מבוא למערכות תבוניות, חכמות וקוגניטיביות -

דו"ח פרויקט סיום

איילת טננבוים

הקדמה

בדו"ח זה אציג את הסוכן שבניתי על סמך החומר הנלמד בקורס ומקריאה של חומרים נוספים באינטרנט.

במצב האידיאלי, הסוכן צריך לדעת להתמודד באופן הטוב ביותר עם שני סוגי העולמות:

1. עולם דטרמיניסטי
2. עולם הסתברותי

כאשר בכל עולם ישנם מצבים שונים:

1. סביבה מוכרת + בעיה מוכרת
2. סביבה מוכרת + בעיה לא מוכרת
3. סביבה ובעיה שאינן מוכרות

(מוכרת = בוצע עליה תהליך למידה)

בהמשך הדו"ח אפרט על הנושאים הבאים:

- המצבים בהם בחרתי להתמקד והשיטה בה הסוכן נוקט בכל מצב.
- מימושים שצלחו וכאלו שלא.
- השוואה בין תוצאות הסוכן בשיטות שונות.
- מסקנות ושיפורים להמשך.

עולם דטרמיניסטי

בסימולטור של הקורס קיים planner שפועל על עולמות דטרמיניסטיים ללא אובייקטים חבויים. הוא מקבל דומיין ובעיה ומחזיר את רצף הפעולות שיש לבצע עד להשגת המטרה. תוצאות ה-planner טובות ולכן בחרתי להשתמש בו במצב זה.

<u>שלב הלמידה</u>

אם מתבצע שלב למידה, אז מכיוון שהסוכן לא זקוק לשלב זה כאשר הוא משתמש ב-planner, הוא בודק האם הדומיין הוא דטרמיניסטי וללא אובייקטים חבויים, ואם כן, הוא יוצא מהתוכנית עם הפקודה exit(128) המציינת שניתן לעבור ישירות לשלב הביצוע.

<u>שלב הביצוע</u>

בשלב זה שוב מתבצעת בדיקה אם הדומיין הוא דטרמיניסטי וללא אובייקטים חבויים, ואם כן, הסוכן מפעיל את ה-planner, מקבל ממנו את רצף הפעולות שיש לבצע, ומבצע אותן אחת אחרי השנייה עד להשגת המטרה.

בהמשך אציג שימוש נוסף ב-planner גם בעולמות שאינם דטרמיניסטיים.

עולם לא דטרמיניסטי

<u>שלב הלמידה</u>

שלב זה מבוסס לכאורה על אלגוריתם Dyna-Q המשלב Q-learning יחד עם planning במטרה לשפר את Q-learning, אך מכיוון שבפועל ברוב הניסויים האלגוריתם הגיע לתוצאות פחות טובות מאשר Q-learning רגיל, נוטרל החלק בו מתבצע planning.

אלגוריתם Dyna-Q

אלגוריתם Dyna-Q הוא אלגוריתם המשלב למידה ותכנון. הוא לומד ומשפר גם את ה-Q-table וגם את הסביבה ומשתמש במודל הסביבה כדי להאיץ את תהליך הלמידה.

למידה - הכוונה ל-Q-learning שמתבסס על חוויה אמיתית של הסוכן כדי לשפר את ה-policy.
תכנון - הכוונה לשימוש בחוויה מדומה שנוצרה על ידי המודל כדי לשפר את ה-policy, בלי לבצע פעולות בפועל.

פסאודו קוד של האלגוריתם עבור סביבה דטרמיניסטית:

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Loop forever:
- (a) $S \leftarrow$ current (nonterminal) state
- (b) $A \leftarrow \varepsilon$-greedy(S,Q)
- (c) Take action A; observe resultant reward, R, and state, S'
- (d) $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
- (e) $Model(S,A) \leftarrow R, S'$
- (f) Loop repeat n times:
  - $S \leftarrow$ random previously observed state
  - $A \leftarrow$ random action previously taken in S
  - $R, S' \leftarrow Model(S,A)$
  - $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

Q-Learning

Model Update

Planning Step

הסבר: בשלב האתחול, מאתחלים Q-table ומודל שהמפתח של כל אחד מהם הוא מצב ופעולה. לאחר מכן, מתבצעות איטרציות הלמידה, כאשר כל איטרציה מורכבת מ-6 שלבים:

ארבעת השלבים הראשונים זהים ל-Q-learning.

