תרגיל ריצה - חיפוש

הבעיה

עליכם לממש מנוע חיפוש התומך במספר אלגוריתמי חיפוש כדי לפתור את משחק ה- Extended NxM-tile puzzle (הכללה של ה- tile-puzzle שראינו בשיעור).
במשחק נתון לוח בגודל NxM המכיל NxM-1 בלוקים הממוספרים מ- 1 ועד NxM-1 ובלוק ריק, **או** שהלוח מכיל NxM-2 בלוקים הממוספרים מ- 1 ועד NxM-2 ושני בלוקים ריקים. הבלוקים מסודרים בסדר התחלתי נתון כלשהו, והמטרה היא למצוא את מספר הפעולות הזול ביותר מהסידור ההתחלתי למצב הסופי, שנתון אף הוא.

הפעולות
ניתן להזיז כל בלוק שנמצא בסמוך לבלוק הריק, ועלות פעולה זאת היא 5. אומנם, במידה ובלוח הנתון יש 2 בלוקים ריקים, קיימת במשחק זה פעולה נוספת- הזזה של 2 בלוקים במקביל. פעולה זו מותרת רק במקרה בו 2 הבלוקים הריקים צמודים אחד לשני, במאונך או במאוזן. הזזה של 2 בלוקים במקביל במאוזן עולה 6 והזזה של 2 בלוקים במקביל במאונך עולה 7.
לדוגמה, אם הלוח במצב הזה:

1	2	3	4
5		6	
9	10	7	8

נוכל להזיז את 6 שמאלה ואז את 7 ו- 8 ביחד למעלה, כדי להגיע למצב הסופי. עלות המסלול המתואר תהיה 5+7=12.

מימוש

קלט
התוכנית תקרא את כל הקלט שלה מקובץ יחיד- input.txt. השורה הראשונה בקובץ תקבע באיזה אלגוריתם להשתמש: BFS, DFID, A*, IDA*, או DFBnB. השורה השנייה בקובץ תקבע האם להדפיס את זמן הריצה (with time) או לא (no time). השורה השלישית תקבע האם להדפיס למסך את ה- open list בכל שלב של ריצת אלגוריתם החיפוש (with open) או לא (no open). השורה הרביעית תכיל את גודל הלוח בפורמט הבא: NxM, ז"א לוח המכיל N שורות ו- M עמודות. לאחר מכן יופיע הסידור ההתחלתי של הלוח לפי שורות, כאשר יש פסיקים בין מספרי הבלוקים. הבלוק הריק יסומן כ- "_".
לאחר מכן תופיע שורה בה יהיה כתוב "Goal state:" ולאחריה יופיע הסידור הסופי של הלוח אליו צריך להגיע (באותו פורמט של הסידור ההתחלתי). ניתן להניח שקובץ הקלט תקין.

פלט
במידה ונכתב בקובץ הקלט no open, כל הפלט ייכתב לקובץ output.txt. אין להדפיס במקרה כזה שום דבר על המסך.
בשורה הראשונה בקובץ יש לכתוב את סדרת הפעולות שנמצאה ע"י האלגוריתם. בשורה השנייה יש לכתוב "Num: " ואח"כ את מספר הקודקודים שיוצרו. יש לספור גם קודקודים שלא נכנסו ל-open list, ואם קודקוד נוצר כמה פעמים יש לספור כל פעם בה הוא נוצר. בשורה השלישית יש לכתוב "Cost: " ואח"כ את עלות הפתרון שנמצא. אם בקובץ הקלט נכתב שיש להדפיס גם את זמן הריצה, בשורה הרביעית יש לכתוב את הזמן שלקח לאלגוריתם למצוא את הפתרון (בשניות).
הפעולות יסומנו על ידי מספר הבלוק שזז וכיוון ההזזה: R (ימינה), D (למטה), L (שמאלה), U (למעלה). במידה ושני בלוקים זזו יחד יש להפריד בניהם באמצעות &. הפעולות עצמם יופרדו ע"י מקף. לדוגמה, המסלול המתואר קודם ייכתב בקובץ הפלט כ- 6L-7&8U.

