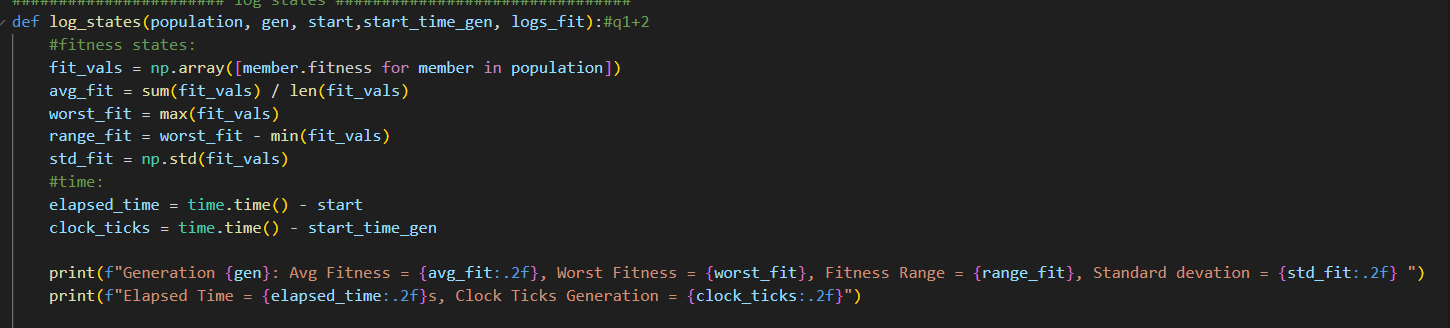
מעבדה בבינה מלאכותית

דו"ח תרגיל בית 1 - אלגוריתמים גנטיים (מערכות מתארגנות עצמית)

ת.ז :

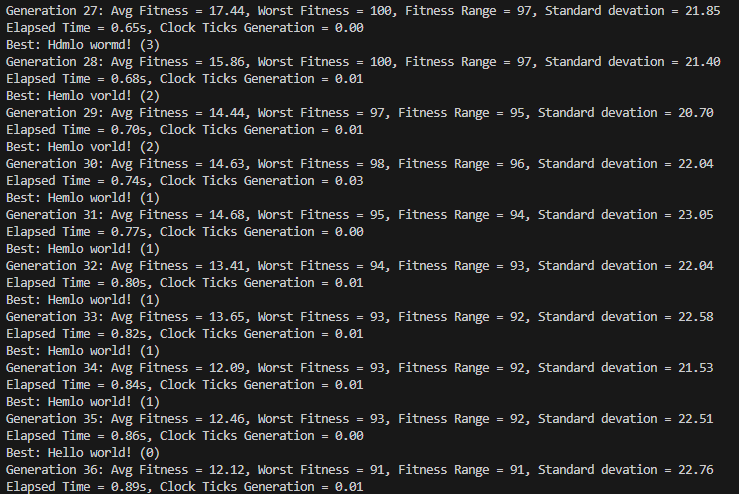
1. הפונקציה מחשבת ומדווחת את הממוצע של ה-FITNESS ,סטיית התקן, הפרט עם הפיטנס הגרוע ביותר ואת הטווח בין המצטיין לגרוע:

חישוב ממוצע: סוכמים את כל ערכי ה-FITNESS ב-population ומחלקים בגודל האוכלוסייה (שזה גם מספר הפריטים שיש להם פיטנס).

