוצות של 5 קלפים כל אחת, כך ש:

- סכום הקלפים בקבוצה הראשונה יהיה קרוב ככל האפשר ל-36.
- מכפלת הקלפים בקבוצה השניה תהיה קרובה ככל האפשר ל-360.

שאלה 5 (18 נק': 12 נק' לסעיף א'; 6 נק' לסעיף ב')

א. נתון עץ משחק שבו יש שני בנים לכל צומת פנימי. מלבד השורש יש בעץ ארבע רמות נוספות (לעץ יש 16 עלים). ערכי העלים בעץ הם (לפי הסדר):

$$5, 7, 4, 8, 9, 10, 2, 4, 4, 11, 12, 9, 1, 8, 15, 18$$

1. ציירו את העץ, וקבעו את ערכי הצמתים הפנימיים לפי אלגוריתם מינימקס. סמנו את החלטת המינימקס בשורש העץ.
2. הפעילו גיזום אלפא-ביתא על העץ, כאשר סריקת העץ היא משמאל לימין. ציינו אלו ענפים ייגזמו.
3. הפעילו גיזום אלפא-ביתא על העץ, כאשר סריקת העץ היא מימין לשמאל. ציינו אלו ענפים ייגזמו.

ב. תנו דוגמה לעץ משחק בעל עומק 3 (בנוסף לשורש יש עוד שלוש רמות), שבו לכל צומת פנימי יש שני בנים (יש לעץ 8 עלים), ואין כל חיסכון אם משתמשים באלגוריתם אלפא-ביתא, בין אם מבצעים סריקה של העץ משמאל לימין, ובין אם מבצעים סריקה של העץ מימין לשמאל. (בכל מקרה מבקרים בכל הצמתים).
בחרו את הערכים של העלים מן המספרים הטבעיים שבין 1 ל-8, כך שלכל עלה יהיה ערך שונה.

מטלת מנחה (ממ"ן) 13 - להרצה

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: פרקים 1-5

מספר השאלות: 3

משקל המטלה: 4

סמסטר: 2019א

מועד אחרון להגשה: 24.11.2018

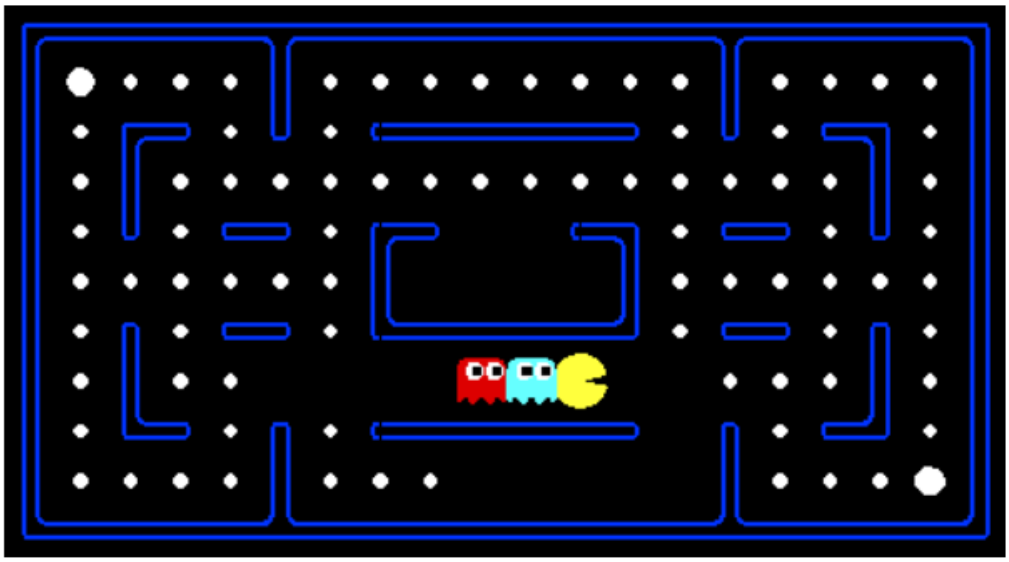
(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

Pac-Man מרובה סוכנים - Multi-Agent Pac-Man



Pacman, now with ghosts.

הקדמה

במטלה זו, עליכם לתכנן סוכנים עבור הגירסה הקלאסית של Pacman עם רוחות רפאים. לצורך כך תממשו את אלגוריתם מינימקס, אלפא-ביתא ותתנסו בתכנון פונקציית הערכה.
כקודמתה, גם מטלה זו פותחה באוניברסיטת ברקלי על-ידי John DeNero ו-Dan Klein.

הקוד עבור מטלה זו לא השתנה בהרבה ביחס לזה של המטלה הקודמת, אך רצוי מאד כי תתקינו אותו מחדש. בכל אופן, אתם יכולים להשתמש בקבצים search.py ו-searchAgents.py שלכם בכל דרך שתרצו.
את כל הקוד והקבצים הנלווים ניתן להוריד כקובץ zip.

