מבוא לבינה מלאכותית - 67842 - פרויקט גמר

דוד רייטבלט קטיה צ'ירקו

23 במרץ 2016

תיאור המשחק

וויסט הוא משחק של 4 שחקנים שפותח באנגליה במאה ה־18, והתפתח ממשחק קלפים פרימיטיבי אחר, הנקרא Ruff and Honours. במאה האחרונה, המשחק וויסט איבד מהפופולריות שלו בגלל משחק הברידג', הדומה לו מאוד בחוקי המשחק, אך פשוט יותר להבנה ולמשחק על ידי אנשים. למרות זאת, באנגליה המשחק עדיין יחסית פופולארי ואף נערכים טורנירים בעלי מספר ימים הידועים בכינוי "Whist Drives". הודות לכך שמשחק הברידג' תפס את מקומו של וויסט, הרבה מהמחקרים בתחום הבינה המלאכותית מתמקדים במשחק הברידג' ואין הרבה מחקרים על הוויסט. עם זאת, וויסט הוא משחק מעניין בגלל חופש הפעולה הרחב שיש לשחקן בכל שלב במשחק, ובגלל העובדה שבתחילת כל משחק השחקן מציב לעצמו מטרה שבה הוא מחוייב לעמוד במדוייקת. לרובת שחקני ברידג' מאמינים כי משחקם מתוחכם יותר ומורכב יותר כך שדורש רמת תחכום גבוהה יותר מהשחקנים, אך למעשה המשחק ברידג' משאיר קצת מקום ליוזמה חופשית ולהברקות אישיות לעומת וויסט, כיוון שבמשחק זה ניתן בצורה חכמה לגרום לשאר השחקנים להחטיא את התחייבותם בעוד שבעצמך להרוויח מכך. ניטען כי משחק הוויסט מביא לידי ביטוי את האישיות והאופי של השחקן, כך שבנדם אימפולסיבי ישחק בצורה הניראת חסרת עמדה מסויימת אך למעשה בעלת תיכנון לטווח רחוק. חוקי המשחק וויסט הם כלהלן:

- המשחק מיועד לארבעה שחקנים
- כל חבילת הקלפים מחולקת שווה בשווה בין כלל השחקנים.
- המשחק אינו בעל ידיעה שלמה, כלומר כל שחקן לא יודע את הקלפים של השחקנים האחרים.
- בתחילת המשחק נערך שלב המכרז שבו כל שחקן מכריז על מספר השלבים שבהם ינצח - באופן שקול, מספר הלקיחות שהוא יצליח לקחת, ועל הסדרה השולטת שהוא רוצה.
- כל המשחק מתנהל ב־13 שלבים, כמספר הקלפים שיש לכל שחקן בתחילת המשחק.
- בכל שלב, השחקנים שמים כל אחד בתורו קלף אחד על השולחן, כאשר השחקן הראשון ששם קלף גם מחליט למעשה על הסדרה שבה מנוהל השלב הנוכחי. כעת, כל שחקן מחויב לשים קלף מאותה הסדרה, אם יש לו, ואחרת הוא יכול לשים כל קלף אחר, כאשר קלפים שאינם מהסדרה השולטת ואינם מהסדרה שבה מנוהל השלב אינם בעלי ערך בשלב זה.
- השחקן עם הקלף הגבוה ביותר מנצח בשלב הזה, כאשר קלף גבוה משמעותו ערך גבוה ביותר מבין כל הקלפים בעלי הסדרה שבה מנוהל השלב הנוכחי, אלה אם כן קיימים קלפים מהסדרה השולטת ואז הקף הגבוה ביותר הינו הקלף מהסדרה השולטת בעל ערך גבוה ביותר. ארבעת הקלפים שהשתתפו בשלב הנוכחי מוצעים מהמשחק ולא ישוחקו יותר במשחק זה. מצב זה נקרא "לקיחה", כיוון שלמעשה השחקן לוקח לעצמו את ארבעת הקלפים שיצאו מהמשחק (אך אינו מכניס אותם לחבילת הקלפים שבידו ואינו יכול לשחק איתם).
- בסוף המשחק, הניקוד של כל שחקן מחושב לפי כמות הלקיחות שלו ולפי מספר הלקיחות שעליו התחייב בתחילת המשחק, בשלב הבידינג. כאשר שחקן עמד ביעד שהציב לעצמו הוא מקבל ניקוד חיובי (תלוי במספר הלקיחות שלו), אחרת, אם השחקן עשה פחות לקיחות או אפילו יותר לקיחות, הוא מקבל ניקוד שלילי (אשר גם כן תלוי במספר הלקיחות שלו).

מהלך המשחק:

1. תחילת המשחק מחולקים כל הקלפים בין השחקנים, כאשר כל שחקן רואה רק את הקלפים שברשותו.