השלב החמישי הוא שלב של למידת הסביבה, כלומר, עדכון המודל. עבור המצב הנוכחי והפעולה שנבחרה, מעודכן המצב החדש וה-reward שהתקבלו.

השלב האחרון משתמש במודל הנלמד כדי לעדכן את ה-Q-table n פעמים בלי לבצע פעולות בפועל (n מוגדר מראש להיות מס' הפעמים שהשלב הזה יתבצע בכל איטרציה). בכל אחת מ-n הפעמים, נבחר באופן רנדומלי מצב מבין כל המצבים שהסוכן כבר ביקר בהם ונבחרת באופן רנדומלי פעולה שנלקחה בעבר עבור המצב הזה. על בסיס בחירות אלו יתקבלו מהמודל ה-reward והמצב החדש, וה-Q-table יעודכן בהתאם.

מכיוון שבחרתי להשתמש באלגוריתם עבור עולמות לא דטרמיניסטיים, בוצע בו השינוי הבא: כל זוג של מצב ופעולה במודל, שומר מעבר ל-reward ולמצב החדש גם את מספר הפעמים שהפעולה הובילה למצב החדש, כך שהמודל יכול גם ללמוד את ההסתברויות לכל מצב. כעת, בשביל לקבל את הערכים הדרושים לעדכון ה-Q-value, מחשבים את ה-reward הצפוי כסכום משוקלל של ה-reward-ים לפי ההסתברות של הפעולות, ובאופן דומה מחשבים גם את ערך ה-Q המקסימלי המתקבל מהמצב החדש.

בפועל, בניסויים שערכתי על בעיות שונות עם ערכי n שונים, התוצאות הטובות ביותר התקבלו לרוב עבור n=0, כלומר, עבור שימוש ב-Q-learning רגיל. לכן, השארתי את המימוש של Dyna-Q בקוד עם אפשרות להחליט האם להשתמש בו או לא, ובאופן דיפולטיבי – אין בו שימוש. לכן אמשיך לפרט על שלב הלמידה עבור Q-learning.

Q-learning

<u>פונקציית reward</u>

פונקציית ה-reward בנויה באופן הבא:

- צעד שהוביל למצב ללא מוצא, כלומר, מצב שבו המטרה עדיין לא הושגה אבל לא ניתן להתקדם הלאה – מקבל תגמול שלילי גדול.
- צעד שהוביל להשגת המטרה הסופית – מקבל תגמול חיובי גדול.
- צעד שהוביל להשגת תתי-מטרות – מקבל תגמול חיובי בהתאם למספר המטרות שהושגו ובהתאם למידת ההתקדמות לעבר המטרה הסופית. כלומר, ככל שמספר תתי המטרות שכבר הושגו גבוה יותר, אז כל תת מטרה נוספת נותנת תגמול גדול יותר.
- צעד שהוביל לאיבוד תתי מטרות – מקבל תגמול שלילי הגדול בערכו המוחלט ב-1 מהתגמול שהתקבל עבור השגת תתי המטרות האלו. הסיבה לכך היא מניעת מצב בו הסוכן יעדיף לאבד מטרות כדי להשיג אותן שוב מאשר להתקדם לעבר המטרה הסופית.
- צעד רגיל שלא שינה את כמות תתי המטרות שהושגו – מקבל תגמול שלילי קטן.

ניצול זמן הלמידה

לטובת ניצול זמן הלמידה במלואו, בכל פעם שהסוכן מגיע למטרה, הוא מתחיל סבב למידה חדש מהמצב ההתחלתי הקיים בקובץ הבעיה, כך שהוא לומד את הבעיה שוב ושוב כמה שניתן.

שלב הביצוע

בשלב זה יש לבצע הפרדה בין מצב בו התבצעה למידה למצב בו לא התבצעה למידה.

אם התבצעה למידה – הסוכן פועל על פי ה-policy שהוא למד בשלב הלמידה.

כדי למנוע מהסוכן להיתקע בלולאה אינסופית של מעבר בין שני מצבים שמובילים זה לזה, קיים מנגנון שמזהה לולאה אפשרית ובמצב של זיהוי כזה – בחירת הפעולה הבאה מתבצעת באופן רנדומלי מתוך כל הפעולות האפשריות, במקום על פי ה-policy.

אם לא התבצעה למידה, ישנם שני מקרים:

מקרה 1: לא קיימים בסביבה אובייקטים חבויים.