הקוד עבור המטלה מכיל את הקבצים הבאים:

<u>קבצים שיש לקרוא</u>:

multiAgents.py	כל סוכני החיפוש (multi-agent) שלכם יימצאו שם.
pacman.py	הקובץ העיקרי שמריץ משחקי פקמן. הקובץ מתאר את הטיפוס GameState שתשתמשו בו הרבה במטלה זו
game.py	הלוגיקה שמאחורי אופן פעולת עולם הפקמן. קובץ זה מתאר מספר טיפוסי עזר כגון AgentState, Agent, Direction, Grid
util.py	מבני נתונים שימושיים למימוש אלגוריתמי חיפוש

<u>קבצים שאין צורך להתעמק בקוד שלהם</u>:

graphicsDisplay.py	גרפיקה עבור פקמן
graphicsUtils.py	תמיכה לגרפיקה של פקמן
textDisplay.py	גרפיקת אסקי לפקמן
ghostAgents.py	סוכנים לבקרה על רוחות רפאים
keyboardAgents.py	ממשקי מקלדת לבקרה על פקמן
layout.py	קוד לקריאת קבצי פריסה (layout) ואחסון תוכנם

<u>עליכם להגיש</u>
במהלך ביצוע המטלה יש למלא חלקים חסרים בקוד של multiAgents.py.
עליכם לשלוח למנחה קובץ זה (בלבד) וקובץ עם שמכם, ת.ז. והתיעוד.
הנכם מתבקשים לא לשנות שמות של פונקציות או מחלקות הנתונות בקוד.

Pac-Man מרובה סוכנים

תחילה הריצו משחק של Pac-Man הקלאסי:

```
python   pacman.py
```

כעת הריצו את ReflexAgent הנתון ב-multiAgents.py:

```
python  pacman.py  -p  ReflexAgent
```

שימו לב שהוא משחק די גרוע אפילו עבור פריסות (layouts) פשוטות:

```
python  pacman.py  -p  ReflexAgent  -l  testClassic
```

בדקו את הקוד (ב-multiAgents.py) וודאו שאתם מבינים את אופן פעולתו.

שאלה 1

שפרו את ReflexAgent ב-multiAgents.py כדי שישחק ברמה סבירה.

(בקוד הנתון עבור סוכן של תגובה-פשוטה (reflex agent) יש כמה דוגמאות שיכולות להיות לעזר של שיטות שמתשאלות את ה-GameState).

סוכן טוב של תגובה פשוטה צריך לקחת בחשבון הן מיקומים של אוכל והן מיקומים של רוחות רפאים. הסוכן שלכם אמור לאפס בקלות את פריסת ה-testClassic:

```
python  pacman.py  -p   ReflexAgent  -l   testClassic
```

נסו את הסוכן תגובה-פשוטה שלכם על פריסת ברירת המחדל mediumClassic- עם רוח רפאים אחת או שתיים (וכבו את האנימציה כדי להאיץ את הריצה):

```
python  pacman.py   --frameTime   0   -p   ReflexAgent   -k   1

python  pacman.py   --frameTime   0   -p   ReflexAgent   -k   2
```

קרוב לוודאי שהסוכן שלכם ימות לעתים קרובות כאשר יש שתי רוחות רפאים על לוח ברירת המחדל, אלא אם כן פונקציית ההערכה שלכם טובה למדי.

שימו לב:

- לא יתכן שיהיו לכם יותר רוחות רפאים מאשר מאפשרת ה-layout.
- פונקציית ההערכה שתכתבו מעריכה זוגות של מצב-פעולה; בהמשך המטלה תעריכו מצבים.

אופציות:

הרוחות שהן ברירת המחדל מתנהגות באופן אקראי. אתם יכולים בשביל הכיף לשחק עם רוחות קצת יותר חכמות בעזרת -g DirectionalGhost . אם האקראיות מונעת מכם להסיק האם הסוכן שלכם השתפר, אתם יכולים להשתמש ב -f כדי להריץ עם גרעין אקראי קבוע (אותן בחירות אקראיות בכל משחק). אתם יכולים לשחק משחקים מרובים ברצף עם -n. כבו את הגרפיקה עם q– כדי להריץ הרבה משחקים במהירות.

ניקוד מלא ינתן לסוכן שיכול לאפס במהירות את פריסת ה- openClassic עשר פעמים, מבלי למות יותר מפעמיים או לנוע הלוך ושוב באופן חוזר ונשנה בין שני מיקומים מבלי להתקדם.

```
python pacman.py -p ReflexAgent -l openClassic -n 10 –q
```

שאלה 2

כעת עליכם לכתוב קוד עבור סוכן חיפוש בתנאי יריבות במחלקה MinimaxAgent אשר ב-multiAgents.py. סוכן המינימקס שלכם אמור לעבוד לכל מספר של רוחות רפאים, לכן עליכם לכתוב אלגוריתם שיהיה קצת יותר כללי מזה שבספר הלימוד. ובמיוחד, לעץ המינימקס שלכם יהיו מספר רמות Min (אחת לכל רוח רפאים) עבור כל רמה של שחקן Max.
בנוסף, הקוד שתכתבו צריך לפתח את עץ המשחק לעומק כלשהו.
ערכי העלים בעץ המינימקס שלכם ייקבעו על ידי self.evaluationFunction הנתונה, כאשר ברירת המחדל שלה היא scoreEvaluationFunction. המחלקה MinimaxAgent יורשת את MultiAgentAgent , שנותנת גישה ל- self.depth ו-self.evaluationFunction . וודאו כי קוד המינימקס שלכם משתמש במשתנים אלה כשצריך, שכן משתנים אלה מקבלים ערך כתוצאה מהאפשרויות של שורת הפקודה (command line).

