תכנות מונחה עצמים

מטלה 0 חלק א'

מגישים: אמיר סבג 316049311, אורי דרשן 212458244

קישור לגיthub של הפרויקט:

https://github.com/amiramir96/OOP-Ex0

סקירת ספרות:

הסבר כללי ביוטיוב על אלגוריתמי מעליות וגם איך ניתן ליישם מימוש בOOP

https://www.youtube.com/watch?v=siqiJAJWUVg&list=PLK8IOvtbwVssQ59IEL73S2ucQdTLiSZC6&index=2

הסרטון שם דגש על תכנות מונחה עצמים וכיצד ליישם פתרון בצורה נכונה, מטרת הסרטון היא הכנת הצופים לראיונות עבודה בהם נשאלים על תכנות מונחה עצמים.

שני מאמרים שמעלים רעיונות וגם בודקים אותם (חלקם גם מבוססי בינה מלאכותית אז החלקים האלו פחות רלוונטיים) על כיצד לייעל אלגוריתם של מעלית.

את המאמרים מומלץ להוריד למחשב כPDF ולקרוא אחרת זה סיוט לבצע קריאה בתוך האתר

https://www.sciencedirect.com/science/article/pii/S2405896316302671

המאמר מציע פתרון המבוסס על פתרון לבעיה קודמת של אופטימיזציית מסלול של רכב בעזרת קוד גנטי (קוד השואב השראה מתהליך האבולוציה, להרחבה: אלגוריתם גנטי – ויקיפדיה (wikipedia.org))

https://www.researchgate.net/publication/220590321_Optimization_of_Group_Elevator_Scheduling_With_Advance_Information

המאמר מציע ייעול של אלגוריתם המעליות בעזרת קביעת לוח זמנים יומי המבוסס על תנועת המכוניות המגיעות לבניין.

מצגת שגם כן סוקרת את האפשרויות לייעל ולעבוד עם אלגוריתם מעלית (וגם דברים שלא קשורים)

http://co-at-work.zib.de/berlin2009/downloads/2009-10-01/2009-10-01-1100-BH-Online-Optimization.pdf

במצגת מוצג גם הרעיון של "הקצאה מעוכבת", בו לא מוקצית מעלית לכל קריאה באופן מידי וההקצאה מתרחשת כאשר מתברר מהי המעלית הרלוונטית ביותר.

השראה מהמרצה המיתולוגי להסתברות מהסרט 21 (<<<< פרופסור דן חפץ ללא צל של ספק כן?)

https://www.youtube.com/watch?v=iBdjqtR2iK4

עבור הרעיון של complicated case – וויתור על הבחירה שחשבנו כי היא הטובה ביותר בהווה עבור הגדלת הסיכויים שלנו לפתרון טוב יותר בצעד הבא!

אלגוריתם offline:

1. קבלת רשימה המכילה – זמן הזמנה, קומת מקור, קומת יעד.
2. הפעלת אלגוריתם חמדן: "בעיית בחירת פעילויות" על מנת ליצור רצפים ארוכים ככל הניתן של פעולות עבור כל מעלית (האלגוריתם בוחר בכל פעם את הפעולה עם זמן הסיום הקצר ביותר שיכולה להתקבל).
כל העצירות יישמרו כרצף (רשימה) של מספרי קומות בהם צריך לעצור ופרקי זמן של מנוחה בהם המעלית נשארת במקום.
3. עבור המשימות שנשארו: מנגנון "הצעת מחיר", מנגנון הבודק את המיקום הטוב ביותר בו כל מעלית יכול להכניס את הקריאה הנוספת.
הפתרון הטוב ביותר- פתרון בו זמן ההגעה + תוספת הזמן לקריאות הבאות (בכמה זמן נדחו שאר הקריאות של המעלית) הוא מינימלי.
<u>הערה:</u> בהצעת המחיר ניתן לחשב גם זמני מנוחה של המעלית, ניתן להוסיף למעלית פקודה של מנוחה בקומה מסוימת על מנת שתוכל לאסוף את הקריאה החדשה. זמן ההמתנה יתווסף לזמן העיכוב בחישוב הצעת המחיר.
4. הצעת המחיר חייבת לקיים את התנאים הבאים:
   a. הגעה לקומות המקור והיעד
   b. יישום כל משימות הביניים האפשריות (עצירה בדרך והעמסת לקוחות נוספים) בתנועה לאותו הכיוון!
   c. הגדרה – משימת ביניים אפשרי צריכה לעמוד בקריטריונים הבאים:
      i. קומת המקור נמצאת בתוך המסלול שבין קומת המקור של המשימה לקומת היעד של המשימה
      ii. כיוון התנועה של המשימה המקורית הוא זהה לכיוון משימת הביניים הפוטנציאלית
5. חישוב הצעת המחיר יתבצע כך: עבור כל מעלית, עבור כל עצירה של המעלית, נבדוק האם ניתן להכניס את עצירת הביניים על פי הקריטריונים בסעיף 4. אם כן- מחשבים את זמן ההגעה של המשימה החדשה + זמן העיכוב של שאר המשימות (כולל זמן המתנה במידה והמעלית צריכה להמתין על מנת לקבל את הקריאה).
כל מעלית מחזירה את חישוב הזמן המינימלי בתוספת אינדקסים בהם יש להכניס את העצירות החדשות במידה וקיימות.
המעלית בעלת ההצעה הטובה ביותר היא המעלית הזוכה והעצירות החדשות מתווספות לרשימת העצירות של המעלית.
6. העסקת המעלית בעלת הצעת המחיר הנמוכה ביותר (לפי הקריטריונים מעלה)
7. חזרה לסעיף 3 עבור הקריאה הבאה ברשימה, עד להקצאת מעלית עבור כל אחת מהקריאות.
8. הפעלת המעליות על פי סדר העצירות והמנוחות שנקבע מראש.