במקרה זה הסוכן עושה שימוש במחלקה Converter שתפקידה להמיר דומיין לא דטרמיניסטי לדומיין דטרמיניסטי כך שניתן יהיה להשתמש ב-planner לצורך הפתרון. ה-Converter מקבל את קובץ הדומיין, ויוצר קובץ דומיין חדש שבו הפעולות נוצרות באופן הבא:

- כל פעולה שאינה הסתברותית בדומיין המקורי, מועתקת כפי שהיא לדומיין החדש.
- עבור כל פעולה הסתברותית בדומיין המקורי, נבחר האפקט בעל ההסתברות הגבוהה ביותר ונוצרת פעולה לא הסתברותית בדומיין החדש עם אותו אפקט. אם יש מספר אפקטים בעלי הסתברות גבוהה ביותר – נוצרת פעולה עבור כל אחד מהם.

כאשר נוצר קובץ דומיין דטרמיניסטי, מופעל ה-planner. הוא מקבל את הדומיין החדש והבעיה המקורית, ומחזיר את רצף הפעולות שיש לבצע עד להשגת המטרה.

בכל צעד, הסוכן בודק מה הפעולה הדטרמיניסטית הבאה שעליו לבצע לפי ה-planner, ממיר אותה לפעולה המקורית כפי שמופיעה בדומיין המקורי, ומבצע אותה.

מכיוון שהפעולות מתבצעות בעולם שאינו דטרמיניסטי, לא מובטח שהפעולה אכן הובילה למצב אליו ה-planner תכנן להגיע. לכן, לאחר כל צעד, ה-planner מופעל מחדש, כאשר המצב ההתחלתי שלו הוא המצב החדש אליו הגיע הסוכן. הפעלה זו מתבצעת ע"י יצירת קובץ בעיה חדש, הזהה לקובץ הבעיה המקורי, מלבד המצב ההתחלתי שמוחלף למצב הנוכחי של הסוכן.

המקרה היחיד בו הסוכן לא מפעיל שוב את ה-planner לאחר ביצוע פעולה הוא כאשר הסוכן לא שינה את מצבו ביחס למצב הקודם. הסיבה היא שבכל הנראה ה-planner יחזיר את אותה תוכנית פעולה כמו הקודמת ולכן אין טעם להפעיל אותו שוב. במקרה זה הסוכן יבצע שוב את הפעולה האחרונה שביצע.

נשים לב שמכיוון שרק חלק מהאפקטים של הדומיין המקורי מופיעים בדומיין החדש, ייתכן שה-planner כלל לא יצליח למצוא תוכנית פעולה. לכן, ניסיתי להמיר את הדומיין המקורי לדומיין חדש המכיל פעולה חדשה עבור כל אפקט של פעולה מקורית, ללא תלות בהסתברות שלו.
התוצאות שהתקבלו בדרך זו היו פחות טובות באופן משמעותי, לכן החלטתי לפעול רק על פי הדרך

הראשונה (הפונקציה עבור הדרך השנייה נמצאת בהערה ב-Converter), ואם ה-planner לא מוצא תוכנית פעולה, אז הוא פועל כמו בסביבה בעלת אובייקטים חבויים שלא נלמדה.

להלן השוואה בין 3 שיטות שונות עבור הדומיין ההסתברותי maze_domain_multi_effect_food המקורי, עם הבעיה t_5_5_5_food-prob0 ללא אובייקטים חבויים (הפרדיקט (at food1 g4) מופיע מהתחלה):

	המרת דומיין לדטרמיניסטי	Q-learning	רנדום (כפי שמוסבר במקרה 2)
מס' פעולות ממוצע לריצה	21	24	101

*מספר הפעולות הממוצע חושב ב-10 ריצות שונות של שלב הביצוע.

השוואה זו היא דוגמא לכך שהשיטה של המרת הדומיין לדטרמיניסטי בהחלט אפקטיבית מאוד.

<u>מקרה 2</u>: קיימים בסביבה אובייקטים חבויים.

אסביר מה הסוכן מבצע בפועל ומה הייתה השאיפה שלא הספקתי לסיים לממש טרם ההגשה.

