QUORIDOR AI

הדר לייבושור

ת.ז. 209782002

מנחים: אלון חיימוביץ'

תאריך: מאי 2022, תשפ"ב

תוכן עניינים

תקציר

תיאור הנושא

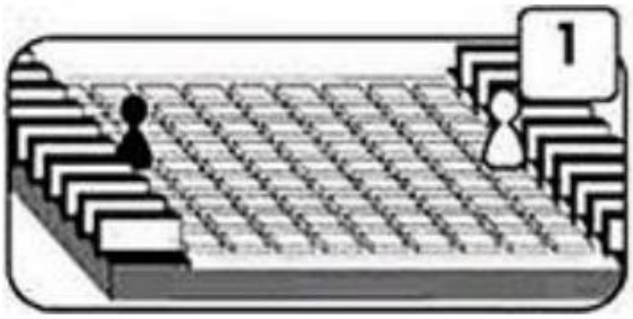
בכל תור השחקן יכול לבחור האם לזוז או האם לחסום.

חוקי המשחק

כעת אסביר מהם חוקי המשחק, אילו מהלכים אפשריים ואילו לא אפשריים, וכיצד המשחק מתקדם.

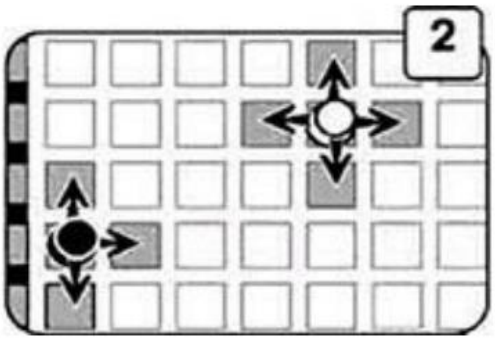
תחילת המשחק

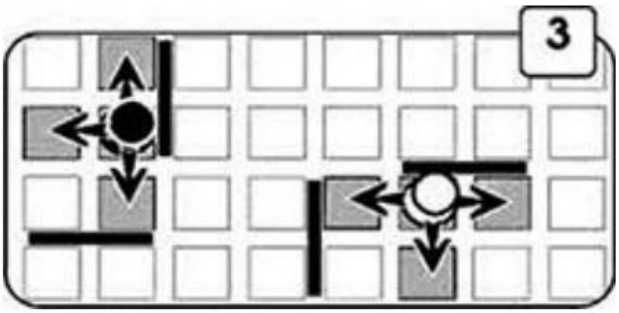
המשחק מתחיל כאשר שני שחקנים עומדים בקצוות שונים של הלוח באמצע השורה:



תזוזת חיילים

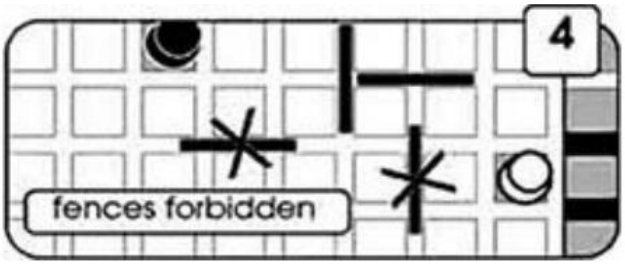
שחקן יכול לזוז בתורו צעד אחד קדימה, אחורה, ימינה או שמאלה במידה והכיוון לא חסום לו:



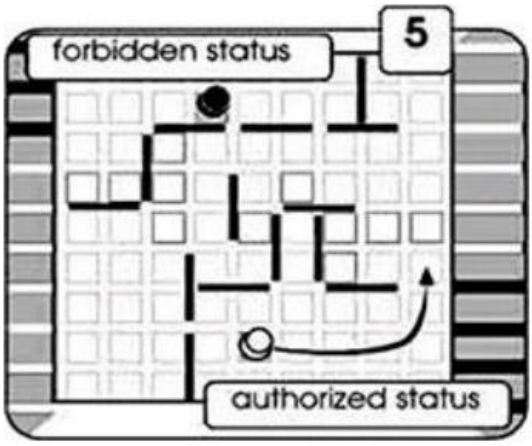


מיקום מחסומים

כאשר שחקן מניח מחסום, על המחסום להפריד בין שני סטים של שני ריבועים:



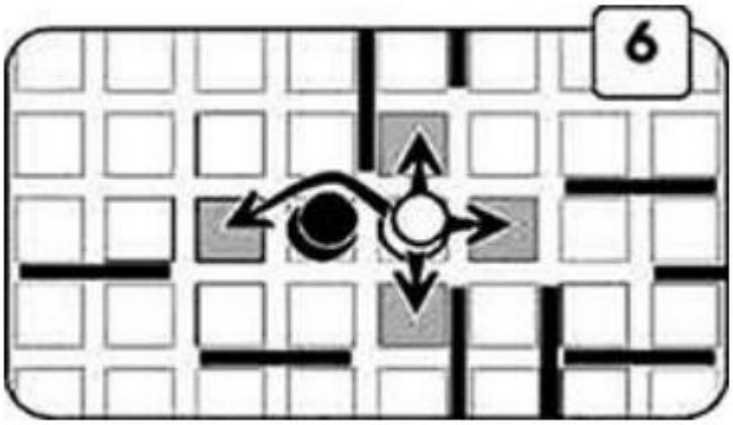
אסור להניח מחסומים בדרך שלא יאפשר פתח יציאה של שחקן (לנעול שחקן):



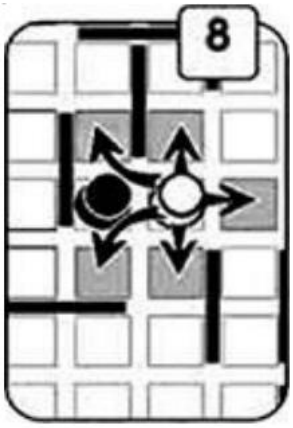
בדוגמא ניתן לראות שהשחקן השחור תקוע על ידי מחסומים
במצב שבלתי אפשרי לצאת ממנו. המצב המתואר לא חוקי

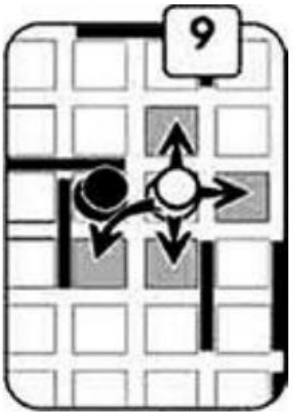
פנים מול פנים

במפגש "פנים מול פנים" בין שני השחקנים, כאשר אין מחסום המפריד ביניהם, שחקן שתורו יכול לקפוץ מעל אויבו בצורה הבאה:



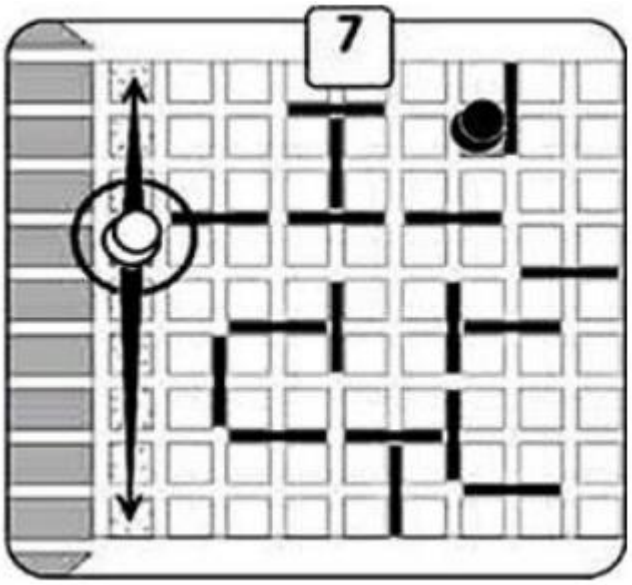
במצב "פנים מול פנים", אם יש מחסום מאחורי האויב, שחקן יכול לקפוץ לצד האויב (כל עוד מחסום לא חוסם תנועה כזאת):





מטרת המשחק

השחקן הראשון שמגיע לשורה שבו האויב התחיל ממנה מנצח, וכך גם נגמר המשחק:



רקע תאורטי

מהו משחק אסטרטגי?