2. לאחר מכן מתחיל שלב המכרז: השחקן הראשון, הניקרא דילר, אומר בקול את הסדרה שלפי דעתו צריכה להיות הסדרה השולטת ומספר הלקיחות עליו הוא מתחייב. לאחר מכן, לפי סיבוב השעון כל שחקן אומר בתורו גם כן מהי לפי דעתו צריכה להיות הסדרה השולטת ומספר הלקיחות עליו הוא מתחייב. לבסוף השחקן שאמר את מספר הלקיחות הגדול ביותר מוכרז המנצח של המכרז והסדרה שלפי דעתו צריכה להיות הסדרה השולטת נהיית הסדרה השולטת. כעת, לפי בחירת הסדרה השולטתת כל שחקן מכריז על מספר הלקיחות עליו הוא מתחייבת ובכך נגמר שלב המכרז. אסור שסכום ההכרזות של כל השחקנים יהיה כמספר הקלפים שיש לכל שחקן, כיוון שבמקרה זה כל השחקנים יצליחו לצאת עם ניקוד חיובי ואז המשחק לא מעניין. אם כן, על השחקן האחרון שמכריז על מספר הלקיחות שהוא מתחייב קיים אילוץ נוסף. אם הסכום הזה קטן ממספר הקלפים שיש לכל שחקן המשחק נקרא 'under', ואחרת, אם הסכום גבוה ממספר הקלפים שיש לכל שחקן, המשחק נקרא 'over'.

3. השחקן שניצח במכרז מתחיל את שלב המשחק הראשון. הוא מניח קלף כלשהו לבחירתו מהקלפים שלו ולאחריו לפי הסדר כל שחקן מניח קלף חוקי. אם יש לשחקנים קלפים מאותה הסדרה כמו הקלף המתחיל, הם מחוייבים לשים את אחד מהקלפים האלה. אם אין להם קלפים כאלה, הם יכולים לשים כל קלף אחר, אך קלפים אלה לא יוכלו לנצח את השלב, אלה אם כן הקלף הינו מהסדרה השלטת, ואז הוא גם נחשב חזר יותר מכל קלף אחר שאינו מהסדרה השולטת. השחקן בעל הקלף הגבוה ביותר מנצח את השלב, והוא גם זה שמתחיל את השלב הבא.

4. לאחר מכן, כל שלב מנוהל בצורה דומה: השחקן הראשון בשלב שם קלף על השולחן ולאחריו כל שחקן מניח קלף חוקי שברשותו והשחקן בעל הקלף הגבוה ביותר מנצח את השלב.

5. בסוף המשחק מחושב הניקוד של כל שחקן בצורה הבאה: אם השחקן הכריז שיינצח ב־$n > 0$ שלבים, כלומר יעשה n לקיחות, והצליח בכך, הוא זוכה ב־$n^2 + 10$ נקודות. אחרת, אם השחקן עשה m לקיחות, כאשר $m \neq n$, אזי הוא מקבל $-10\,|n - m|$ נקודות. קיים מקרה נוסף מיוחד בו השחקן הכריז שיבצע $n = 0$ לקיחות, כלומר בכל מהלך המשחק אינו יקח אף לא יד אחת, אזי אם אכן השחקן הצליח במשימתו ולא עשה שום לקיחות והמשחק הוא $over$ הוא מקבל 50 נקודות, ואם המשחק הוא $under$ הוא מקבל 25. אם לעומת זאת השחקן כשל וכן לקח יד אחת לפחות אזי , הניקוד הוא $-50 + 10 \cdot (n - 1)$.

מבנה הפרוייקט

את הפרוייקט כתבנו בשפת פייתון. הפרוייקט מורכב מקבצים הדרושים להרצת המשחק עצמו ולמימוש הלוגיקה, וכמו כן מקובץ $Agents.py$ המכיל את צורות השחקנים השונות. למרות שלפי חוקי המשחק במשחק משתתפים 4 שחקנים כאשר כל אחד מתחיל עם 13 קלפים, הוספנו את האופציה לשנות את מספר השחקנים ומספר הקלפים ההתחלתי, על מנת שנוכל לבדוק את ביצועי השחקנים השונים ביעילות ובזמן סביר. גודל החפיסה מותאם למספר השחקנים ומספר הקלפים ביד ההתחלתית.

הפרוייקט בנוי מהקבצים הבאים:

1. $Whist.py$- המחלקה שמחזיקה את פונקציית ה - $main$ (הוראות להרצת המשחק ניתן למצוא בקובץ), וממשת את אובייקט משחק הוויסט, כולל פרמטרים של המשחק כמו מספר השחקנים השונה (ניתן להריץ את המשחק בעזרת 2 עד 4 שחקנים), מספר הקלפים שכל שחקן יתחיל איתם (ניתן להריץ את המשחק בעזרת 1 עד 13 קלפים לכל שחקן) ומספר המשחקים שרוצים להריץ ברצף (אין הגבלה על כמות המשחקים הרצופים שניתן להריץ). בנוסף קיימת פונקצית טסטים שניתן להריץ אותה ולראות בפעולה טסטים נבחרים המראים את ריצת המשחק עם מספר שחקנים שונה, מספר קלפים שונה, מספר משחקים שונה וכמובן מספר שחקנים מסוגים שונים המשחקים אחד מול השני. פונקציה זו מדפיסה לקובץ את התוצאות של הטסטים כך שניתן לראות את הניקדו של כל שחקן והזמנים שלקח להריץ את מספר המשחקים הרצוי.

2. $GameState.py$ - מחלקה המתארת מצב משחק כלשהו וכוללת את כל המידע הדרוש כדי להמשיך להריץ את המשחק מנקודה זו, כולל כל המידע שניתן להפיק ממהלך המשחק, כגון כמה לקיחות עשה כבר כל שחקן. מחלקה זו כוללת את כל המידע לגבי השחקנים, הסדרה השולטת, מהם הקלפים על השולחן, מיהו השחקן שתורו, מהם הקלפים שכבר שוחקו ומחוץ למשחק, סוג המשחק וכמה לקיחות עשה כבר כל שחקן, ובנוסף פונקציות שימושיות לניהול מידע זה.