חשוב:

שלב אחד בחיפוש נחשב כמהלך אחד של Pac-Man וכל התגובות של רוחות הרפאים, לכן עומק 2 בחיפוש פירושו שני מהלכים של Pac-Man ושל כל רוח רפאים.

רמזים והערות

- פונקציית ההערכה עבור שאלה זו כתובה כבר (self.evaluationFunction). אינכם אמורים לשנות אותה, אך שימו לב לכך שכעת אנו מעריכים מצבים ולא פעולות, כפי שעשינו עבור הסוכן של תגובה פשוטה. סוכנים המסתכלים-קדימה מעריכים מצבים עתידיים בעוד שסוכנים של תגובה פשוטה מעריכים פעולות מהמצב הנוכחי.
- ערכי המינימקס עבור המצב ההתחלתי בסידור ה-minimaxClassic הם 9,8,7,-492 לעומקים 1,2,3,4 בהתאמה. שימו לב לכך שסוכן המינימקס שלכם ינצח לעתים קרובות (665/1000 משחקים) למרות שהחיזוי שלו הוא שמצבים בעומק 4 הם גרועים.

```
python pacman.py -p MinimaxAgent -l minimaxClassic -a depth=4
```

- כדי להגדיל את עומק החיפוש שתוכלו להשיג באמצעות הסוכן שלכם, הסירו את הפעולה Directions.STOP מהרשימה של Pac-Man של כל הפעולות האפשריות עבורו. עומק 2 יהיה מהיר יחסית אך עומק 3 או 4 יהיה איטי. בשאלה הבאה נטפל במהירות החיפוש.
- פקמן הוא תמיד סוכן 0, והסוכנים נעים בסדר עולה של האינדקסים שלהם.
- כל המצבים במינימקס צריכים להיות GameStates המועברים ל-getAction או נוצרים באמצעות GameState.generateSuccessor.
- עבור לוחות גדולים יותר כגון openClassic ו-mediumClassic (ברירת המחדל), תמצאו לנכון ש-Pac-Man עושה חיל ב"לא למות" אך גרוע ב"לנצח". לעתים קרובות הוא יסתובב סביב עצמו מבלי להתקדם. הוא אפילו עלול להסתובב ממש ליד נקודה מבלי לאכול אותה, משום שאינו יודע לאן ללכת לאחר שיאכל אותה.
- כאשר Pac-Man מאמין שמותו בלתי נמנע, הוא ינסה לסיים את המשחק מהר ככל האפשר, בגלל העונש הקבוע עבור הישארות בחיים. לעיתים זוהי הבחירה השגויה לטיפול ברוחות רפאים אקראיות אך סוכני מינימקס מניחים תמיד את הגרוע ביותר:

```
python pacman.py -p MinimaxAgent -l trappedClassic -a depth=3
```

וודאו כי אתם מבינים מדוע Pac-Man נחפז לרוח הרפאים הקרובה ביותר במקרה זה.

שאלה 3

צרו ב-AlphaBetaAgent סוכן חדש המשתמש בגיזום אלפא-ביתא כדי לסרוק את עץ המינימקס בצורה יעילה. שוב, האלגוריתם שלכם יהיה מעט יותר כלי מהפסאודו קוד שבספר הלימוד, כך שחלק מהאתגר הוא להרחיב את הרעיון שבגיזום אלפא-ביתא כך שיתאים לסוכני מינימום רבים.

עליכם לראות שיפור במהירות (למשל, אולי אלפא-ביתא בעומק 3 ירוץ באותו זמן של מינימקס בעומק 2). המצב האידיאלי הוא שעומק 3 ב-smallClassic ירוץ במספר שניות לכל מהלך או אף מהר יותר.

```
python pacman.py -p AlphaBetaAgent -a depth=3 -l smallClassic
```

ערכי המינימקס של AlphaBetaAgent צריכים להיות זהים לערכי המינימקס של MinimaxAgent, למרות שהפעולות שהוא בוחר יכולות להשתנות בגלל התנהלות שונה במקרים של החלטות שונות לשבירת שוויון. שוב, ערכי המינימקס של המצב ההתחלתי בסידור ה-minimaxClassic הם 9,8,7,-492 עבור העומקים 1,2,3,4 בהתאמה.

שאלה/ות נוספת/ות (והניקוד לכל השאלות במטלה) יתפרסמו באתר הקורס.

מטלת מנחה (ממ"ן) 14

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: פרקים 6-9

מספר השאלות: 5 **משקל המטלה:** 3 נקודות

סמסטר: 2019א **מועד אחרון להגשה:** 8.12.2018

(אב)

קיימות שתי חלופות להגשת מטלות:
- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

שאלה 1 (27 נק')

נתאר בעיה של פתרון תשבץ כבעית סיפוק אילוצים.

התשבץ מכיל שש מילים בנות שלוש אותיות כל אחת, שלוש מילים במאוזן (A1,A2,A3) ושלוש מילים במאונך (D1,D2,D3).

	D1 ↓	D2 ↓	D3 ↓
A1 →			
A2 →			
A3 →			

כל מילה צריכה להיבחר מתוך רשימת 40 המילים הבאות:

add, ado, age, ago, aid, ail, aim, air, and, any, ape, apt, arc, are, ark, arm, art, ash, ask,auk,

awe, awl, aye, bad, bag, ban, bat, bee, boa, ear, eel, eft, far, fat, fit, lee, oaf, rat, tar, tie

א. יש מספר אפשרויות לבחירת המשתנים לבעיה וביניהן:

i. ייצוג בו תחומי המשתנים מכילים מילים באנגלית.

ii. ייצוג בו תחומי המשתנים מכילים אותיות האלפבית.

