

תומר ינאי 305187296

יוגב מטלון 201390408

נעם תור 203302021

# איחוד וסדרור הזמנות בקו ליקוט

**תכן מפעלים**

**מרצה:** פרופ'יוסי בוקצ'ין

**עוזרת הוראה:** גב' יעל מילון

# תקציר

...

# הקדמה

פרויקט זה עוסק בסביבה של ליקוט פריטים בקו יחיד ע"י K עובדים, אשר עבודתם מתבצעת תחת משטר bucket brigade. תחת משטר זה סדר המלקטים בקו נקבע מראש, ונשאר קבוע לכל אורך הליקוט כאשר הליכה קדימה נמשכת עד אשר העובד אסף את כל הפריטים שהיה צריך לאסוף בארגז מסוים, או עד אשר נפגש בעובד שלפניו ההולך אחורנית והם ביצעו תהליך hand-off (העברת ארגז). הליכה אחורה נמשכת עד אשר ניתקל בעובד הקודם לנו, ניקח ממנו ארז ונתחיל תהליך קדימה, וכאשר מדובר בעובד הראשון ההליכה אחורנית תימשך עד לתחילת הקו להתחלת ליקוט ארגז חדש.

בפרויקט זה תהליך הליקוט מתאפיין בכך שכל ארגז נקרא batch, כאשר בכל batch יש n הזמנות המכילות פריטים שונים. בתהליך זה נקבל שתי הגבלות עיקריות: הזמנה שייכת לbatch אחד ויחיד, ובכל batch יש הגבלה על כמות הפריטים שניתן ללקט.

אם כן- בהינתן מטריצת הזמנות (F תאי ליקוט ו-N הזמנות), K עובדים (לכל עובד זמני הליכה סטוכסטים- קדימה/אחורה/ליקוט) ומגבלת כמות פריטים בארגז C, פיתחנו מודל לאיחוד הזמנות בארגזים (batches) וקביעת סדר הליקוט כל זאת ע"מ להקטין את משך הליקוט הכולל.

את המודל נבדוק על מספר קלטים המכילים מספר עובדים שונה, קצב עובדים שונה, כמות הזמנות שונה, סוג פריטים וקיבולת ארגז שונה.

# הצגת המודל

## הנחות המודל

* משך החלפה בין עובדים זניח
* חילוף בין עובדים מתרחש בתאים עצמם ולא ביניהם
* במידה ומלקט קדמי פוגש מלקט אחורי בעת איסוף, הראשון ימתין עד השני יסיים
* נפח פריטים זהה- רק הכמות מהווה מגבלת קיבולת
* רק העובד הראשון מתחיל ארגז חדש (אין עקיפות אחורה)

## המודל

מטרת פרויקט זה הינו למזער את משך הליקוט הכולל, ולכן פונקציית המטרה הינה:

- כמות מפריט i בהזמנה j מארגז b

**s.t:**

בעיה זו נחשבת לבעיה קשה מאוד עקב המשטר הנדרש בליקוט ההזמנות (bucket brigade), והן בגלל קצבים סטוכסטים של העובדים.

כדי לבצע אופטימיזציה לפונקציית המטרה נציג אלגוריתם מקורב הפותר בעיה זו באמצעות אלגוריתם גנטי הנעזר בתכנות לינארי ואלגוריתם חמדן בשלבי זריעת האוכלוסייה. את תוצאת הפתרונות נקרב בעזרת תכנית סימולציה שנכתבה בחבילת simpy (ספריית סימולציה של אירועים דיסקרטיים).

## הגדרת האוכלוסייה בבעיה

האוכלוסייה תהיה מורכבת מאוסף פריטים המייצגים את איחוד וסדרור ההזמנות. כל פרט יהיה בנוי כמערך חד מימדי, באורך מספר ההזמנות ((, כאשר כל תא במערך יכיל את מספר הארגז אליו שייכת ההזמנה .

**לדוגמה**, עבור בעיה המכילה 10 הזמנות, פתרון אפשרי ייוצג על ידי: 1 3 4 2 4 4 4 2 3 5. כלומר, משימה 1 משובצת לארגז החמישי בסדר, משימה 2 לארגז השלישי בסדר, משימה 3 לארגז השני וכן הלאה...

נציין כי אין הכרח שמספרי הארגזים יופיעו באופן עוקב. לדוגמה, גם הפתרון הנ"ל קביל: 1 3 4 6 4 4 4 6 3 5 (ארגז מספר 2 לא מופיע כלל, והדבר פשוט אומר כי סדר הארגזים הינו- 1,3,4,5,6, ובסימולציה "נתעלם" מארגז 2).

## זיווג ושחלוף

## מוטציות ואליטיזם

## זריעה

בשלב הראשון, עלינו ליצור אוכלוסיית פתרונות התחלתית ("דור ראשון"), ממנה נתחיל את תהליך החיפוש אחר הקירוב לפתרון האופטימלי לבעיה. אנו שואפים למצוא הורים (או הורה יחיד) בדור הראשון אשר יהווה "נקודת פתיחה טובה" לתהליך החיפוש, היות וזהו גורם המשפיע מאוד על קצב התכנסות האלגוריתם (וזוהי למעשה ה"זריעה" – Seeding).