3. $SingleGame.py$ - מחלקה זו מאפשרת להריץ משחק יחיד בעל מספר כלשהו של שלבים (לפי מספר הקלפים שיש לכל שחקן) וכוללת את כל הלוגיקה של המשחק וויסט. במחלקה זו ממומש חילוק הקלפים, המכרז ההתחלתי וכן הרצת כל השלבים לפי הסדר. מחלקה זו דואגת להפעיל את כל השחקנים לפי הסדר ולשמר את חוקיות המשחק, ולאפשר לראות על המסך את השתלשלות המשחק, בעזרת הדפסות אינפורמטיביות לגבי מה כל שחקן עשה או איזה קלף הוא בחר.

4. $Agents.py$ - המחלקה המרכזית שממשת את כל אלגוריתמי החיפוש, כלומר כל השחקנים השונים שיצרנו למשחק וויסט, ובעצם מהווה את לב הבינה המלאכותית שמשחקת במשחק וויסט. מופיע הסבר נרחב לגבי השחקנים השונים בהמשך.

5. $Card.py$ - מחלקה פשוטה ליצוג קלף במשחק, כוללת גם את הפונקצית ליצירת חבילת קלפים בגודל מתאים לכמות השחקנים ומספר הקלפים שיהיו לכל שחקן.

6. $Constants.py$ - מחלקה פשוטה לאיגוד כל הקבועים המשמשים אותנו בפרויקט.

7. $util.py$ - מחלקה פשוטה למימוש תור עדיפויות המשמשת להרצת אלגוריתם A^*.

השחקנים שמימשנו הם:

1. שחקן רנדומלי - תחילה, בשלב המכרז השחקן בוחר רנדומלי גם את הסדרה השולטת וגם מספר רנדומלי של לקיחות. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן שולף קלף רנדומלי מבין הקלפים החוקיים שהוא יכול לשים, ללא התחשבות האם כבר עבר את מספר הלקיחות שעליו הוא התחייב או לא.

2. שחקן רציונלי - השחקן פועל לפי אלגוריתם פרימיטיבי (לא AI) הגיוני התואם אסרטגיות משחק אנושיות. תחילה, בשלב המכרז השחקן בוחר את הסדרה השולטת כסדרה שממנה יש לו הכי הרבה קלפים, כאשר כל קלף נספר לפי ערכו, כך שקלף גבוה מסדרה מסוימת מוסיף יותר ערך לסדרה זו. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן מנסה לקחת כמה שיותר לקיחות, כל עוד הוא אינו עבר את מספר הלקיחות שהוא הכריז בשלב הבידינג, וכאשר הגיע כבר למכסת הלקיחות שלו הוא מנסה שלא לקחת שום קלף נוסף.

3. שחקן אנושי - נותן אפשרות לשחקן אנושי לשחק מול השחקנים השונים שמומשו, לעת עתה השחן האנושי יכול לצפות בקלפים של שאר השחקנים, מה שהופך את המשחק להרבה יותר פשוט עבור השחן האנושי.

4. שחקן $Max - Min$ - השחקן פועל לפי אלגוריתם בינה מלאכותית שממקסם את המינימום שהוא יכול להבטיח לעצמו. תחילה, בשלב המכרז השחקן פועל כפי שפועל השחקן הרציונלי. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן מנסה כל קלף חוקי שברשותו, פותח את כל עץ המשחק עם כל האפשרויות של הקלפים שיש לשחקנים האחרים ובוחר את הקלף שממקסם את מינימום הניקוד שהוא יכול להבטיח לעצמו לקבל בהנתן שיבחר לשחק בקלף זה. השחקן פועל מידיעה שלמה - הוא יודע בדיוק איזה קלפים יש לשאר השחקנים במשחק.

5. שחקן $Expecti - Max$ - השחקן פועל לפי אלגוריתם בינה מלאכותית שממקסם את התוחלת. תחילה, בשלב המכרז השחקן פועל כפי שפועל השחקן הרציונלי. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן מנסה כל קלף חוקי שברשותו, פותח את כל עץ המשחק עם כל האפשרויות של הקלפים שיש לשחקנים האחרים ובוחר את הקלף שממקסם את ממוצע הניקוד שהשחקן יכול לקבל בהנתן שיבחר לשחק בקלף זה. השחקן פועל מידיעה שלמה - הוא יודע בדיוק איזה קלפים יש לשאר השחקנים במשחק.

6. שחקן $Min - Max$ עם $alpha\text{-}beta\,pruning$ - שחקן שפועל לפי אלגוריתם בינה מלאכותית $alpha\text{-}beta\,pruning$ - תחילה, בשלב המכרז השחקן פועל כפי שפועל השחקן הרציונלי. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן מנסה כל קלף חוקי שברשותו, אך בשונה מהשחקנים לעיל, הוא אינו פותח את כל עץ המשחק עם כל האפשרויות של הקלפים שיש לשחקנים האחרים אלא רק ענפים שיובילו אותו לתוצאה לא פחות גרועה ממה שהוא כבר יכול להבטיח לעצמו. ברגע שהוא נתקל בתוצאה גרועה יותר מתוצאה כלשהי שהוא כבר הגיע אליה עם קלף אחר, הוא מפסיק לפתוח ענף זה בעץ. השחקן בוחר את הקלף שממקסם את ממוצע הניקוד שהוא יכול לקבל בהנתן שיבחר לשחק בקלף זה. השחקן פועל מידיעה שלמה - הוא יודע בדיוק איזה קלפים יש לשאר השחקנים במשחק.