משחק אסטרטגיה מושפט, הוא משחק המתאפיין בתכונות הבאות:

1. אין אלמנט של מזל, כמו למשל זריקת קובייה או בחירת קלף רנדומלי.

2. קיימת ידיעה מלאה של מצב המשחק בזמן אמת, כלומר בכל רגע כל השחקנים יודעים הכול על מצב המשחק.

3. המהלכים מתבצעים בזה אחר זה, לא כמו משחק אבן נייר ומספריים בו שני השחקנים צריכים להראות את ידיהם בו זמנית.

בהתאם לכך, גורל המשחק נקבע רק על פי ההחלטות של השחקנים ולפי החשיבה שלהם.

מהו שחקן ממוחשב?

שחקן ממוחשב (בוט) הוא בעצם תוכנה שמבצעת פעולות אוטומטיות במשחק מחשב בהתאם לאדם המשחק מולו ובהתאם לאלגוריתם שנכתב לו. השחקן הממוחשב ישחק לפי חוקי המשחק המקוריים וינסה לחשוב על כל דרכי הפעולה המתאימות בשביל לנצח בזמן הקצר ביותר ובצורה היעילה ביותר.

אלגוריתם

אלגוריתם הוא דרך שיטתית וחד-משמעית לביצוע של משימה מסוימת, במספר סופי של צעדים. אלגוריתם נועד לפתור בעיה או לבצע משימה על ידי מעקב אחר הוראות ,אלגוריתם משמש לדוגמה עבור חישובים, קבלת החלטות, עיבוד מידע ועוד.

בינה מלאכותית

מצב שבו מדמים את יכולת החשיבה של בן אדם על ידי אמצעים טכנולוגיים, בינה מלאכותית מבצעת פעולות שסיכויהם להשיג את מטרתם גבוהה במיוחד. מערכת מוגדרת כבינה מלאכותית אם היא מצליחה לזהות את מרכיבי סביבתה ומבצעת פעולות המשפרות את סיכויי השגת מטרתה באופן מקסימלי. לדוגמה: רכבים אוטונומים, המלצות לפי העדפות אישיות בתחומים שונים.

אפליקציה

היא סוג של תוכנת מחשב אשר מנצלת את יכולות המחשב ישירות לביצוע משימות אותן המשתמש מבקש לבצע.

תיאור הבעיה האלגוריתמית

הבעיה האלגוריתמית בביצוע הפרויקט היא יצירת אלגוריתם נכון ויעיל שמסוגל לבצע החלטות חכמות בזמן אמת.

*** להרחיב כאן קצת ***

סקירת אלגוריתמים בתחום הבעיה

DFS

אלגוריתם המאפשר לנו לעבור על קודקודי הגרף (במקרה שלנו עץ המשחק) לעומק, עד שמגיעים לעלה (קודקוד בעומק הרצוי שהגדרנו). כאשר נגיע לעלה, ננקד את הלוח באמצעות ה – heuristic function. נוכל לשפר את ביצועיו של אלגוריתם זה באמצעות minimax pruning, alpha beta ו table transposition שעליהם אסביר בהמשך.

MiniMax

אלגוריתם המאפשר למצוא את המהלך הטוב ביותר עבורינו.

האלגוריתם בעצם מעין של "מדמה" את המשחק, כך שבכל שלב במשחק (בכל תור) מחזיר את הציון המקסימלי של המהלך אשר אנו יכולים לבצע, ואת הציון המינימלי עבור מהלכים שהיריב יכול לבצע, וכך בעצם מאפשר למצוא את המהלך הטוב ביותר, מספר צעדים קדימה, בהתייחסות למהלכים שיכול היריב לבצע.