אלגוריתם online:

1. לכל מעלית מוגדרת רשימה המכילה את כל העצירות העתידיות שלה, בכל עצירה- ההופעה שלה בתחילת הרשימה תימחק והמעלית תעבור למטלה הבאה.
2. קבלת קריאה – input המכיל בתוכו זמן (זמן ההווה שבו התקבלה הקריאה) קומת מקור, קומת יעד, סוג המשימה (עליה/ירידה).
3. הקצאת מעלית למשימה שהתקבלה – לשם כך נקטלג כל מעלית כאחת מן האפשרויות הבאות:
   a. מעלית מועסקת, בתנועה בכיוון זהה לסוג המשימה הנדרש (עליה/ירידה) ומיקומה הנוכחי הוא מתחת לקומת המקור של הקריאה (במשימת עליה) / מעל לקומת המקור של הקריאה. (למשימת ירידה)
   b. מעלית במצב מנוחה.
   c. כל מעלית אחרת.
4. כעת מתוך כל המעליות העונות לאפשרות a או b נבצע פעולת "בדיקת הוספה"
   - הגדרת "בדיקת הוספה": אם ובמידה והמשימה תועסק על ידי המעלית המדוברת, באיזה מיקום תשובצנה קומות המקור והיעד של הקריאה בתוך מסלול המעלית הנוכחי?
     - קומות המקור והיעד תמיד ישובצו באותו רצף קומות בעלות כיוון משימה משותף (אך לא בהכרח כעוקבות) **או** בסוף המסלול כולו
   a. לאחר מכן, נבדוק בהתאם למסלול ההיפותטי, מהו **הזמן המשוערך לסיום המשימה** – הגעה לקומת היעד, כולל כל העצירות וזמני ההמתנה שבדרך.
   b. מבין כל המעליות מהקבוצות a-2 ו b- 2 תיבחר המעלית והזמן הטוב ביותר.
   c. במידה ויותר ממעלית אחת עומדת בדרישות הכניסה לסעיף, תיבדקנה האפשרות של complicated case ותיבחר המעלית האופטימלית בהתאם (הסבר בסוף האלגוריתם למקרה מיוחד זה)
   d. במידה ונמצאה לפחות מעלית אחת – מעבר לסעיף 5. אחרת, מעבר לסעיף 4.
5. במידה ולא מצאנו מעלית העומדת בדרישות סעיף 3, נבצע את תהליך בדיקת הוספה ולאחריו בדיקת הזמן המשוערך לסיום משימה לכל המעליות האחרות (קבוצת c-2)
6. הקצאת המשימה אל המעלית שנמצאה כאופטימלית למשימה והחזרת המיקומים האידיאליים להוספה בתוך הרשימה.
7. הוספת קומות המקור וקומת היעד אל המקום הרלוונטי ברשימת העצירות ועדכון המסלול הנוכחי אותו המעלית דרושה לבצע.
8. חזרה לסעיף 2.

נסמן X_f כמות הקומות בבניין.

נסמן X_e כמות המעליות בבניין ו AVG_{speed} מהירות ממוצעת של מעלית בבניין (סכום המהירויות חלקי כמות המעליות)

נגדיר $Epsilon_{time} = 1 - \frac{AVG_{speed}}{X_f} * 1.5$ סטיית זמן.