לכל אחד משני הייצוגים הללו, תארו את קבוצת המשתנים, כתבו את מספר המשתנים שיידרשו לייצוג הבעיה שלעיל.

<u>התייחסו בסעיפים הבאים (ב-ה) לייצוג הראשון (i) מבין השניים שלעיל</u>:

ב. השתמשו כאמור בייצוג הראשון (i) ושרטטו את גרף האילוצים עבור בעיה זו.

ג. הפעילו אילוצים אונריים על גרף זה וכתבו (אם ניתן) לכל משתנה מהו התחום המצומצם שלו.

ד. הפעילו אלגוריתם עקביות קשת בגרף זה וכתבו (אם ניתן) לכל משתנה מהו התחום המצומצם שלו.

ה. הפעילו חיפוש backtracking , עם היוריסטיקות הרלבנטיות (MRV, יוריסטיקת הדרגה, LCV) כדי לפתור בעיה זו.

במקרה של שוויון בהערכת מספר מצבים, בחרו על-פי סדר אלפביתי וכתבו את הפתרון שהאלגוריתם מוצא לבעיה.

שאלה 2 (23 נק׳)

בחדר נמצאים שלושה אנשים A, B, C.

A נשוי ו- C לא נשוי.

A מסתכל על B ו- B מסתכל על C.

א. תארו את הבעיה בעזרת פסוקים בצורה נורמלית.

ב. הסיקו בעזרת רזולוציה שנמצא בחדר אדם נשוי שמביט על אדם לא-נשוי.

שאלה 3 (14 נק׳: 10 נק׳ לסעיף א׳; 4 נק׳ לסעיף ב׳**)**

נתונות הפסוקיות שלהלן:

א. השתמשו ביוריסטיקות pure-symbol ו-unit-clause (סעיף 7.6.1) כדי להראות כיצד ניתן לפשט פסוקיות אלה לפני שמתחילים בחיפוש הצבה המספקת את כל הפסוקיות. בכל שלב של תהליך הפישוט ציינו איזו יוריסטיקה מאפשרת אותו.

1. $P \vee Q \vee \neg R$
2. $\neg Q \vee R \vee \neg S$
3. $\neg S \vee \neg R \vee W$
4. $R \vee W \vee Y \vee Z$
5. S
6. $\neg W \vee \neg S$
7. $\neg Y \vee \neg Z$

ב. כמה מצבים (ממרחב החיפוש) ניתן לחסוך כאשר נעשה שימוש ביוריסטיקות הללו? הסבירו את תשובתכם.

שאלה 4 (6 נק': 1 נק' לכל סעיף)

להלן נתונים זוגות של ביטויים. עבור כל זוג בדקו האם ניתן לבצע האחדה בין שני הביטויים.
אם כן מצאו את המאחד הכללי ביותר (MGU) (אם הוא קיים כאמור); אחרת כתבו הסבר קצר.

א. *In (z, Office-of(z))* , *In (x,y)*

ב. *In (z, Office-of(z))* , *In (x,x)*

ג. *In (z, Office-of(w))* , *In (x,y)*

ד. *P(A,y,z)* , *P(x,B,B)*

ה. *P(A,y,z)* , *P(y,y,B)*

ו. *P(F(y,F(y,A)),A)* , *P(F(x,x),A)*

שאלה 5 (30 נק')

דן נרצח. גיא, דודי ועופר חשודים ברצח.
גיא אומר שהוא לא רצח. הוא אומר שדודי היה החבר של דן אבל עופר שנא את דן.
דודי אומר שהוא היה מחוץ לעיר ביום הרצח ובנוסף לכך הוא אפילו לא מכיר את האיש.
עופר טוען שהוא זכאי והוא ראה את גיא ואת דודי עם הנרצח ממש לפני הרצח.

בהנחה שכולם – פרט אולי לרוצח – דוברים אמת, השתמשו ברזולוציה כדי לגלות מי הרוצח.
באיזה סוג של יוריסטיקה השתמשתם בתהליך הרזולוציה?

הערה: באם הנכם סבורים כי נחוצים קשרים נוספים, ניתן להוסיפם ולהסביר מדוע הוספתם כל אחד מהם.

מטלת מנחה (ממ"ן) 15 - להרצה

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: פרקים 1-10

מספר השאלות: 14

משקל המטלה: 5

סמסטר: 2019א

מועד אחרון להגשה: 22.12.2018

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

Graphplan

הקדמה

במטלה זו, עליכם לממש חלקים מהאלגוריתם Graphplan ולתכנן יוריסטיקות מתוך גרף התכנון.

בחלק הראשון תשלימו את המימוש של Graphplan ותבדקו את האלגוריתמים על בעיות מהתחום של ה- "dock worker robot" (הוזכר במדריך הלמידה).

בחלק השני תשתמשו בגירסה מוחלשת של גרף התכנון כדי ליצור יוריסטיקות ל-A*.

בחלק האחרון, תיצרו קבצים לבעיית מגדלי האנוי.

הקוד עבור מטלה זו מורכב ממספר קבצי Python, חלקם יהיה עליכם לקרוא ולהבין כדי לבצע את המטלה ומהאחרים תוכלו להתעלם.

את כל הקוד והקבצים הנלווים ניתן להוריד כקובץ zip.