במקרה זה לסוכן אין תוכנית לפעול על פיה. הוא לא למד את הבעיה ולכן אין לו policy, וגם לא ניתן להפעיל את ה-planner מכיוון שיש בסביבה אובייקטים חבויים. לכן, בכל צעד, הסוכן בוחר פעולה באופן רנדומלי, מתוך הפעולות האפשריות שלא ביצע לאחרונה, אם קיימת כזו (בנוסף, אם הפעולה האחרונה לא הצליחה, הוא יכול לבחור בה שוב). אם לא, הוא בוחר באופן רנדומלי מתוך כל הפעולות האפשריות. המטרה היא מצד אחד לגרום לסוכן לנסות מגוון פעולות כדי שיבקר בהרבה מצבים בסביבה ושלא יתקע במקום, ומצד שני לא לכפות עליו יותר מדי בחירת פעולות חדשות כי ייתכן שהפעולות האחרונות כן עתידות לקדם אותו לעבר המטרה.

התכנון המקורי שלי עבור מקרה 2 שאני מאמינה שיכול להביא לשיפור משמעותי בביצועים, הוא להיעזר במודל הסביבה אם קיים. כלומר, במצב בו לא התבצעה למידה של הבעיה הספציפית, אבל כן התבצעה למידה של הסביבה, ניתן להיעזר במודל הסביבה (אותו אפשר ליצור באופן דומה לעדכון המודל באלגוריתם Dyna-Q שהצגתי) באחד משני האופנים הבאים:

1. בהמשך לשיטה של המרת עולם לא דטרמיניסטי לעולם דטרמיניסטי, ניתן לשמור עבור כל סביבה את כל הפרדיקטים החבויים שלה, וכאשר מתקבלת בעיה חדשה לביצוע בסביבה מוכרת, קובץ הבעיה יומר לקובץ בעיה המכיל את כל הפרדיקטים שקיימים בסביבה, ללא פרדיקטים חבויים, ואז הסוכן יוכל לפעול כמו במקרה הראשון שתואר, כאשר לא קיימים בסביבה אובייקטים חבויים.
2. שיטה אחרת היא בניית גרף המייצג את הסביבה.
אם הבעיה נלמדה אפילו פעם אחת בלבד, המצב ההתחלתי והמצב הסופי יופיעו בגרף, כך שניתן יהיה למצוא מסלול ביניהם ולפעול על פיו כמה שניתן.
אם הבעיה לא נלמדה כלל, ניתן יהיה לעשות סריקה של הגרף כך שהסוכן יבקר בכמה שיותר מצבים וכך בסבירות גבוהה השגת המטרה תהיה מהירה יותר מאשר בחירה רנדומלית של פעולות.

סיכום

בפרויקט זה מומש סוכן הפועל במגוון דומיינים ומבצע בהם משימות שונות. הדומיינים יכולים להיות דטרמיניסטיים או הסתברותיים, ובעלי אובייקטים חבויים או ללא. המשימות יכולות להיות בעלות מטרה אחת או יותר.

במצב בו הסוכן מבצע למידה של הבעיה טרם ביצועה, הוא היה צריך להשתמש באלגוריתם Dyna-Q הלומד את הסביבה תוך כדי למידה ושיפור של ה-policy ונעזר במודל זה כדי לייעל את למידת ה-policy. בפועל, האלגוריתם שהיה צריך לשפר את הביצועים, הביא לרוב לביצועים פחות טובים, ולכן הוחלט להשתמש ב-Q-learning הלומד את ה-policy בלבד. פונקציית התגמול של הלמידה מבוססת על השגת/איבוד מטרות, דבר שמשפר את הביצועים עבור בעיות עם יותר ממטרה אחת.

במצב בו הסוכן לא מבצע למידה של הבעיה טרם ביצועה: אם הבעיה דטרמיניסטית – הוא מקבל תוכנית פעולה מה-planner ופועל על פיה. אם הבעיה לא דטרמיניסטית – הוא מנסה להמיר אותה לבעיה דטרמיניסטית ולפתור אותה בעזרת ה-planner, ואם לא מצליח – מבצע פעולה רנדומלית תחת הגבלות להימנעות מחזרה על פעולות אחרונות.

ההתמקדות שלי הייתה בפתרון בעיות מוכרות עליהן בוצעה למידה מלאה, ובפתרון בעיות שאינן מוכרות כלל. לא הספקתי לסיים לממש טיפול בבעיות שרק הסביבה שלהן מוכרת, אך הצגתי את עיקרי הרעיונות שלי לביצוע.