7. שחקן $AStar$ - שחקן שפועל לפי אלגוריתם בינה מלאכותית A^* - תחילה, בשלב המכרז השחקן פועל כפי שפועל השחקן הרציונלי. לאחר מכן, בכל שלב שבו צריך

לשים קלף השחקן מנסה כל קלף חוקי שברשותו, אך בשונה מהשחקנים לעיל, הוא אינו פותח את כל עץ המשחק עם כל האפשרויות של הקלפים שיש לשחקנים האחרים אלא משתמש ביוריסטיקה על מנת לפתוח רק חלקים בעץ שיובילו לתוצאה הטובה ביותר. השחקן בוחר את הקלף שממקסם את ממוצע הניקוד שהוא יכול לקבל בהנתן שיבחר לשחק בקלף זה. השחקן פועל מידיעה שלמה - הוא יודע בדיוק איזה קלפים יש לשאר השחקנים במשחק.

8. שחקן מונטה קרלו - שחקן שפועל לפי אלגוריתם בינה מלאכותית $monte-carlo$ - תחילה, בשלב המכרז השחקן פועל כפי שפועל השחקן הרציונלי. לאחר מכן, בכל שלב שבו צריך לשים קלף השחקן פועל בודמה לשחקן $AStar$, אך בשונה ממנו, אינו פועל מתוך ידיעה שלמה. שחקן זה אינו משתמש בקלפים האמיתיים של השחקנים האחרים, אלה בעצמו מגריל לשאר השחקנים את הקלפים שלהם ומשתמש במידע זה כדי להריץ את החיפוש שהשחקן $AStar$ מבצע. למעשה שחקן זה מגריל לשאר השחקנים את הקלפים המון פעמים ומריץ חיפוש לאורך המון הגרלות שונות עבור קלפים לשאר השחקנים, ולבסוף בוחר את הקלף שמנצח בהכי הרבה משחקים אפשריים, בתקווה כי ההגרלות של הקלפים לשאר השחקנים ממדלים בצורה טובה את המציאות. שחקן זה כמובן אינו פועל מידיעה שלמה - הוא אינו יודע בדיוק איזה קלפים יש לשאר השחקנים במשחק, ולמעשה אפילו אינו נעזר במידע חלקי לגבי הקלפים שעשויים להיות לשאר השחקנים.

9. $alpha\text{-}beta$ בשלב המכרז - לכל השחקנים בעלי בינה מלאכותית, שחקנים מספר $4-8$, הוספנו אפשרות של מכרז בצורה חכמה, כלומר לבצע את המכרז הראשוני עבור הסדרה השולטת ומספר הלקיחות האופטימלי בעזרת אלגוריתם חיפוש שמבוצע לפי אסטרטגיית $min-max$ ומשתמש ב - $alpha\text{-}beta\,pruning$ בשביל היעילות. עד כה בשלב המכרז, לכל השחקנים, ספרנו את מספר הקלפים מכל סדרה ולא התחשבנו באלגוריתמים של בינה מלאכותית כדי לבצע בצורה חכמה את הבחירה של הסדרה ושל מספר הלקיחות.

אסטרטגיות המשחק

השחקן הרציונאלי:

השחקן הרציונלי בעצם ממדל מבחינתנו את השחקן האנושי החמדן אשר נוקט באסטרטגיה הבאה: בכל סבב במשחק, כשמגיע תורו, אם עדיין אינו הגיע ליעדו הוא בוחר את הקלף הגבוה ביותר מרשימת הקלפים החוקיים שברשותו ומשחק אתו, אם לחלופין הוא כבר לקח את מספר הלקיחות שבתחילה הוא התחייב אליו הוא בוחר את את הקלף הנמוך ביותר מרשימת הקלפים החוקיים שברשותו ומשחק אתו. מהלך פעולה זה גורם לכך שהוא מנסה בהתחלה לקחת כמה שיותר קלפים עד אשר הגיע ליעדו ולאחר מכן הוא מנסה להפסיד כמה שיותר לקיחות כדי לא לעבור את יעדו. נבחין, כי בניגוד למרבית השחקנים החכמים האחרים שממישנו, השחקן הרציונלי לא יודע את הקלפים של יריביו. ניתן לראות שיפור ניכר בביצועים של השחקן הרציונלי על פני הביצועים של השחקן הרנדומלי - לדוגמא, תוצאות של 10000 משחקים עם 2 שחקנים רנדומליים ושני שחקנים רציונליים:

$$defaultAgent1\,new\,score\,is:-205126$$

$$defaultAgent2\,new\,score\,is:-206853$$

$$RationalAgent3\ new\ score\ is: -116089$$

$$RationalAgent4\ new\ score\ is: -112164$$

וכפי שניתן לראות מהתוצאות, הביצועים של השחקן הרציוני כמעט פי 2 טובים יותר מהביצועים של השחקן הרנדומי.