מטלה זו נלקחה מהקורס המקביל באוניברסיטה העברית.

הקבצים שיש לקרוא:

אלגוריתמי ה-graphPlan – מודול האחראי על יצירת Graphplan, הרחבתו במידת הצורך ויצירת תכנית plan.	graphPlan.py
ייצוג רמה אחת של הגרף (action layer & proposition layer)	planGraphLevel.py
ייצוג בעיית התכנון כבעיית חיפוש	planningProblem.py
יצירת קובץ הבעיה	hanoy.py

קבצים שאין צורך להתעמק בקוד שלהם:

action.py

proposition.py

actionLayer.py

propositionLayer.py

util.py

עליכם להגיש

יש למלא חלקים חסרים בקוד של graphPlan.py, planGraphLevel.py, planningProblem.py, ו-hanoy.py במהלך ביצוע המטלה.

בנוסף, עליכם ליצור שני קבצים dwr1.txt, dwr2.txt הדרושים לפתרון שאלה 9 וכן קובץ README.txt (כבמטלות הרצה הקודמות).

כל הקבצים שתגישו יהיו ארוזים בקובץ zip יחיד אותו עליכם לשלוח למנחה.

הנכם מתבקשים לא לשנות שמות של פונקציות או מחלקות הנתונות בקוד.

לפתרון המטלה קראו החל משאלה 1 (ולמעשה גם 5 שורות לפניה) ועד הסוף בקישור:

http://www.cs.huji.ac.il/~ai/graphplan/graphplan.html ופתרו את כל השאלות.

מטלת מנחה (ממ"ן) 16

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: פרק 10, פרקים 13-14

מספר השאלות: 4 **משקל המטלה:** 2 נקודות

סמסטר: 2019א **מועד אחרון להגשה:** 5.1.2019

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

שאלה 1 (25 נק')

פתרו את שאלה 10.3 שבספר הלימוד.

הערה: בסעיף השלישי (c) של השאלה יש לענות רק על החלק השני (האחרון) שלו, כלומר יש לענות בסעיף זה על השאלה:

נניח כי הקוף רוצה להטעות את המדענים, שיצאו להפסקת תה, ע"י כך שיתפוס את הבננות, אך ישאיר את הקופסה במקומה המקורי.

האם ניתן להגיע (לפתור) למטרה זו באמצעות מע' תכנון קלאסי?

שאלה 2 (25 נק')

פתרו את שאלה 10.4 שבספר הלימוד.

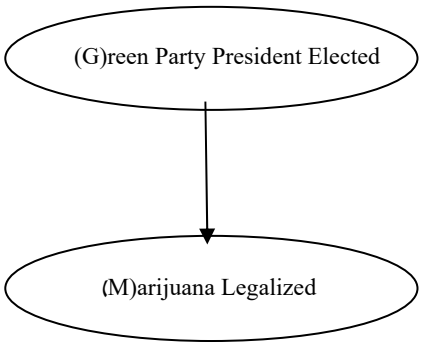
המשך המטלה בעמודים הבאים

שאלה 3 (35 נק')

כעת תקופת הבחירות העירוניות ויתכן שבעיר סדום יבחר מועמד מפלגת הירוקים. פלוני מאמין שיש סיכוי גבוה יותר שנשיא ממפלגת הירוקים יהפוך את המריחואנה לחוקית לעומת מועמדים ממפלגות אחרות, אבל גם כל נבחר אחר יכול להפוך את המריחואנה לחוקית.

נתאר את המצב בעזרת הרשת הביסיאנית שלהלן:

P(G)
0.1



G	P(M|G)
t	0.667
f	0.25

א. אמצעי התקשורת הודיעו שהמריחואנה עומדת להיות חוקית אך לא ידוע עדיין מי זכה בבחירות.

מהי ההסתברות P(G|M) לכך שנבחר נשיא ממפלגת הירוקים?

ב. אם יהיה לנו יותר מידע, נוכל לבצע היסקים טובים יותר. נרחיב את הרשת הביסיאנית כלהלן, על ידי הוספת 2 משתנים אקראיים:

B – האם התקציב מאוזן?

C – האם אחוז ההגעה למפגשי ההנחיה יגדל?

