פאזל 15 - 10 מהלכי ערבוב:

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 37 | 48 | 4 | 8 |
| אורך הפתרון | 5 | 5 | 5 | 5 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 65 | 72 | 4 | 6 |
| אורך הפתרון | 5 | 5 | 5 | 5 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 15 | 0 | 0 | 0 |
| מצבים שנבדקו | 1440 | 777 | 8 | 14 |
| אורך הפתרון | 9 | 9 | 9 | 9 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 45 | 48 | 4 | 7 |
| אורך הפתרון | 5 | 5 | 5 | 5 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 37 | 70 | 4 | 5 |
| אורך הפתרון | 5 | 5 | 5 | 5 |

פאזל 24 - 10 מהלכי ערבוב:

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 48 | 79 | 0 | 0 |
| מצבים שנבדקו | 9375 | 9640 | 19 | 30 |
| אורך הפתרון | 11 | 11 | 11 | 11 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 9 | 5 | 2 | 3 |
| אורך הפתרון | 3 | 3 | 3 | 3 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 394 | 189 | 6 | 18 |
| אורך הפתרון | 7 | 7 | 7 | 7 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 494 | 226 | 6 | 11 |
| אורך הפתרון | 7 | 7 | 7 | 7 |

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0 | 0 | 0 | 0 |
| מצבים שנבדקו | 6 | 8 | 2 | 5 |
| אורך הפתרון | 3 | 3 | 3 | 3 |

פאזל 15 ערבוב 10 - ממוצע 50 הרצות:

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0.58 | 1.54 | 0 | 0 |
| מצבים שנבדקו | 311.9 | 311.88 | 4.62 | 7.92 |
| אורך הפתרון | 5.2 | 5.2 | 5.2 | 5.2 |

פאזל 24 ערבוב 10 ממוצע 50 הרצות:

|  | BFS | dijkstra | manhattan | incompatible heuristic |
| --- | --- | --- | --- | --- |
| זמן ריצה (מילי שניות) | 0.92 | 1.26 | 0 | 0.02 |
| מצבים שנבדקו | 265.66 | 286.94 | 4.08 | 7.26 |
| אורך הפתרון | 5 | 5 | 5 | 5 |