האלגוריתם בעצם יוצר עץ משחק בגובה שהוגדר (העומק אותו אנו מגדירים בקריאה לפונקציה של האלגוריתם), עובר עליו, ומחזיר את המהלך ואת ניקוד המהלך הטוב ביותר אשר אנו יכולים לבצע בהסתכלות של מספר מהלכים קדימה (מספר זה הוא העומק שהגדרנו). אלגוריתם זה הוא כמובן רקורסיבי.

NegaMax

אלגוריתם שהוא בעצם סוג של MiniMax.

אלגוריתם זה מחזיר את אותו הדבר כמו אלגוריתם ה - MiniMax אך עושה זאת עם קוד קצר יותר ומובן יותר לקריאה. זאת מכיוון שבאלגוריתם זה, במקום שתהיה בדיקה של מינימום ומקסימום, בכל פעם בוחרים את המהלך בעל הניקוד המקסימלי ,ובקריאה הרקורסיבית קוראים למינוס אחד כפול הערך שיחזיר ה - NegaMax בעצם ניתן לקצר את הקוד כמעט פי 2 .זאת מכיוון שכאשר אני קורא לאלגוריתם עם מינוס לפני, ובוחר את המקסימום, אני בעצם בוחר את המינימום (מה שצריך לעשות כאשר התור הוא תור היריב, זאת מכיוון שכאשר היריב מבצע את המהלך המקסימלי, כלומר הטוב ביותר עבורו, זה בעצם המהלך הגרוע ביותר שיכל לבצע עבורי - משחק סכום אפס ולכן מה שטוב ליריב בטוח רע בשבילי). אלגוריתם זה בעצם זהה ברעיון לאלגוריתם ה - MiniMax ובאמצעותו ניתן למצוא את המהלך הטוב ביותר שניתן לבצע בהסתכלות לעץ המשחק בעומק מסוים, על ידי הסתכלות על המהלכים האפשריים אשר אני יכול לבצע בכל תור והמהלכים האפשריים אשר היריב יכול לבצע בכל תור.

Alpha Beta Pruning

אופטימיזציה ושיפור של אלגוריתם ה – MiniMax / NegaMax על ידי "גזימת" תתי עצים.

בעצם ישנם מצבים בהם ניתן לדעת מראש שהמהלכים בתת עץ מסוים אינם רלוונטים ושהמהלך הטוב ביותר אשר אנו יכולים לבצע אינו נמצא בתתי עצים אלו. זה נעשה על ידי התייחסות לעוד שני פרמטרים, alpha ו – beta. כאשר alpha מתאר את החסם העליון (ניקוד המהלך הטוב ביותר של השחקן) של מהלך שהשחקן יכול לבצע, ו beta אשר מתאר את החסם העליון ביותר של מהלך שהיריב יכול לבצע. כאשר alpha גדול מ – beta ניתן לדעת כי תת העץ אשר אנו בודקים אינו יכול להביא תוצאה אשר תשנה את alpha או beta ולכן ניתן "לגזום" אותו, כלומר להפסיק לבדוק אותו.

אלגוריתם זה כמעט תמיד משפר משמעותית את הביצועים של ה - MiniMax או ה – NegaMax, ובמקרה הגרוע ביותר פועל באותה סיבוכיות כמוהם.

Transposition Table

מבנה נתונים המאפשר לנו לשמור מעין cache של מצבים בלוח שכבר בדקנו, וכך בעצם לשפר את יעילות החיפוש של המהלך הטוב ביותר.

זאת מכיוון שכאשר אנו שומרים מצב של הלוח אותו אנחנו בודקים במעבר על עץ המשחק, ואנו מגיעים אליו פעם נוספת, כבר נדע מה המהלך הטוב ביותר לבצע במצב זה של הלוח ואת ה score של הלוח באותו מצב, ולא נצטרך לבצע את החישובים שוב, דבר אשר ישפר את היעילות באופן משמעותי.

באמצעות שמירת המהלכים נוכל גם באופן מידי לעדכן את הציון שניתן למצב הלוח ואת ערכי ה - alpha ו - beta הנוכחיים. עבור כל מצב שנגיע אליו בלוח והוא לא נמצא כבר ב – Transposition Table, נכניס אותו אליה, בכדי שפעם הבאה שניגע למצב כזה בלוח, לא נצטרך לחשב דברים מחדש.