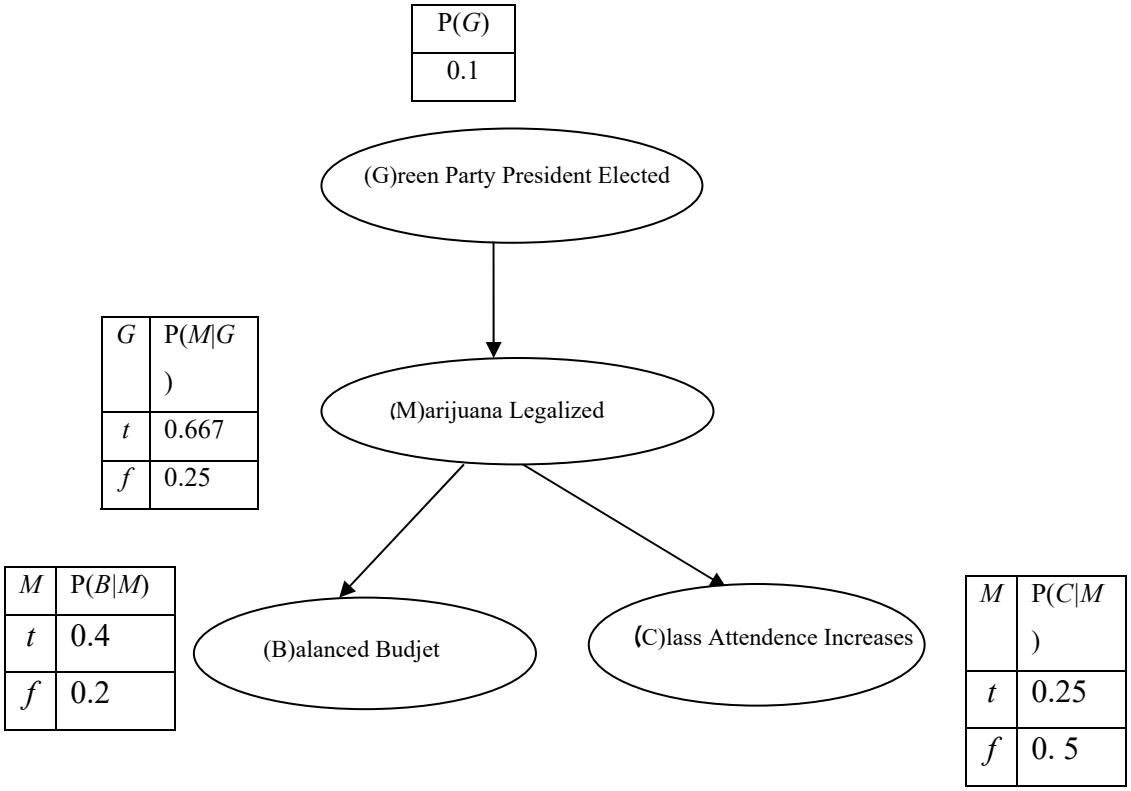
1. חשבו את ההסתברות המשותפת (joint distribution) של המשתנים G,M,B,C.

2. נניח כי אינכם יודעים עדיין את תוצאות הבחירות, אך הבחנתם כי אחוז ההגעה למפגשי ההנחייה קטן.
   מהי ההסתברות שמועמד מפלגת הירוקים נבחר?
   האם תשובתכם הגיונית? נמקו.

3. חשבו: P(B) ,P(C|B) ,P(B|M,G)

הערה: יש להימנע מחישובים מיותרים.

המשך השאלה בעמוד הבא



ג. נוסיף כעת לרשת צומת S המשקף את האפשרות שמחקר מדעי עשוי להשפיע על ההסתברות לקבלת החלטה שהמריחואנה תהיה חוקית.
הניחו כי המחקר אינו משפיע ישירות על B או על C.

1. שרטטו את הרשת החדשה המתקבלת.

2. איזה (אילו) CPT צריכים להשתנות בעקבות הוספת S?

3. בהתבסס על מבנה הרשת שהתקבלה בסעיף ג׳1 בלבד (ולא על ערכי ה-CPT), איזו(אילו) מהטענות הבאות נכונה, איזו(אילו) אינה נכונה ואיזו(אילו) מהטענות לא נובעת ממבנה הרשת? **נמקו.**

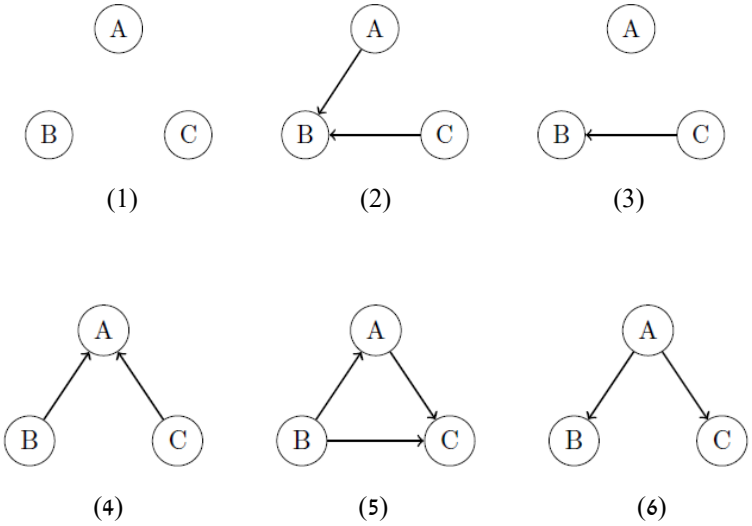
i. $P(B,C) = P(B) \cdot P(C)$
ii. $P(B) = P(B|G)$
iii. $P(G) = P(G|S)$
iv. $P(C|G,M) = P(C|G)$
v. $P(G|S,M)=P(G|S)$
vi. $P(G|S,B)=P(G|S)$
vii. $P(B|C,G)=P(B|C)$

שאלה 4 (15 נק')

נתונה טבלת ההסתברות המשותפת הבאה:

A	B	C	P(A,B,C)
0	0	0	0.15
0	0	1	0.1
0	1	0	0
0	1	1	0.25
1	0	0	0.15
1	0	1	0.1
1	1	0	0
1	1	1	0.25

אילו מהרשתות הבייסיאניות הבאות (אם בכלל) מייצגות את ההתפלגות שלעיל:



מטלת מנחה (ממ"ן) 17

הקורס: **20551 – מבוא לבינה מלאכותית**

חומר הלימוד למטלה: פרקים 18-17

מספר השאלות: 2 **משקל המטלה:** 2 נקודות

סמסטר: 2019א **מועד אחרון להגשה:** 23.1.2019

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

שאלה 1 (50 נק')

נתון MDP עם 6 מצבים: חדר רחצה, מטבח, חדר שינה, חדר אוכל, תחת מתקפה, מת

המייצגים תחום של עכבר רובוטי המחפש מזון בבית שבו יש ארבעה חדרים (חדר רחצה, מטבח, חדר שינה, חדר אוכל).