במטרה למצוא פתרון התחלתי טוב, נבצע ניתוח איכותי לגורמים בבעיה המשפיעים במיוחד על זמן העבודה בקו:

* **זמני הליכה:** זמני הליכה יכולים להוות גורם משמעותי המשפיע על הזמן הסופי של הפתרון. זמני ההליכה נקבעים בעיקר לפי **כמות הארגזים** שבפתרון הנתון. לכן, בפתרון ההתחלתי שלנו נרצה למזער את כמות הארגזים (בהתחשב במסגרת האילוצים כמובן).
* **Blocking:** גורם משמעותי נוסף המשפיע על משך זמן הפתרון, הינו מצבי Block (אירוע בו עובד נחסם על ידי העובד שלפניו). על כן, אנו רוצים כי כל עובד יוכל להתקדם בקצב מהיר יותר מהעובד שמאחוריו. מצאנו שלושה דרכים מרכזיות בהן ניתן לנסות ולמנוע מצבי Block:

1. **מהירות העובדים** - היינו שואפים כי העובדים המוגדרים כמהירים יהיו לפנים, ובכך נקטין את סיכויי ה-Block (נתעלם בשלב זה מהסטוכסטיות של הבעיה).
2. **כמות יורדת של פריטים מכל סוג לאורך הקו** – הדבר מאפשר לעובד לנוע מהר יותר ככל שהוא בנקודות מתקדמת יותר בקו. כלומר, קיימת סבירות גבוהה יותר כי עובדים קדמיים יוכלו להתקדם מהר יותר מעובדים אחוריים.
3. שונות פנימית בארגז
4. סידור ארגזים מקטן לגדול – באופן זה

## תכנית הסימולציה

הסימולציה פותחה בעזרת חבילת simpy, כאשר תהליך הליקוט נבנה כמערכת שירות של "שרתים" ו"לקוחות". נותני השירות (שרתים) הינם תאי הליקוט (=הפריטים), והלקוחות הם העובדים עצמם כאשר משך השירות הינו זמן הליקוט של העובד.

לצורך התכנית נבנו ארבעה אובייקטים:

1. מפעל- המפעל מכיל את סדר ה-batches ותכולת איסוף דרושה לכל batch. בנוסף, המפעל מכיל רשימת העובדים שלו (מסוג אובייקט עובד) ואת רשימת המשאבים שלו (מסוג אובייקט משאב של simpy). כמות המשאבים היא כמספר התאים.
2. ארגז- אובייקט זה מכיל את הפריטים והכמות הדרושים לליקוט, ותכונה של העובד אשר עם ארגז זה. הארגז מכיל פונקציה של הליכה קדימה כאשר ארגז תמיד מתקדם (אלא אם כן הוא עומד במקום עקב block) עד סיום batch, והעובדים מתחלפים בדרך.
3. עובד- אובייקט זה מכיל את תכונות קצב הליכה קדימה/אחורה/ליקוט, כיוון התקדמות (אחורה/קדימה), האם סיים את עבודתו לגמרי ועם איזה ארגז הוא נמצא (אם בכלל). לאובייקט זה פרוצדורה של הליכה אחורה.
4. משאב- אובייקט מובנה בחבילת הסימולציה. תחת אובייקט זה נשתמש במתודות של בקשת משאב ושחרור, וכך נגרום לכך שעובדים לא יוכלו להתקדם בלי שהעובד שלפניהם סיים להשתמש במשאב (בתא הליקוט).

בנוסף, נחזיק פרוצדורת חילוף ארגזים, אתחול ופונקציה מרכזית המניעה את התהליך. תהליך הסימולציה כולו נבנה באמצעות דיאגרמת המצבים הבאה:

TBD

## ניתוח תוצאות

תוצאות של האלגוריתם הגנטי להתכנסות, ולהראות שהפתרונות הראשוניים היו הטובים.

ע"מ שנוכל לנתח את התוצאות בצורה נכונה, נדרש להחליט על מספר האיטרציות לביצוע כל מופע בסימולציה. כדי לעשות זאת בחרנו להריץ כל מופע בבעיה מספר קטן של פעמים (5), ולדרוש גודל טעות מקסימלי 0.1 (חצי רווח סמך) ברמת מובהקות 0.95.

רווח הסמך של זמן הסימולציה הינו:

- שונות תחושב מ-5 האיטרציות הראשונות

- נדרוש טעות מקסימלית של 0.1

- 0.05 מרמת מובהקות נדרשת

**מכאן:**

על כן, הרצנו N פעמים עבור כל מופע וממוצע המופעים הקצר ביותר עבור כל בעיה מרוכז בטבלה הבאה:

להוסיף טבלה מסכמת- לכל בעיה שניתנה את הפירוט שלה ותוצאת הממוצע לבעיה

בנוסף, נראה Box-plot של המופעים הנבחרים:

מצורף קוד שעושה זאת. יש רק להדביק את התוצאות ולנתח