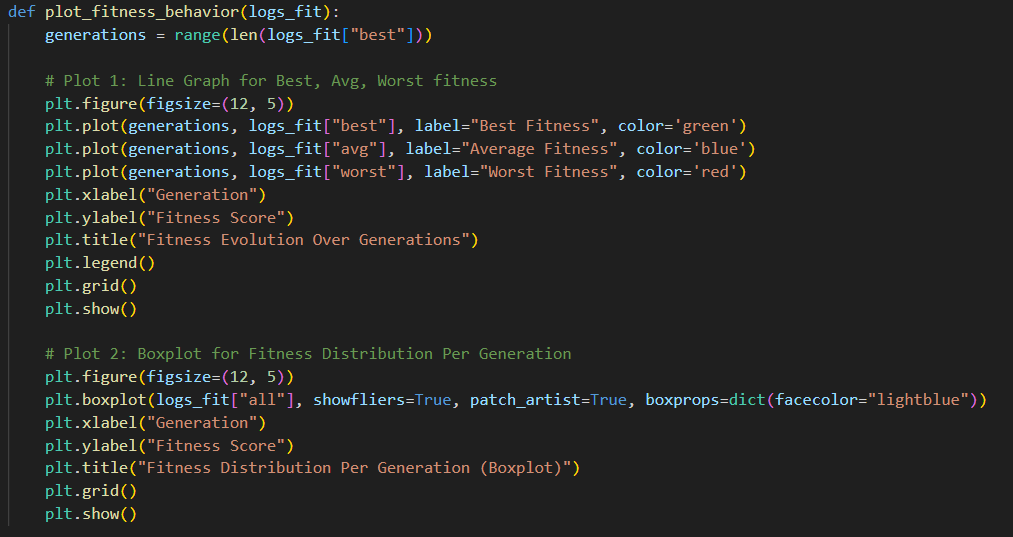
חישוב סטיית תקן : על ידי פונקציה של ספריית np.

חישוב הפריט הגרוע ביותר: הוא הפריט עם ערך הפיטנס הגבוה ביותר (כי לפי חישוב הפיטנס הוא הכי רחוק מלהיות דומה ל-GA\_TARGET ).

1. כפי שניתן לראות בצילום מסך בשאלה 1 , הפונקציה מקבלת את זמן ההתחלה של התוכנית וזמן ההתחלה של הדור ומחשבת את זמן הריצה בהתאם.







להוסיף תמונה של הגרפים

1. הגרף – השתנות הפיטנס מדור לדור (פיטנס של הפרט הטוב ביותר באוכלוסיה, הפרט הממוצע, הפרט הגרוע ביותר).

אנחנו רואים בעזרת הגרף איך האלגוריתם מתקרב לפתרון על ידי כך שאנחנו עוקבים אחרי הפיטנס בשלוש הדרכים הבאות:

קו ירוק (הכי טוב) - השאיפה שהקו ירד עם הזמן ויגיע ל-0 (הגענו לפתרון אידיאלי).

קו כחול (ממוצע) - הקו הכחול נותן אינדיקציה לרמת האוכלוסייה. כלומר, האם רוב הפרטים משתפרים או שרק חלק קטן משתפר (למשל, רואים שהקו הירוק יורד אבל הקו הכחול לא). נשאוף שהממוצע ישתפר עם הזמן.

הקו האדום (הפריט הגרוע ביותר) - מראה האם האוכלוסייה כולה משתפרת או רק חלקה , אם הקו עולה אז האוכלסייה כולה לא משתפרת אבל אם הקו יורד אז האוכלוסייה כולה משתפרת. אם הקו האדום לא משתפר אז יש בעיה בגיוון האוכלוסיה.

אם הפער בין הקווים מצטמצם עם הזמן אז ניתן להגיד שהאוכלוסיה מתכנסת סביב פתרון (זה יכול להיות טוב אם הפתרון טוב אבל זה גם יכול להיות פיתרון לא טוב).

באופן כללי הגרף מראה את מגמת השיפור של האוכלוסיה.

1. הגרף - השתנות הפיטנס מדור לדור בBOXPLOT

אורך התיבה (הטווח הבין־רבעוני) מייצג את הפיזור של ערכי הפיטנס עבור רוב הפרטים.

אם התיבה קטנה אז ניתן ללמוד מכך שהפיטנס של רוב הפרטים דומה כלומר יש התכנסות לפתרון כלשהו.

אם התיבה רחבה אז יש גיוון באוכלוסיה , כי יש פרטים עם רמות פיטנס שונות ולכן עדיין מתבצע חיפוש לפתרון.