נגדיר $Delta_{floors} = X_f * 0.15$ סטיית קומות.

- Complicated case: מצב בו קיימות מספר מעליות העונות לאותה הקטגוריה וקיים חשש כי יש שתי מעליות בעלות זמני סיום משימה קרובים עד לכדי $Epsilon_{time}$ אך אחת מהן צריכה לבצע מסלול ארוך יותר משמעותית עד כדי $Delta_{floors}$ קומות.
- תפעול – בדיקה ביחס למעלית בעלת זמן ביצוע המשימה הטוב ביותר האם קיימת מעלית העונה לדרישות הבאות:
  - המרחק בין קומת המקור בקריאה לתחנה הקודמת לה במסלול המעלית הוא קטן יותר בלפחות $Delta_{floors}$ קומות מהמרחק בין קומת המקור בקריאה לתחנה הקודמת לה במסלול המעלית הנבדקת.

- זמן הביצוע של המעלית הנבדקת הוא קרוב ב **עד** $Epsilon_{time}$ לזמן של המעלית האופטימלית המקורית. (מדובר ב10 שניות גם לבניינים הגבוהים ביותר בעולם בתוספת מעליות אשר "טסות" במהירות ממוצעת של 150 קמ"ש – יותר מהיר משמעותית מהמעליות הבדיוניות שהיו בסימולטור)

קישור לקוד האלגוריתם myElevatorAlgo

https://github.com/amiramir96/OOP-Ex0/blob/main/ex0/src/ex0/algo/myElevatorAlgo.java

קישור לקוד מבנה הנתונים

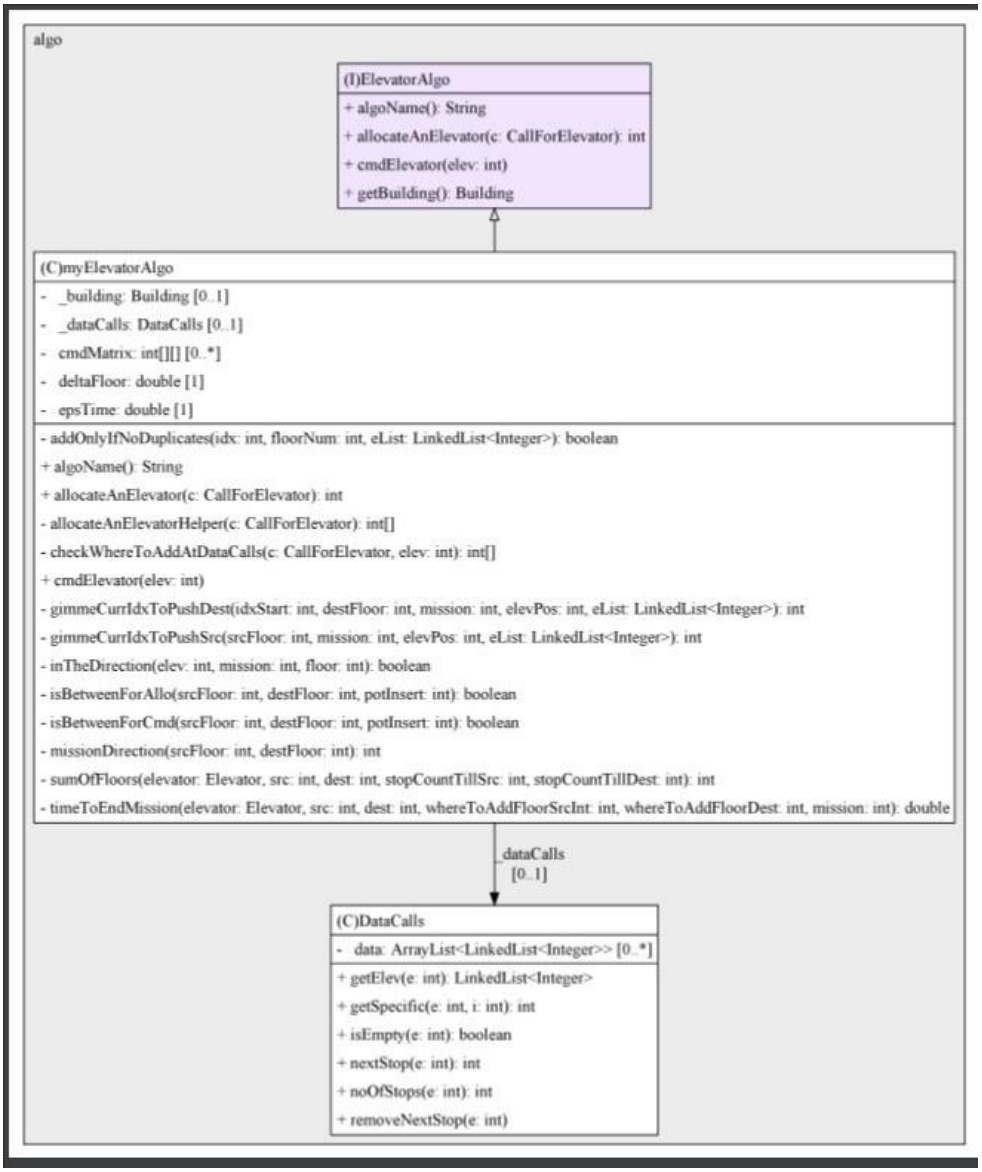
https://github.com/amiramir96/OOP-Ex0/blob/main/ex0/src/ex0/algo/DataCalls.java