מינימקס, אקספקטי־מקס ואלפא־ביתא:

שחקן המינימקס נוקט באסטרטגיה הבאה: בכל סבב במשחק, כשמגיע תורו, הוא מקבל את רשימת הקלפים החוקיים שברשותו. עבור כל קלף, הוא מדמה את כל המשחקים האפשריים שיכולים להיות אם ישחק בקלף זה, מחשב את התוצאה המינימלית שהוא יכול להגיע אליה אם ישחק בכל קלף, ובוחר את הקלף אם תוצאה מינימלית מקסימלית.

אלפא־ביתא:

מכיוון שהשחקן פותח עץ שלם של כל המשחקים האפשריים, החישוב של הקלף האופטימאלי איטי מאוד, ולוקח כ־20 דקות בממוצע למשחק עם 4 שחקנים ו־5 קלפים. לשם שיפור זמן הריצה הוספנו מימוש של אלגוריתם ה־ $alpha-beta\,pruning$: האלגוריתם פותח במלואו את עץ המשחק עבור הקלף הראשון, ושומר את התוצאה המינימלית האפשרית. אם בהמשך כשהוא פותח את עצי המשחק עבור קלפים אחרים, הוא רואה תוצאה אפשרית קטנה יותר מהמינימום שקיבל עד כה, הוא לא ממשיך לפתוח את עץ המשחק עבור אותו הקלף. אם הוא רואה קלף שהתוצאה האפשרית המינימלית עבורו יותר גדולה מהמינימום שהוא שומר, הוא מחליף את המינימום.

אלגוריתם ה־ $alpha-beta\,pruning$ משיג שיפור משמעותי בזמן הריצה שלו לעומת אלגוריתם המינימקס הנאיבי, וזמן הריצה שלו בממוצע משחק עם 4 שחקנים ו־5 קלפים ירד מכ־15 דקות לדקה אחת בלבד. כמובן שבמקרה הגרוע האלגוריתם מתנהג כמו אלגוריתם המינימקס הרגיל.

שחקן המינימקס לא משיג תוצאות טובות יותר, ואף נופל מהתוצאות של השחקן הרציונלי. תוצאות של משחק באורך 1000 של שחקן רציונלי מול שחקן המינימקס עם 5 קלפים:

$$RationalAgent1\ new\ score\ is: -1151$$

$$ExpectiMaxAgent2\ new\ score\ is: -1519$$

אקספקטי־מקס:

למראית עין, אסטרטגיית המינימקס לא מתאימה בצורה הטובה ביותר למשחק הוויסט־השחקן רודף אחר ההפסד המינימלי במקום לרדוף אחר הרווח, וכך יכול להיות שלעולם לא ינצח אלא רק ימזער הפסדים, ועדיין עלול להיות השחקן עם מספר הנקודות המינימלי. ע"מ לבדוק השערה זו, בנינו את שחקן ה־ $expecti-max$ במקום למקסם את הרווח המינימלי שלו מכל קלף אפשרי, השחקן ממקסם את הרווח הממוצע הצפוי. כלומר, בחישוב הקלף האופטימלי, השחקן מחשב את ממוצע הנקודות שיקבל בין כל המשחקים האפשריים שיכולים

להתפתח כתוצאה מבחירת קלף זה, וממקסם ערך זה. עם זאת, לא נראה שיפור בתוצאות עבור שחקן ה־ expecti-max: התוצאות של משחק באורך 100 של שחקן מינימקס מול שחקן ה־expecti-max, עם 5 קלפים לכל שחקן:

$$MinMaxAgent1\, new\, score\, is: -2712$$

$$ExpectiMaxAgent2\, new\, score\, is: -2719$$

א־סטאר:

מימוש שחקן ה־ א־סטאר מהווה גם שינוי באסטרטגיה וגם שיפור ניכר בביצועי זמן הריצה לעומת השחקנים הקודמים. כעת, במקום למקסם את התוצאה המינימלית שהשחקן יכול לקבל מקלף מסויים, או את ממוצע כלל התוצאות, השחקן שואף למקסם את התוצאה המקסימלית, כלומר השחקן שואף לניצחון במקום למזעור ההפסד. האלגוריתם ממומש בעזרת תור קדימויות ויורסטיקה שנרחיב עליה בהמשך.

זמן הריצה:

הושג שיפור ניכר מאוד בזמן ריצה ־ במשחק של 4 שחקנים עם 13 קלפים, משימה שהשחקנים הקודמים לא יכלו להתמודד עמה בזמן סביר, ממוצע הזמן למשחק עם שחקן א־סטאר אחד הוא כדקה, וכמובן שהזמן יורד אקספוננציאלית ככל שמורידים את מספר הקלפים. עם זאת, עדיין זמן הריצה במקרה הגרוע הוא כזמן הריצה של המינימקס הנאיבי. ע"מ להתמודד עם בעיה זו, מימשנו חיתוך של ה־ $fringe$ של תור הקדימויות ־ כאשר מספר התאים בתור הקדימויות עובר מספר מסויים (30000) מורידים את התאים הגרועים ביותר מהתור כך שגודלו לא יעלה על 30000. לא התאפשר לנו לבדוק כיצד זה משפיע על הביצועים, מכיוון שבלי שאנחנו מבצעים את החיתוך של ה־ $fringe$, כל התוכנית עלולה לרוץ יותר מדי זמן. כמו כן, גודל התור מגיע למספר זה לעיתים רחוקות מאוד, וזיופים באחוזים קטנים משפיעים מעט על התוצאה הסופית.