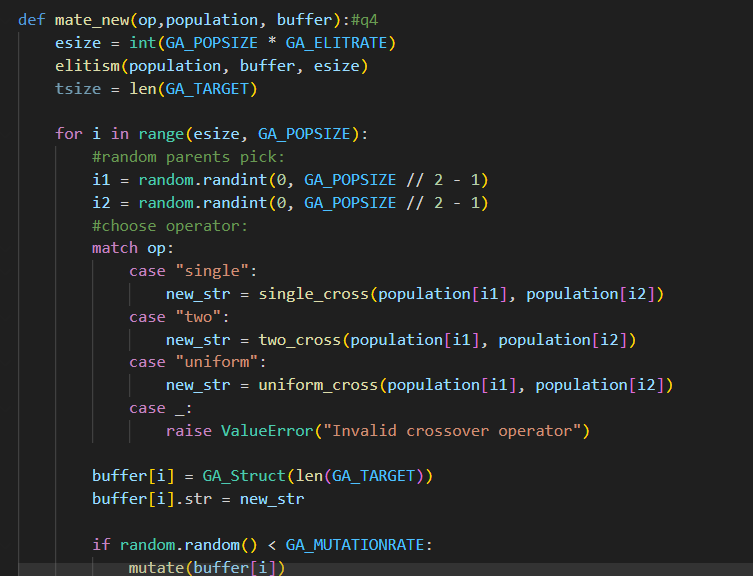
הקו שבתוך התיבה (החציון) מראה את הערך המרכזי של הפיטנס בכל דור.

הקווים הקיצוניים (whiskers) מציינים את תחום הפיטנס שאליו שייכים רוב הפרטים.

אם רואים שקיימים ערכים חריגים והערך שלהם טוב אז זה אומר שיש פרטים מוצלחים שצריך לשכפל.

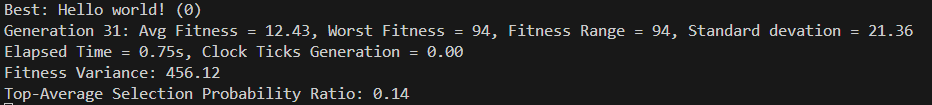
באופן כללי הגרף מדגים את הפיזור והגיוון של הפיטנס בכל דור.

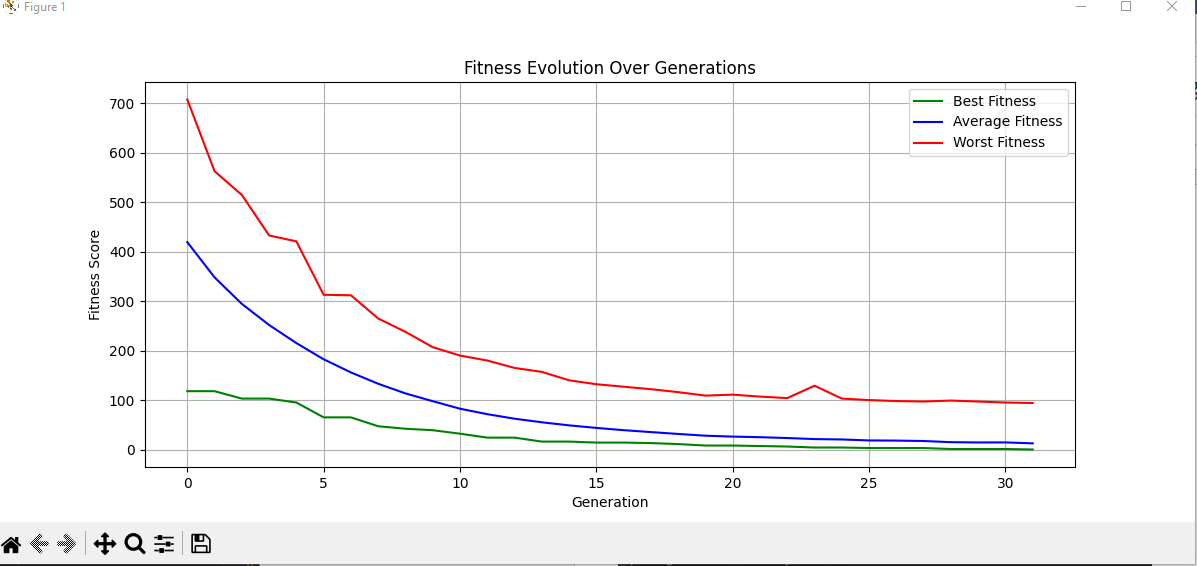




השוואה:

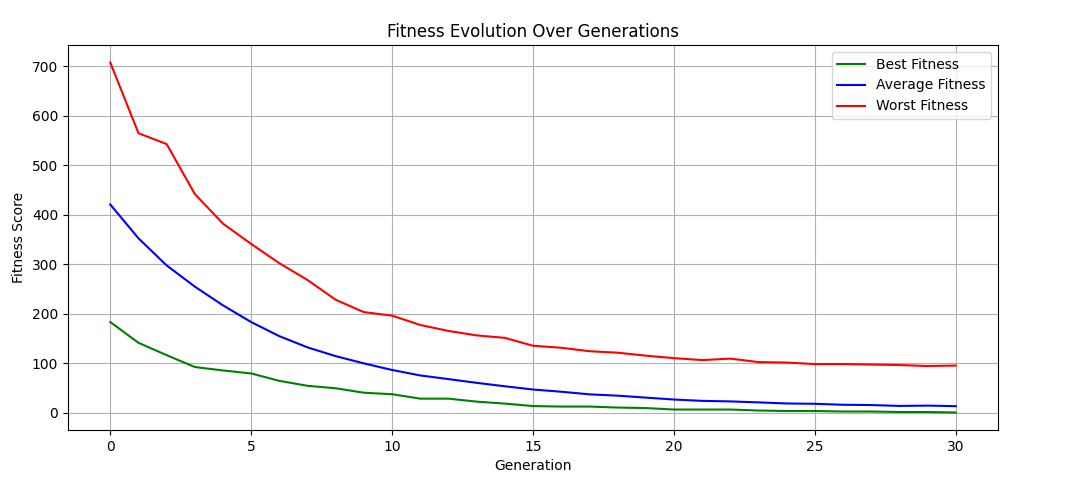
* Single:





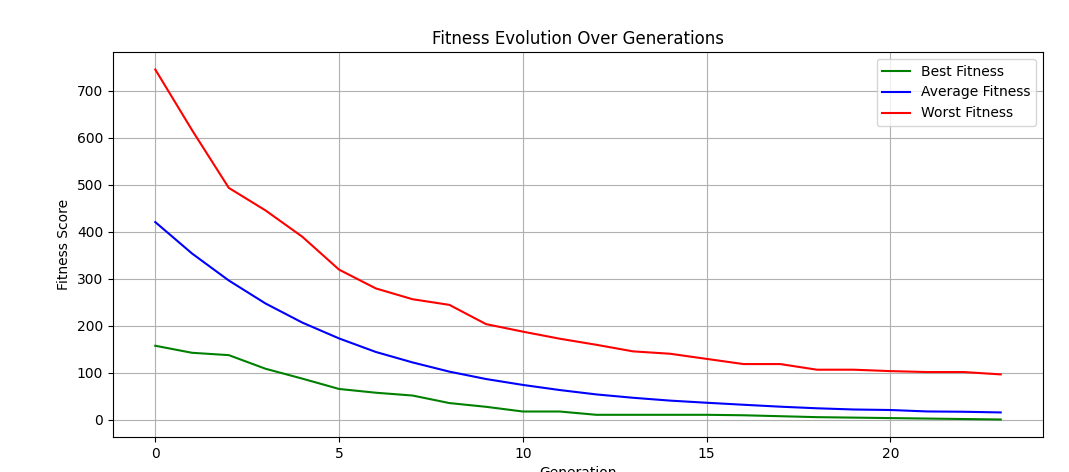
* Two:





* Uniform:







Exploration -

* מוטציה – שינוי אקראי (לפי if random.random() < GA\_MUTATIONRATE:) בפרטים. על ידי השינוי הזה אפשר לגלות פתרונות חדשים במרחב החיפוש.
* Init\_population – מאתחלים את האוכלוסייה עם מגוון פתרונות שנבחרים בצורה אקראית. לכן האוכלוסיה מגוונת וניתן לחקור אזורים שונים במרחב החיפוש.
* בסעיפים הבאים הוספנו SUS – דוגם את כל טווח הפיטנס בצורה יותר אחידה מ־RWS ולכן פחות נוטה לבחור את הפתרונות הטובים ביותר ומעודד יותר גיוון באוכלוסייה.
* Aging – נפטרים מפרטים כל עוד הם זקנים בלי קשר לאיכות הפתרון ומכניסים פרטים חדשים במקומם, הפרטים הצעירים מאותחלים עם פתרון אקראי ולכן אפשר לגלות פתרונות חדשים במרחב החיפוש.
* Crossover – יוצרים פתרונות חדשים על ידי שילוב בין שני פרטים.