על מנת לקבל פלט אחיד ככל שניתן, נקבע שסדר יצירת הקודקודים בעלי אב משותף יהיה לפי האופרטור שיצר אותם בסדר הבא: 2 בלוקים שמאלה, 2 בלוקים למעלה, 2 בלוקים ימינה, 2 בלוקים למטה. אם לא ניתן להזיז 2 בלוקים הסדר יהיה: שמאלה, למעלה, ימינה, למטה, אל הבלוק הריק שנמצא קרוב יותר לשורה הראשונה ואז אל הבלוק הריק השני. אם שני הבלוקים באותה שורה תהיה עדיפות לפעולות אל הבלוק השמאלי יותר. לדוגמה, אם הלוח במצב כזה:

1	2	3	4
5		6	10
9		7	8

המצב הראשון שנייצר יהיה ע"י הפעולה 6&7L. המצב הבא ייווצר ע"י הפעולה 5&9R, אח"כ ע"י 6L, 5R, 2D, 7L, ולבסוף 9R. אם הלוח במצב כזה:

1	2	3	4
5		6	
9	10	7	8

המצב הראשון שנייצר יהיה ע"י הפעולה 6L, המצב הבא ייווצר ע"י הפעולה 10U, אח"כ ע"י 5R, 2D, 8U, 6R, ולבסוף 4D.

בנוסף, נחיל יחס סדר על קודקודים בעלי ערך זהה בפונקציית ההערכה f(n), לפי זמן הייצור שלהם. זאת אומרת, נניח שיש שני קודקודים a,b בעלי אותו ערך של f(n), וכרגע A* צריך לבחור אחד מהם (כי ערך ה- f(n) שלהם הוא הקטן ביותר בתור העדיפויות). במקרה כזה A* יבחר את a אם הוא נוצר באיטרציה השנייה ו- b נוצר באיטרציה הרביעית, או אם a ו- b נוצרו באיטרציה השנייה אבל a נוצר ע"י האופרטור למעלה, ו- b נוצר ע"י האופרטור ימינה. כך גם ב- DFBnB, שממיין את הקודקודים לפי ערכי f(n), אם יש מספר קודקודים בעלי אותו ערך הם יסודרו לפי זמן הייצור שלהם.

במידה ונכתב בקובץ הפלט with open, הפלט ייכתב לקובץ ה- output בדיוק כמו מקודם, אלא שבנוסף יש להדפיס **למסך** את התוכן של ה- open-list בכל איטרציה של האלגוריתם (=לפני כל הוצאה מה- open list). אני לא מגדיר פורמט מדויק (כי האופציה הזאת היא יותר בשביל debug) אבל אני מצפה לראות בצורה ברורה איך נראה הלוח בכל אחד מהמצבים שנמצאים ב- open-list.

בנוסף יש להגיש קובץ וורד details.docx. בתחילת הקובץ יש לכתוב את פרטי המגיש (שם ות.ז.). לאחר מכן יש לתאר במילים את הפונקציה היוריסטית בה בחרתם להשתמש ולהוכיח בצורה פורמאלית מדוע היא admissible ו- consistent.

דגשים

- BFS ו- A* ימומשו עם closed list. יש להשתמש ב- hash-table גם עבור ה- open list כמו שלמדנו.
- IDA* ו- DFBnB ימומשו עם מחסנית וללא closed-list אך עם loop-avoidance, ז"א בדיקה האם הקודקוד המפותח נמצא על הענף שעליו אנחנו עובדים או כבר במחסנית.
- DFID ימומש בצורה רקורסיבית, ללא closed-list אך עם loop-avoidance.
- אם לא נמצא מסלול יש לכתוב: "no path" בשורה הראשונה של קובץ הפלט. בשורה השנייה יש לכתוב " Num:" ואח"כ את מספר הקודקודים שיוצרו.
- ב- DFID האיטרציה הראשונה היא כאשר l=1, כי ברור שהמצב ההתחלתי אינו המצב הסופי.
- למרות שהמטרה שלנו היא מציאת המסלול הזול ביותר, BFS ו- DFID לא ימצאו בהכרח את המסלול הזול ביותר אלא את המסלול הקצר ביותר (=עם הכי פחות פעולות הזזה). עדיין בקובץ הפלט יש להחזיר את העלות של המסלול שהם מצאו (ולא את מספר הצעדים של המסלול).
- יש לממש את האלגוריתמים לפי מה שלמדנו בכיתה. בפרט, אין לבצע פעולה ומיד אחריה את הפעולה ההופכית לה (כמו הזזת בלוק ימינה ואז החזרתו שמאלה, או הזזת 2 בלוקים למעלה ואז הזזתם בחזרה למטה) בנוסף, אין לבצע פעולה של הזזת 2 בלוקים ואז את הפעולה הופכית של הזזת בלוק אחד (כמו הזזת 2 בלוקים למעלה ואז החזרת אחד מהם למטה).
- כדי לא לאבד סתם נקודות, הקפידו על פלט בדיוק לפי ההוראות: רווחים, אותיות גדולות, 4x5 ולא 4X5, וכו'.