היוריסטיקה:

במימוש ה־ א־סטאר אנחנו משתמשים באותה היוריסטיקה שאנחנו משתמשים בה בשלב המכרז ־ השחקן מחשב כמה קלפים סביר שהוא ייקח במהלך המשחק. השחקן זוכר כמה קלפים הוא כבר לקח במהלך המשחק (נסמן מספר זה ב־ t) ומה מספר הסבבים שהכריז שינצח בהם בשלב המכרז (b). המטרה של השחקן, אם כן, לקחת $b-t$ קלפים אם $b>t$, ולא לקחת קלפים כלל אחרת. השחקן מחשב מה מספר הסבבים שינצח בהם במהלך יתר המשחק בצורה הבאה:

- הוא מחשב כמה קלפים, מתוך קלפים מהסדרה השלטת שיש לו, נכנסים לחלק ה־ $5/13$ מתוך הקלפים הגבוהים מהסדרה השלטת סה"כ. כלומר, אם יש לשחקן 13 קלפים בתחילת המשחק, הוא סופר כמה קלפים מתוך הקלפים שנשארו לו הם מתוך קבוצת 5 הקלפים הגבוהים ביותר מהסדרה השלטת: הקבוצה {אס, מלך, מלכה, נסיך, 10}. יש צורך לחשב את החלק היחסי כי מספר הקלפים שבחפיסה ההתחלתית יכול להשתנות בהתאם למספר השחקנים ולמספר הקלפים ההתחלתי שיש להם ביד.

- הוא מחשב כמה קלפים מתוך הקלפים הנותרים נכנסים לחלק ה־ $3/13$ מתוך הקלפים הגבוהים ביותר האפשריים. כלומר, אם יש לשחקן 13 קלפים בתחילת המשחק, הוא סופר כמה קלפים מתוך הקלפים שנשארו לו הם מתוך קבוצת 3 הקלפים הגבוהים ביותר: הקבוצה {אס, מלך, מלכה}.
- הוא מחבר שני מספרים אלה, ומקבל את מספר הסבבים שצפוי שינצח בהם מעתה ועד סוף המשחק. נסמן מספר זה ב־ e.

כעת, הוא מחשב את הניקוד שלו כאילו סיים את המשחק עם $t+e$ נצחונות, כאשר הכריז בהתחלה שיינצח ב־ b סבבים. לפי זה מחושב הניקוד הצפוי של המשחק.

ביצועים:

השחקן הנ"ל אכן השיג שיפור גם על פני שחקן המינימקס וגם על פני השחקן הרציונלי: תוצאות משחק באורך 1000 של שחקן ה־ א־סטאר מול שחקן המינימקס עם 5 קלפים לכל שחקן התחילת המשחק:

$$AStarAgent1\,new\,score\,is:-14513$$

$$Alpha-BetaAgent2\,new\,score\,is:-17829$$

תוצאות משחק באורך 1000 של שחקן ה־ א־סטאר מול השחקן הרציונלי עם 10 קלפים לכל שחקן התחילת המשחק:

$$AStarAgent1\,new\,score\,is:-17807$$

$$RationalAgent2\,new\,score\,is:-20704$$

מונטה־קרלו:

על מנת לבטל את ההנחה הקודמת שלנו שכל שחקן יודע את הקלפים של השחקנים האחרים, הרחבנו את שחקן ה־ א־סטאר עם שיטת המונטה קרלו: בכל פעם שהשחקן בוחר קלף, הוא מגריל משחקים עתידיים אפשריים שונים ע"י כך שמגריל את הקלפים שיש ליריביו. הוא מריץ סימולציה של המשחק על כל אחת מהאפשרויות האלה, ובוחר את הקלף שמנצח במספר הגבוה ביותר של המשחקים. מספר האפשרויות השונות שהוא מגריל פרופורציונלי למספר הקלפים החוקיים שיש ברשות השחקן. כמובן שאסטרטגיה זו פוגעת בזמן ריצה־ שעת משחק עם 4 שחקנים, 3 רציונליים ואחד מונטה־ קרלו, עם 10 קלפים בלבד לוקח כ־5 דקות. עם זאת, הביצועים של השחקן לא נפגעים. להלן תוצאות של משחק באורך 1000 עם 7 קלפים, של שחקן ה־ א־סטאר מול שחקן המונטה קרלו (במשחק זה הגדלנו את גודל החפיסה, כך שהקלפים שהיו לשחקן המונטה קרלו לא קבעו באופן חד משמעי את הקלפים של השחקן השני).

$$AStarAgent1\,new\,score\,is:-14905$$

$$Monte-CarloAgent2\,new\,score\,is:-14891$$

הוספת אלפא־ביתא לשלב המכרז:

בשלב המכרז, עד כה, כל השחקנים פעלו בצורה ברורה ללא חיפוש לעומק, ולמעשה נקטו בגישה חמדנית בלי ראייה רחבה על המשחק. כעת, הוספנו לכל שחקן בעל בינה מלאכותית גם אפשרות לשחק בעזרת חיפוש לעומק בשלב המכרז. כיוון ששלב המכרז נעשה פעם אחת במשחק, ומכך שהאלגוריתם A^* סיפק תוצאות לא טובות בשלב זה, פנינו לממש את שלב זה בעזרת $alpha\text{-}beta\,pruning$. לכל שחקן הוספנו אפשרות לחפש לעומק את המספר המקסימלי של הקלפים שהוא רוצה להתחייב עליו בעזרת האלגוריתם אלפה־בטה פרונינג, וראינו כי הוא אינו מביא לתוצאות טובות יותר מהאלגוריתם הנאיבי אך גם כן לא לתוצאות גרועות ביחס לאלגוריתם A^*. לאחר צפיה בתוצאות משחקים רבים אנו נוטים לחשוב כי לא מתקבלות תוצאות טובות יותר מכיוון שלמעשה כמעט בכל תת עץ משחק קיימות אפשרויות גרועות וטובות, וניתן להניח אפילו כי התוצאות מפוזרות בצורה אחידה בכל תתי העצים של המשחק. גם כאשר ניסינו לשלוט על עומק החיפוש קיבלנו כי זמן החיפוש נשאר גדול וגם התוצאה הסופית לרוב נמוכה מאוד, מה שגורם לכך שהשחקן למעשה לא מנצל את מלוא פוטנציאל הקלפים שיש לו ביד.

סיכום וכיווני המשך

ראינו כי המשחק עצמו בצורתו הכללית ביותר קשה לחישוב, ואלגוריתמים פשוטים לחיפוש בעץ המשחק לא מצליחים להתמודד עם כמות האפשרויות השונות במשחק עם כמות רבה של שחקנים או עם משחק בעל כמות גדולה של קלפים ביד של כל שחקן. לאחר כמה ניסיונות כושלים בעזרת פונקציות היוריסטיות שאינן מתאימות הצלחנו להפיק פונקציה טובה, בעלת זמן חישוב סביר המביאה לתוצאות טובות ביחס לשחקנים אחרים.

ראינו כי בשלב המכרז, חיפוש בעזרת $alpha\text{-}beta\,pruning$, אינו מביא לתוצאות טובות יותר מהאלגוריתם הנאיבי, מכמה סיבות. ניתן להבין מכך כי למעשה דרוש חישוב כלשהו השונה מחיפוש מעמיק לרוחב כל האפשרויות, המתחשב רק בקלפים שיש לנו ביד ויצליח לסווג קבוצות של קלפים למספר הלקיחות שהן מאפשרות, או אפילו מבטיחות, לעשות. ניתן להרחיב בכיוון הזה את הפרויקט בעזרת אלגוריתם למידה שיצליח לחזות בצורה טובה כמה שווה היד הנוכחית של השחקן. אפשר לנסות להריץ את המשחק המון פעמים בעזרת שחקן המנסה לקחת כמה שיותר קלפים מול שחקנים אינטיליגנטיים כלשהם ולראות האם ידיים מסויימות אכן מבטיחות מספר לקיחות כלשהו, או שאולי מספר הלקיחות תלוי בצורה רבה בשאר השחקנים ובצורת המשחק שלהם. למעשה, לבסוף לא הצלחנו להתקדם כלל בכיוון של שחקן אינטיליגנטי המצליח במכרז יותר טוב מהשחקן הרציונלי (אשר כפי שכבר טענו, ממדל טוב התנהגות אנושית במשחק זה).

בשלב המשחק עצמו, כאשר על השחקנים לבחור קלפים מסויימים קיבלנו תוצאות שונות, וחלקן לא לחלוטין גרועות. התחלנו מכך שהרצנו חיפוש מעמיק על עץ המשחק לקבלת הפתרון הטוב ביותר שניתן להבטיח. כפי שחשדנו בהתחלה, חיפוש כזה אינו יעיל ולא מאפשר להגיע לשום תוצאות עם יותר משני שחקנים ועם מספר לא טריויאלי של קלפים. לאחר מכן ניסינו לשפר את החיפוש בעזרת האלגוריתם $alpha\text{-}beta\,pruning$ אשר אכן שיפר את זמן החישוב וגרם לכך שאכן ניתן להריץ את המשחק עם ארבעה שחקנים, אך עדיין מספר הקלפים נשאר מצומצם, ובכך בעצם פגע ברעיון עצמו של החיפוש, כיוון שעבור מספר קטן של קלפים, התוצאות נקבעות בצורה חזקה הרבה יותר על ידי החלוקה הראשונית של הקלפים. אם כן, ניסינו להגביל את עומק החיפוש עצמו על עץ המשחק, אך כיוון זה התברר כפוגע בחיפוש מהכיוון השני, כיוון שכעט האלגוריתם עוצר בזמן סביר אך מחזיר פתרון רחוק מאופטימלי ולכן המשחק עצמו לא ניראה חכם במיוחד.