דיאגרמת מימוש של אלגוריתם אונליין:

קישור לקוד הדיאגרמה בgithub – בכדי להציגו כתרשים, יש להורידו אל inteleji עם הפלאגין planuml

https://github.com/amiramir96/OOP-Ex0/blob/main/ex0/src/ex0/algo/ElevatorAlgo_structure.puml

בדיאגרמה מופיעים רק החלקים אותם התבקשנו לממש, לא מחלקות שמומשו מראש.



תכנון בדיקות לאלגוריתם ONLINE:

קישור לקבצי הבדיקות הבסיסיים על בסיס junit

https://github.com/amiramir96/OOP-Ex0/tree/main/ex0/src/tests

קישור לקבצי הבדיקות המעמיקים יותר, על בסיס תרחישי סימולציה ייעודיים אשר בנינו בכדי לדקור שרשראות של קריאות/תהליכים אשר יביאו למקרי קיצון

https://github.com/amiramir96/OOP-Ex0/tree/main/tests

1- בדיקת שפיות בסיסית, לראות כי האלגוריתם מצליח לבצע קריאה בודדת בהצלחה.
2- הקצאת קריאות בסדר הנכון כגון:
   a. כיוון המשימה וכיוון התנועה של המעלית או מצב מנוחה של המעלית מתועדף על מעלית הנעה כנגד כיוון המשימה
      i. שיטת בדיקה תהיה על ידי הזרקת תרחישים שונים בהם המעליות בתרחיש יצטרכו לנוע בהתאם למצופה וסדר הקריאות לא יתבלגן.
   b. הקצאת משימה למעלית בעלת הזמן המיטבי ביותר מתוך הקטגוריה אליה היא שוייכה
      i. שיטת בדיקה בדומה לi-a
   c. המערכת מנסה לבצע את הליך ה complicated case ואכן מתעדפת מעליות הנמצאות עונות בדרישות ההליך במידה וקיימות
      i. שיטת בדיקה – על ידי תזמון קריאות בצורה מדוייקת כך שמעלית כלשהי בהכרח תעמוד בדרישות c אך לא תהיה המיטבית (דורש חישוב חשבוני פשוט בהתאם למהירות המעליות וגובה הבניין)
3- שלב ההוספה למבנה הנתונים המאחסן את המסלול הייעודי לכל מעלית:
   a. הוספה בindex המתאים בתוך המסלול
      i. שיטת בדיקה – הזנת קריאות בשיטות שונות לדוגמה קריאה מ 1 ל6 ואז שניה לאחר מכן תיכנס קריאה מ2 ל4 אז נוודא כי אכן המסלול המעודכן יהיה 1 -> 2 -> 4 -> 6
   b. הוספת קומה לתוך מסלול קיים **אם ורק אם** היא איננה זהה לקומה שהייתה באותו המיקום (וכעת תזוז "ימינה") – כך לא יהיו כפילויות של קומות זהות צמודות בשרשרת המסלול
      i. שיטת בדיקה – בדומה לi-a-3 רק שהפעם ע"פ הדוגמה: קריאה מ1->6 וקריאה נוספת לאחר שניה מ 1->8 אזי המסלול המעודכן יהיה 1->6->8
4- שלב התנועה – אין צורך בבדיקה ייעודית מכיוון שזהו מנגנון שמבצע refresh כל tick ועל כן במידה וכל הבדיקות 1-3 יוצאות תקינות, אזי המעליות אכן נעות בכיוון ובתזמון הנדרש.