קבלת החלטות בזמן אמת עם Monte Carlo Tree Search במשחק Ms. Pac-Man

יאיר לוי, אלה שלום

המשחק Ms. Pac-Man הוא משחק Arcade המבוסס על המשחק Pac-Man שיצא בשנת 1980. במשחק השחקן נדרש לבצע קבלת החלטות בזמן-אמת, כאשר מטרתו העיקרית היא להשיג כמה שיותר נקודות. המשחק כולל מספר מבוכים, בהם פאק-מן רשאי לנוע בחופשיות, ובו זמנית ארבע רוחות שרודפות אחריו במטרה למנוע ממנו מלהשיג ניקוד. במסמך זה נשווה בין אלגוריתם MCTS שנלמד בקורס, לבין שיטות אופטימיזציה שונות, במטרה לסייע לפאק-מן שלנו להשיג תוצאות טובות ככל הניתן.

תיאור הבעיה

המטרה העיקרית של משחק הפאק-מן היא להשיג כמה שיותר נקודות. נקודות ניתן להשיג באמצעות אכילת pills המפוזרים ברחבי המבוך, או באמצעות אכילת רוחות מפוחדות לאחר אכילת power pill. כמצוין לעיל, לצד פאק-מן ישנן ארבע רוחות עוינות שמטרתן היא לחסל אותו, ובכך למנוע ממנו נקודות נוספות. בסביבה שלנו, נרצה למקסם את התועלת של פאק-מן ממעבר במסדרוני המבוך, תוך מזעור של הנזק הנגרם מצד הרוחות.

אחד האתגרים הגדולים ביותר – אם לא הגדול ביותר – שנתמודד איתו במשחק הוא יצירת האיזון בין תכנון ארוך-טווח לבין תכנון קצר-טווח. במצבים רבים במהלך המשחק, הדרך האופטימלית של פאק-מן תסתור את הפרס המידי שיכול לקבל בטווח הקצר, ולכן נצטרך למצוא דרך פתרון שכוללת בטוחה מעט גמישות ו-"חיזוי" של העתיד, תוך איסוף כמה שיותר נקודות בדרך.

גישות הפתרון

האלגוריתם הראשי אותו בחרנו לחקור הוא לא אחר מאשר האלגוריתם המפורסם Monte Carlo Tree Search (MCTS). אלגוריתם MCTS נחל הצלחה רבה בקרב משחקי Real-Time, וגם בקטגוריות משחקים נוספות כמו פאזלים, משחקי קלפים ומשחקי קופסא, מה שהופך אותו למועמד מוביל להתמודדות בסביבת משחק פאק-מן.

בנוסף למוניטין של MCTS בתור אלגוריתם שמניב תוצאות טובות במשחקים הדורשים קבלת החלטות בזמן-אמת, בחרנו בו מאחר והוא מצליח להתמודד עם האתגר שאותו ציינו, והוא האיזון בין תכנון ארוך-טווח לקצר-טווח. הודות לשלב הסימולציה של האלגוריתם, אם יש סתירה בין קבלת ניקוד בטווח הקצר לבין איכות המסלול לאחר מספר צעדים נוספים, MCTS יידע לתת את ההערכה האופטימלית לקבלת החלטה זו.

אלגוריתם נוסף אותו נרצה לנסות הוא Minimax. בדומה לאלגוריתם MCTS, האלגוריתם הזה גם מבצע קבלת החלטות באמצעות עץ עם מהלכים אפשריים, אך מאחר והוא בוחן את כל האפשרויות, הגובה של העץ שלו צריך להיות קטן יותר משמעותית מזה של MCTS, מאחר ובמשחק הפאק-מן יש הגבלת זמן על קבלת ההלטה לבצע מהלך. הגבלה זו מעידה על כך ש-Minimax מדגיש יותר את הטווח הקרוב מאשר את הטווח הרחוק, אם כי באופן מדויק יותר מ-MCTS.

כדי לנסות לעקוף את בעית זמן החישוב, ננסה להשוות גם את האלגוריתם Alpha-Beta Pruning, שמטרתו לספק תוצאות זהות לאלו של Minimax, עם פחות זמן חישוב.

לבסוף, שיטה אחרונה בה נשתמש תהיה היוריסטיקה. בפשטות, בכל נקודה בה פאק-מן צריך להחליט איזה מהלך לבצע, נסמלץ כל מהלך מבין האפשריים כרגע, ניתן למצבי המשחק החדשים ציון, וניקח את המהלך בעל הציון הגבוה ביותר. שיטה זו שמה את כל משקלה על התכנון לטווח הקצר, ובכלל לא מתייחסת לטווח הארוך.

מה אנו מנסים להשיג?

בפאק-מן צריך לקבל נקודות עד כמה שניתן. נרצה לבצע אופטימיזציה על המהלכים אותם פאק-מן מבצע, כדי לאפשר לו לעשות זאת. שיטות האופטימיזציה השונות שבחרנו דומות בחלקן, אך גם שונות. נרצה לראות איך הן משפיעות על פעילות סוכן בסביבה מורכבת, תוך השקעת מאמצים להשיג מטרה כלשהי, ומתוך כך לנסות להבין האם ניתן ללמוד מכך על בעיות נוספות עם תכונות דומות לזו שלנו.