מימוש ה – Tansposition Table נעשה באמצעות מילון, כאשר הערך הוא המהלך הטוב ביותר, הציון שלו, העומק בו עברתי על מצב לוח זה, ודגל המייצג את היחס של ציון המהלך ל – alpha ו – beta. המפתח של כל entry במילון הוא מפתח הנוצר כפונקציה של מצב לוח הנוכחי, ונקרא Zobrist Key. מפתח זה מיוצר על ידי נתינת ערך מספרי בגודל 64 ביט לכל משבצת על הלוח, לכל חייל על הלוח, ולכל אחד מהשחקנים, ולאחר מכאן ביצוע פעולת xor בין כל ערכי המשבצות (האקראיים) אשר עליהן יש חתיכות, עם ערכי החתיכות (האקראיים) ועם ערכי השחקן (הצבע של החתיכה, האקראיי). ביצוע פעולות xor אלו בין כל המקומות על הלוח בהן יש חתיכות משחק, מחזיר מספר, שהוא בעצם המפתח של מצב הלוח.

Iterative Deepening

שיטה המאפשרת לנו לרוץ לעומק רב יותר, ומאפשרת לנו יותר גמישות בריצה לעומק במהלך המשחק.

השיטה עובדת כך, קובעים פרק זמן בו רוצים ששחקן המחשב יבצע את המהלך, וכל עוד פרק זמן זה עוד לא עבר, מריצים את פונקציית מציאת המהלך הטוב ביותר ובכל פעם מגדילים את עומק החיפוש עד שפרק הזמן עובר, כך בעצם ניתן להגיע לעומק מקסימלי, וניתן להיות גמישים, במצבים בהם ישנם מהלכים אפשריים רבים, נרוץ לעומק נמוך יותר בהשוואה לכאלה שבהם מספר המהלכים האפשריים רב יותר וניתן לרוץ לעומק רב יותר.

Killer Heuristic

הרעיון מאחורי אלגוריתם זה, הוא שאם ישנו מהלך אשר גזם ענף של עץ ברמה מסוימת, ישנו סיכוי גבוה שיכול לגזום גם ענפים אחרים באותה הרמה, ובכך להקטין בעצם את כמות הענפים שאותם ממשיכים לבדוק, ובכך להקטין את מספר המהלכים אותם בודקים, דבר אשר יכול לשפר באופן משמעותי את זמן הריצה של אלגוריתם ה – Mini / Nega Max.

הרעיון הוא שאם מהלך מסוים במצב לוח מסוים גזם ענף, אותו מהלך יוכל לגזום ענפים נוספים באותו הרמה כאשר מצב הלוח דומה למצב בו המהלך גזם.

מושגים

עץ משחק

מושג בתיאוריית המשחקים, המשמש לתיאור גרף בו קיימים כל מצבי הלוח האפשריים במשחק, כאשר כל קודקוד הוא בעצם מצב לוח אפשרי, כאשר המסלול לכל קודקוד מהקודקוד הנוכחי ,מייצג את המהלכים שיש לבצע בכדי להגיע למצב זה של הלוח.

ניתן להשתמש בעץ המשחק בכדי לחשב סיבוכיות של משחק, מכיוון שככל שישנם יותר מצבים אפשריים, המשחק יותר מסובך.

משתמשים בעץ המשחק בכדי למצוא את המהלך הטוב ביותר מנקודה מסוימת. האלגוריתמים Mini / Nega Max שעליהם הסברתי בסעיף הקודם, עוברים על עץ המשחק (לפעמים רק עד עומק מסוים, מכיוון שכמות המצבים עצומה והזמן שייקח לעבור על כולם רב מדי בשביל שהמחשב יוכל לחשב באופן ריאלי מהלך). כמות העלים בעץ משחק, היא כמות המצבים הסופיים האפשריים לסיום המשחק מהמצב הנוכחי. מספר הקודקודים בעץ המשחק הוא עצום, במשחק פשוט מאוד כמו איקס עיגול, כמות העלים היא 168,255 ולכן במשחקים כמו שחמט או שחמט סיני לא ניתן לעבור על כל המהלכים ועוברים על העץ רק עד לעומק מסוים.