במצבים: חדר רחצה, מטבח, חדר שינה, חדר אוכל

יש 3 פעולות אפשריות: להישאר במקום (S), לנוע אופקית (H) או לנוע אנכית (V).

תוך כדי חיפוש מזון, העכבר יכול להיות מותקף על-ידי חתול רובוטי הנמצא אף הוא בבית, דבר הגורם לעכבר להיכנס למצב "תחת מתקפה" אשר בו תיתכן רק פעולה אחת: "למות".

במצב "מת" תיתכן רק הפעולה: "להישאר מת".

התגמולים (rewards) ומודל המעברים נתונים להלן:

	P(s' | s, a)					
s, a	חדר רחצה	מטבח	חדר שינה	חדר אוכל	תחת מתקפה	מת
חדר רחצה, H	0	0.6	0.4	0	0	0
חדר רחצה, V	0	0.4	0.6	0	0	0
חדר רחצה, S	0.75	0	0	0	0.25	0
מטבח, H	0.6	0	0	0.4	0	0
מטבח, V	0.4	0	0	0.6	0	0
מטבח, S	0	0.75	0	0	0.25	0
חדר שינה, H	0.4	0	0	0.6	0	0
חדר שינה, V	0.6	0	0	0.4	0	0
חדר שינה, S	0	0	0.75	0	0.25	0
חדר אוכל, H	0	0.6	0.4	0	0	0
חדר אוכל, V	0	0.4	0.6	0	0	0
חדר אוכל, S	0	0	0	0.75	0.25	0
תחת מתקפה, למות	0	0	0	0	0	1.0
מת, להישאר מת	0	0	0	0	0	1.0

s	R(s)
חדר רחצה	+4
מטבח	+10
חדר שינה	0
חדר אוכל	+2
תחת מתקפה	-50
מת	0

א. מהו המספר הכולל של ה-policies האפשריות?

ב. בצעו value iteration עבור בעיה זו.

הערך ההתחלתי של כל מצב הוא 0.

מכיוון שעכברים רובוטיים הם במידת מה קצרי רואי, השתמשו ב-$\gamma=0.5$.

כתבו את הערכים לכל המצבים, לאחר כל איטרציה.

ניתן להפסיק לאחר 6 איטרציות.

ג. מהי המדיניות האופטימלית בהינתן הערכים עבור המצבים, שקיבלתם בסעיף ב'?

שאלה 2 (50 נק')

נתונות דוגמאות האימון הבאות:

Class	F5	F4	F3	F2	F1	
p	false	false	false	true	true	דוגמא1
p	false	true	true	false	false	דוגמא2
p	false	true	false	false	true	דוגמא3
p	true	false	true	false	true	דוגמא4
n	false	false	false	true	false	דוגמא5
n	true	true	false	true	true	דוגמא6
n	true	true	true	true	false	דוגמא7

א. בנו עץ החלטה תוך שימוש בכל דוגמאות האימון.
פרטו את כל שלבי הבנייה.

ב. כיצד יסווג העץ שבניתם את הדוגמא הבאה:

Class	F5	F4	F3	F2	F1	
?	true	true	false	false	false	דוגמא8

מטלת מנחה (ממ"ן) 18 - להרצה

הקורס: 20551 – מבוא לבינה מלאכותית

חומר הלימוד למטלה: פרק 18 - למידה מדוגמאות

מספר השאלות: 1 **משקל המטלה:** 5 נקודות

סמסטר: 2019א **מועד אחרון להגשה:** 23.2.2019

(אב)

קיימות שתי חלופות להגשת מטלות:

- שליחת מטלות באמצעות מערכת המטלות המקוונת באתר הבית של הקורס
- שליחת מטלות באמצעות הדואר או הגשה ישירה למנחה במפגשי ההנחיה

הסבר מפורט ב"נוהל הגשת מטלות מנחה"

המטלה בעמודים הבאים

מימוש עצי החלטה

המטלה נלקחה מקורס מבוא לבינה מלאכותית CS 540.

In this problem you are to implement a program that builds a decision tree for categorical attributes and 2-class classification tasks. The programming part only requires building a tree from a training dataset and classifying instances of the test set with the learned decision tree.

You are required to implement four methods and one member for the class `DecisionTreeImpl`:

```
1. private DecTreeNode root;
2. DecisionTreeImpl ( DataSet train );
3. public String classify ( Instance instance );
4. public void rootInfoGain ( DataSet train );
5. public void printAccuracy (DataSet test)
```

`DecisionTreeImpl(DataSet train)` learns the decision tree from the training set, `train`.

`classify(Instance instance)` predicts the example, `instance`'s, label using the trained decision tree.

`rootInfoGain(DataSet train)` prints the information gain (one in each line) for all the attributes at the root based on the training set, `train`. The root of your tree should be stored in the member root that has been declared for you. The next sections describe other aspects in detail.

`printAccuracy(DataSet test)` prints the classification accuracy for the instances in the test set, `test`, using the learned decision tree.