כאשר מימשנו את האלגוריתם A^*, בעזרת פונקציה יוריסטית כלשהי, קיבלנו תוצאות

הרבה יותר טובות, מבחינת זמן ריצה, וגם ניכר שיפור כלשהו בהתמודדות מול השחקנים הרציונליים. ניסינו להבין כיצד ניתן לשפר את הפונקציה היוריסטית ולבסוף החלטנו כי פונקציה המשערת כמה קלפים עוד סביר שניקח ממדלת בצורה די טובה את אשר אנו מחפשים. ניסינו למצוא פונקציה היוריסטית כזאת אשר תצליח בצורה לא רעה לקבוע מתי ניתן להתעלם מתת עץ כלשהו או מתי תת עץ מסוים אכן יכול להניב תוצאות טובות. כאשר נעזרנו באלגוריתם זה קיבלנו שחקן AI המאפשר לנו להריץ המון משחקים בזמן סביר, ומאפשר לנו לנתח יותר לעומק את טיב השחקן. כיוון ששאר השחקנים רצים בצורה איטית להחריד על המון קלפים, ניתן היה להשוות את השחקן A^* רק כאשר מראש התחלנו את המשחק עם מספר מועט של קלפים, אך כפי שנטען לעיל, מצב זה משבש את יכולת השחקנים להבין בצורה טובה את המשחק ואת המהלכים שעליהם לבצע ותלוי מאוד בחלוקה הראשונית של הקלפים.

לאחר שקיבלנו שחקן בעל יכולת מהירה לנתח את המשחק ולשחק בפרק זמן סביר, החלטנו לנסות למדל את המשחק בצורה טובה יותר בעזרת אלגוריתם מונטה קרלו, המאפשר, בתקווה, לשחקן להמשיך ולשחק בצורה טובה, בעל זמן תגובה מהיר אך גם ללא אפשרות להציץ בידיים של שאר השחקנים. כלומר, כעת , התקרבנו למשחק המקורי ומנענו מהשחקנים שלנו לראות את הקלפים של שאר השחקנים. להבדיל מהמצב שהשחקן שלנו רואה את הקלפים של שאר השחקנים ולפי זה מחליט מה עליו לעשות, מימשנו אלגוריתם המגריל לשאר השחקנים קלפים בצורה שרירותית, אך חוקית, ומריץ את המשחק פעמים רבות עם הגרלות קלפים שונות. כך עבור כל הגרלת קלפים מתקבל קלף אופטימלי כלשהו ועל ידי התבוננות על תוחלת הניצחונות עבור כל קלף ניתן לבחור קלף אשר ימקסם את המקרה הממוצע. לאחר השוואת שחקן A^* רגיל, בעל ידיעה שלמה על שאר הקלפים של השחקנים האחרים, ושחקן A^* הנעזר במונטה קרלו כדי למדל את השחקנים האחרים ראינו כי למעשה לא נפגעה היכולת של השחקן לנצח בצורה דומה, בתנאי שהוא מגריל מספיק פעמים קלפים שונים לשחקנים. עם זאת, זמן הריצה של השחקן נפגע משמעותית.

מבחינה לעומק של אלגוריתם מונטה קרלו הבחנו כי ניתן להמשיך את הפרויקט בכיוון הנ"ל, למידול טוב יותר של הקלפים שנמצאים אצל שאר השחקנים בעזרת אלגוריתמים של פתרון בעיות אילוצים, CSP, ובכך להקטין את מספר הפעמים שבהם נצטרך להגריל קלפים לשחקנים האחרים וגם להתייחס בצורה אמינה יותר לקלפים שהוגרלו להם. ניתן להפיק אילוצים שונים מצורת המשחק של שאר השחקנים, לדוגמא כאשר בשלב המכרז שחקן כלשהו הכריז כי הוא מתכוון לקחת המון קלפים בעזרת סדרה כלשהי, ניתן להניח כי אותו שחקן אכן מחזיק המון קלפים מאותה הסדרה ולכן כאשר נרצה לחלק קלפים נספק לקלפים מהסדרה הזאת הסתברות גדולה יותר שהם יחולקו אליו. בנוסף, כאשר שחקן מסוים עונה על קלף מסדרה כלשהי בעזרת קלף מסדרה אחרת ניתן להסיק בוודאות מוחלטת כי שחקן זה אינו בעל קלפים מהסדרה האשונה ולכן, במידול אותו השחקן באלגוריתם מונטה קרלו, לא נספק לאותו השחקן שום קלף מהסדרה הזאת. אנו סבורים כי מודל כזה אכן יוכל להוריד בצורה ניכרת את זמן החישוב של אלגוריתם מונטה קרלו ובוודאות יבטיח תוצאות לא פחות טובות מהאלגוריתם שאנו ממשנו.

נשים לב כי בכך אנו נתקלים בבעיה של מידול השחקן הרנדומלי, וכניראה עדיין תיהיה בעיה להפיק שחקן חכם המצליח להתמודד עם שחקן רנדומלי, כיוון שכאשר השחקן הרנדומלי מבצע פעולות כלשהן לא ניתן להפיק מכך שום מידע על הקלפים שנמצאים ברשותו או בכלל על אופן הפעולה שלו בהמשך. כמו כן, מידול מדויק יותר של השחקן הרנדומלי כניראה רק יגרום ליותר טעויות וזמן חישוב מהשחקן החושב, כיוון שכעת, המידול שלנו מתקרב למציאות יותר, ובתקווה מתקרב גם לפעולת שחקן אופטימלי, ובכך מתרחק מאופן הצורה של שחקן רנדומלי, כלומר סביר להניח כי דוקא שחקן חכם יותר ישחק בצורה גרועה יותר מול שחקן רנדומלי.