## מסקנות:

| BFS | נראה שBFS פחות מתאים לבעיה זו.    מכיוון שבעיה זו גדולה מידי למיפוי כל האפשרויות, BFS שהוא אלגוריתם מיפוי ולא עם מטרה להגיע ליעד מסויים פחות יעיל בחיפוש אחר פתרון ספציפי.  האלגוריתם יעבוד בצורה הבאה: תחילה יעבור תחילה על כל המצבים שניתן להגיע אליהם במהלך 1, אחר כך ב2, אחר כך ב3, וכו… עד שיגיע למצב הפתרון.  חוסר האינדיקציה של BFS לגבי האם הוא מתקדם בכיוון הנכון, גורם לו לעשות סיבובים מיותרים ולבדוק אופציות ששחקן אנושי היה משאיר בצד ומתמודד איתם אם וכאשר הן יהיו רלוונטיות.  לכן, ככול שהדרך לפתרון המינימלי תיהיה יותר ארוכה, כמות הבדיקות שיריץ אלגוריתם BFS תיהיה גדולה יותר באופן אקספוננציאלי ולכן הוא פחות מתאים. |
| --- | --- |
| dijkstra | תחילה חשבתי שdijkstra יבצע טוב יותר מBFS ברוב המקרים.  הופתעתי לגלות שלא כך היה המצב (בכמויות ערבוב נמוכות).  לאחר העמקה בשאלה מדוע? הבנתי.  dijkstra - הוא אלגוריתם חמדן, שכל הזמן לוקח את הקשת ה"קלה" ביותר. בתרגיל זה משקל כל קשת שווה למספר המהלכים שלקח להגיע למצב הלוח המוצג. בגלל שמשקל הקשתות מחושב כך, נוצר מצב בו Dijkstra יעבור תחילה על כל המצבים שניתן להגיע אליהם במהלך 1, אחר כך ב2, אחר כך ב3, וכו… עד שיגיע למצב הפתרון, כלומר אם הפתרון האידיאלי לוקח 5 מהלכים, dijkstra יבדוק קודם את כל הלוחות שאפשר להגיע אליהם ב4 מהלכים לפני שיבדוק את הלוח הראשון שמגיעים אליו ב5.  ובעצם הוא סוג של BFS - ופחות מתאים לבעיה. |
| A\*- manhattan | אין ספק שאלגוריתם A\* עם מנהטן בתור הפונקציה היוריסטית היה האלגוריתם הטוב ביותר והיעיל ביותר לפתירת הבעיה.  בשונה מdijkstra אלגוריתם A\* בוחר באיזה כיוון הכי כדאי לו להתקדם בעזרת חישוב שכולל בתוכו לא רק כמה מהלכים לקח להגיע למצב ספציפי ומה המהלך הבא הזול ביותר, אלא גם ניקוד המייצג כמה המצב רחוק ממצב הפתרון.  חישוב זה (שנקרא הפונקציה היוריסטית) מקנה לאלגוריתם מעין "אינטואיציה" מהו הכיוון הנכון להתקדם בו כדי להגיע לפתרון, וככה בהינתן והחישוב נכון וקביל, ה"אינטואיציה" תהיה נכונה, האלגוריתם יגיע לפתרון בכמות צעדים נמוכה, ללא סיבובים מיותרים.  פונקציית מנהטן - עוברת על כל לוח המשחק ומדרגת אותו לפי חישוב של כמה מהלכים נדרשים כדי להעביר כל משבצת למקומה הסופי בפתרון. ככה לוחות שמהצבם קרוב יותר לפתרון, יקבלו ניקוד נמוך מלוחות רחוקים מהפתרון. (\*ניקוד נמוך עדיף מגבוה)  מכיוון שפונקציה זו, נותנת"אינטואיציה" נכונה האלגוריתם עובד בצורתו האופטימלית. |
| A\*-  incompatible heuristic | כמו שהסברתי במסקנות של A\*-manhattan ביצועיו של אלגוריתם A\* תלויים בקשר ישיר באיכות ה "אינטואיציה" שלו שנקבעת באמצעות הפונקציה היוריסטית.  בסעיף זה התבקשנו לתת פונקציה יוריסטית לא קבילה - משמע "אינטואיציה" לא מדוייקת - דבר שיגרור שהאלגוריתם לא יעבוד בצורה אופטימלית.  תחילה - פשוט נתתי מספר רנדומלי לחלוטין לכל מצב, זה יצר מצב בו בחלק מהמקרים מהלכים טובים נתקעים בעדיפות מאוד נמוכה ולעולם לא משוחקים. כשניסיתי להריץ את האלגוריתם, במקרים רבים הוא לא הצליח לפתור את המשחק ונתקע בתוך אוסף האפשרויות העצום.  בכדי להצליח לפתור את הפאזל החלטתי להשמתמש בפונקציית הmanhatan שקיימת ולשבש אותה קצת, ככה רוב "האינטואיצה"  תהיה נכונה, ואוכל לשלוט כמה אני יכול "לבלבל" את האלגוריתם ולגרום לו לבחור אופציה לא נכונה.  במימוש המוגש - הוספתי לmanhattan (שמייצגת כמה מהלכים צריך על מנת להגיע מהמצב הנוכחי למצב הפתרון) מספר רנדומלי בין 0 ל 5. הוספה זאת יוצרת "בלבול" מסוים וכתוצאה ממנו יכול להיות שהאלגוריתם יעדיף לבדוק לוח במרחק 10 מהלכים מהפתרון מלוח במרחק של 8 מהלכים מהפתרון.  מכיוון שה"בלבול" הוא יחסית קטן( 0-5), האלגוריתם עובד טוב יחסית ומתקדם בכיוון הנכון אך לא תמי בדרך האופטימלית.  ככול שנגדיל את טווח המספרים הרנדומלים ככה האלגוריתם יהיה יותר ויותר מפוזר ופחות יעיל. |