Dataset

Our datasets come from a risk loan dataset, which is being used to predict the risk quality of a loan application. There are altogether 1000 examples (also called instances) and we chose 10 categorical attributes to use for this assignment. Each instance is classified as good (class G) or bad (class B), so this is a 2-class classification problem. You can assume other datasets used for testing will also be 2-class classification tasks.

The tables of attributes and their possible values are shown in the table below:

A1: Checking status	x(no checking) n(x<0, negative) b(0<=x<200, bad) g(200<=x, good)
A2: Saving status	n(no known savings) b(x<100) m(100<=x<500) g(500<=x<=1000) w(1000<=x)

A3: Credit history	a(all paid) c(critical/other existing credit) d(delayed previously) e(existing paid) n(no credits)
A4: Housing	r(rent) o(own) f(free)
A5: Job	h(high qualified/self-employed/management) s(skilled) n(unemployed) u(unskilled)
A6: Property magnitude	c(car) l(life insurance) r(real estate) n(no known property)
A7: Number of dependents	1, 2
A8: Number of existing credits	1, 2, 3, 4
A9: Own telephones or not	y(yes), n(no)
A10: Foreign workers or not	y(yes), n(no)

In each file, there will be a header that gives information about the dataset; an example header and the first example in the dataset is shown below. First, there will be several lines starting with // that provide some description and comments about the dataset. Next, the line starting with %% will list all the class labels. Each line starting with ## will give the name of one attribute and all its possible values. We have written the dataset loading part for you according to this header, so do NOT change it. Following the header are the examples in the dataset, one example per line. The first example is shown below and corresponds to the feature vector (A1=x, A2=n, A3=e, A4=r, A5=h, A6=l, A7=1, A8=1, A9=y, A10=y) and its class is G.

```
// Description of the data set
%%,G,B
##,A1,x,n,b,g
##,A2,n,b,m,g,w
##,A3,a,c,d,e,n
##,A4,r,o,f
##,A5,h,s,n,u
##,A6,c,l,r,n
##,A7,1,2
##,A8,1,2,3,4
##,A9,y,n
##,A10,y,n
x,n,e,r,h,l,1,1,y,y,G
…
```

Implementation Details

Predefined Data Types

We have defined four data types to assist your coding, called `Instance`, `DataSet`, `DecTreeNode` and `DecisionTreeImpl`. Their data members and methods are all commented, so it should not be hard to understand their meaning and usage.

Building the Tree

In the `DecisionTreeImpl(DataSet train)` method, you are required to build the decision tree using the training data. Refer to the pseudocode in Figure 18.5 on page 702 of the textbook to see what your code should do.
To finish this part, you may need to write a recursive function corresponding to the `DecisionTreeLearning` function in the textbook.

Classification

`public String classify(Instance instance)` takes an example (called an instance) as its input and computes the classification output (as a string) of the previously-built decision tree. You do not need to worry about printing. That part is already handled in the provided code.

Printing and Information Gain at the Root

The only printing you need to do is in the method

`public void rootInfoGain(DataSet train)` and

`public void printAccuracy(DataSet test).`

In `rootInfoGain`, for each attribute print the output one line at a time: first the name of the attribute and then the information gain achieved by selecting that attribute at the root. The output order of the attributes and associated information gain values must be the *same* as the order that the attributes appear in the training set's header. Print your results with 5 decimal places using `System.out.format("%.5f\n", arg)`

In `printAccuracy`, you should only print out the accuracy with 5 decimal places.

Testing

We will test your program using several training and testing datasets using the command line format:

```
java HW3 <modeFlag> <trainFile> <testFile>
```

where `trainFile` and `testFile` are the names of the training and testing datasets, respectively. `modeFlag` is an integer from 0 to 3, controlling what the program will output. Only `modeFlag` = {0, 1, 2, 3} are required to be implemented for this assignment.
The requirements for each value of `modeFlag`= {0, 1, 2, 3} are described as following:

0: Print the information gain for each attribute at the root node based on the training set

1: Create a decision tree from the training set and print the tree

2: Create a decision tree from the training set and print the classification for each example in the test set

3: Create a decision tree from the training set and print the accuracy of the classification for the test set

To facilitate debugging, we have provided three input files called `example1.txt`, `example2.txt and example3.txt`. It is highly recommended that you write for yourself a small application that produces random input files in the expected format. Actually make them not so random, so that you'll know what to expect when you build your decision trees.

So, here is an example command:

```
java HW3 0 train1.txt test1.txt
```

You are *NOT* responsible for any file input or console output other than public void `rootInfoGain (DataSet train)` and `printAccuracy(DataSet test)`. We have written the class `HW3` for you, which will load the data and pass it to the method you are implementing.

The format of `rootInfoGain (modeFlag == 0)` should look like

```
A1 0.11111
A2 0.11111
…
A10 0.11111
```

The format of `printAccuracy (modeFlag == 3)` should look like

```
0.12345
```

As part of our testing process, we will unzip the file you submit, call javac`*.java` to compile your code, and then call the main method `HW3` with parameters of our choosing.

אופן ההגשה:

עליכם להעלות למערכת המטלות קובץ zip המכיל שני קבצים:

את הקובץ DecisionTreeImpl.java (המכיל את כל הקוד שכתבתם) וכן את הקובץ README.txt (המכיל שם, ת.ז. ותיעוד).