שחקן ממוחשב

שחקן AI אשר מחשב בכל תור את המהלך הטוב ביותר שיכול לבצע, ומבצע אותו. השחקן הממוחשב הוא Responsive. כלומר, בכל רגע מתייחס למה שקורה בלוח ומגיב בהתאם. השחקן הממוחשב חכם, ומסתכל קדימה מספר צעדים בכל תור בכדי לקרב אותו כמו שיותר לניצחון. שחקן ממוחשב יכול להיות בכל מני רמות. כיום במשחק השחמט, השחקנים הממוחשבים הטובים ביותר, טובים משחקני השחמט האנושיים הטובים ביותר.

Heuristic Function

פונקציה המשומשת במשחקי מחשב בכי לתת ניקוד למצב המשחק בכל רגע נתון, ובאמצעות פרמטרים שונים שאליה מתייחסת, להעריך את מצב המשחק.

לרוב הפונקציה מחזירה מספר, שככל שהוא גבוה יותר, מצב הלוח של השחקן שעבורו מנקדת את הלוח, טוב יותר. לרוב משתמשים בפונקציה זו על עלים בעץ המשחק, כמו שעושים באלגורתמים כמו ה – Mini / Nega Max שעליהם הסברתי קודם.

ככל שלפונקציית ה - heuristic יותר פרמטרים והיא מתייחסת ליותר דברים על הלוח, היא מחזירה ציון מדויק יותר של הלוח.

אסטרטגיה

מבנה נתונים

תרשים מחלקות

Use cases

UML

Top-Down Level Design

*** תרשים היררכי של החלוקה של הפרויקט הגדול לתתי המשימות הקטנות, כלומר החלוקה למודולים ולפונקציות אליהם חילקת את העבודה על מנת בסופו של דבר לפתור את הבעיה הגדולה. צריך להראות כמו עץ, עד עומק 3 בערך ***

תיאור סביבת העבודה ושפת התכנות

שפת התכנות

בפרויקט השתמשתי במספר שפות תכנות.

עבור הלקוח, ה – GUI, השתמשתי ב – HTML, CSS ו – JavaScripts.

עבור השרת והלוגיקה של המשחק השתמשתי ב – Java.

סביבת העבודה

Operating System

- Microsoft Windows 10

Code Editor

- Visual Studio Code
  - version 1.66.2 (user setup)

JDK

- jdk 17

תיאור ממשקים

org.glassfish

ממשק המאפשר יצירת שרת של Web sockets ב – Java.

javax.websocket

ממשק המאפשר תקשורת באמצעות Web sockets ב – Java.

com.google.gson

ממשק המאפשר ניהול ושימוש נוח של שליחה וקבלה של Json בצד השרת.

*** אולי להוסיף רק על הממשקים של כל מבני הנתונים שהשתמשת ***

אלגוריתם ראשי

פונקציות ראשיות

מדריך למשתמש

כעת אסביר מהם השלבים הנדרשים על מנת שתוכלו להריץ את הפרויקט על מחשבכם:

1. וודאו שיש לכם JDK, בפיתוח הפרויקט השתמשתי בגרסא 17.
2. העתיקו את קוד הפרויקט למחשבכם מ – GitHub: https://github.com/hadarLeib/Quoridor.git
3. הריצו את הסקריפט server.bat ב – cmd על מנת להריץ את הסרבר.
4. פתחו את הקובץ index.html בדפדפן האהוב עליכם על מנת להתחיל לשחק!

וזהו 😀

*** אולי להוסיף כאן קצת ***

קוד הפרויקט

GitHub

להלן קישור לקוד הפרויקט ב – GitHub: https://github.com/hadarLeib/Quoridor.git

קוד

סיכום אישי

ביבליוגרפיה

There are no sources in the current document.

נספחים