אופן הניקוד

- קוד נכון, שמממש את האלגוריתמים כמו שנלמדו בכיתה, ומחזיר את התוצאה המבוקשת על כל הקלטים החוקיים.
- איכות הפונקציה היוריסטית בה בחרתם להשתמש ב-A*, IDA* ו- DFBnB (זו כמובן אותה הפונקציה), ונכונות ההוכחה שהפונקציה היא admissible ו- consistent.
- קוד מתועד וקריא (שמות משתנים ופונקציות משמעותיים).
- הגשה בזמן.

פרטי ההגשה

- ההגשה ביחידים בלבד. תתבצע בדיקת העתקות.
- ניתן לכתוב את התוכנית ב- Java בלבד, והיא צריכה להתקמפל ולרוץ בגרסת 1.8. שם המחלקה בה נמצאת פונקציית ה- main יהיה Ex1. יש להשתמש ב default-package בלבד (ללא תתי תיקיות). חובה להגיש את קבצי המקור.
- אין לממש GUI.
- עליכם להניח שקובץ ה- input.txt (שאתם מקבלים כקלט) נמצא באותה ספריה בה נמצאת התוכנית, ולכן אין לקרוא את המיקום שלו כארגומנט או לציין ספריה ספציפית בקוד שאתם מגישים (במידה וכן, ירדו על כך נקודות).
- קובץ ה- output.txt (שאתם מוציאים כפלט) צריך להיכתב באותה ספריה בה נמצאת התוכנית, ולכן אין לקרוא את המיקום שלו כארגומנט או לציין ספריה ספציפית בקוד שאתם מגישים (במידה וכן, ירדו על כך נקודות). יש להניח שהקובץ לא קיים, עליכם ליצור אותו בספריה בו התוכנית רצה, ולכתוב לתוכו את הפלט.
- יינתן קלט ופלט לדוגמה. ודאו שתוכנתכם עובדת אתו כמו שצריך, אך זהו לא הקלט היחיד אתו תיבדק התוכנית.
- התוכנית תיבדק דרך ה- command line ולא ב- eclipse. לכן, כדי לוודא שהתוכנית שלכם עובדת עליכם להעתיק את קבצי המקור ואת הקובץ input.txt שניתן כדוגמה לאחת הספריות במחשב, לפתוח command line ולהריץ javac *.java ואז java Ex1. התוכנית תיצור את הקובץ output.txt באותה ספריה והוא צריך להיות זהה לקובץ output.txt שניתן כדוגמה.
- ההגשה נעשית דרך מערכת הגשות submit בכתובת http://submit.org.il/ariel.
  שם המשתמש שלכם הוא שם המשתמש שהוקצה לכם באי-מייל שלכם באריאל, כמו שכתוב במכלול. לדוגמה, אם הדואר שלכם במכלול הוא israel.israeli@msmail.ariel.ac.il, אז שם המשתמש שלכם הוא: israel.israeli.
  הסיסמה היא 4 הספרות האחרונות של תעודת הזהות.
  ברגע שתגישו את המטלה תקבלו למייל הפרטי שלכם שרשום במכלול (לא המייל באריאל) את תוצאות ההגשה- האם התרגיל התקמפל, ואם כן מה היה הפלט שלכם בהשוואה למה הפלט הרצוי. תוכלו כמובן להגיש שוב ולדרוס את ההגשה הקודמת. מכיוון שכתובת המערכת לא מעודכנת ב- DNS של אריאל, המייל מהמערכת יגיע (ככל הנראה) לתיקיית הספאם. תבדקו שם וסמנו אותו כ- "not spam". בנוסף, המערכת לא מצליחה לשלוח מייל ל- walla.
- תאריך הגשת התרגיל- 15.05.21

שימו לב: אין לחכות לרגע האחרון כדי להגיש! לקראת סוף הזמן ההגשה השרת עמוס, ולכן ה- feedback מתקבל בשלב מאוחר יותר (אחרי שתאריך ההגשה כבר עובר).

בהצלחה!