Exploitation –

* Elitism – מתעדפים את הפרטים הטובים ביותר ולכן מתמקדים בשיפור פתרונות טובים.
* Selection – בסעיפים הבאים הוספנו טורניר שממנו לוקחים פרטים בצורה אקראית ובוחרים זוג הורים על ידי השוואה בין כל הפריטים לפי rank. תמיד יבחר הפרט הכי טוב.

גם וגם –

* ב- tournament\_probabilistic יש אלמנט של exploitation כי יש אפשרות לבחור פרט לפי ה-rank הכי טוב אבל גם יש אפשרות שיבחר פרט רנדומלי חלש וזה אלמנט של exploration.
* RWS - סיכוי הבחירה תלוי בפיטנס היחסי – הפרטים הטובים יותר נבחרים לעיתים קרובות, אבל גם לחלשים יש סיכוי קטן.

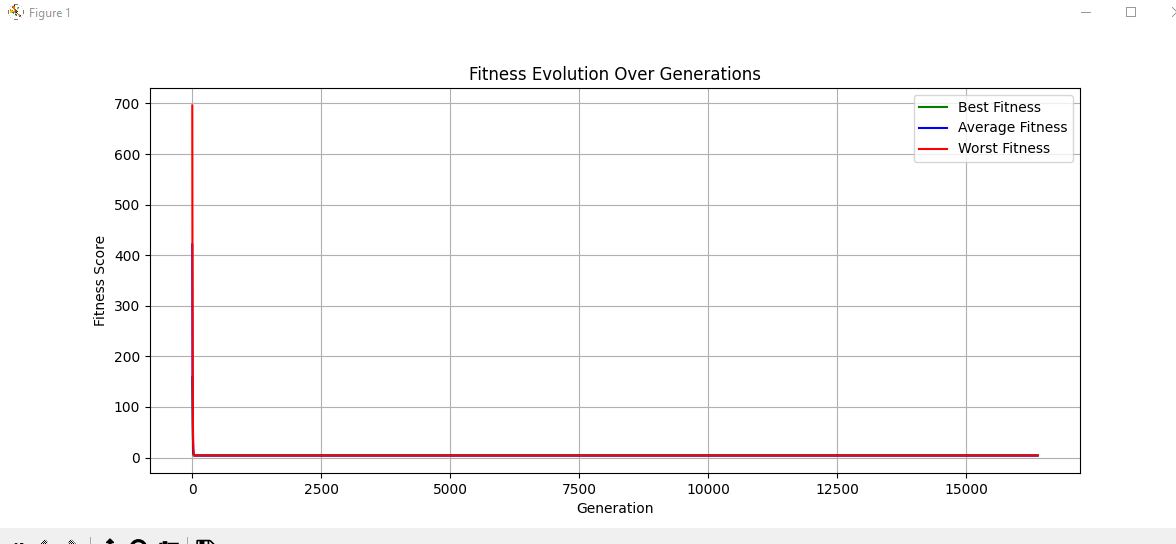


* רק עם שיחלוף ללא מוטציות:

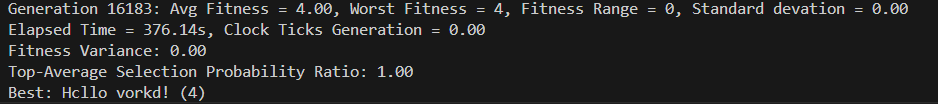
האלגוריתם לא מגיע לפתרון המיטבי, ובנוסף לכך האלגוריתם תקוע על פתרון מסוים כי אין exploration.

הממוצע , הפרט הגרוע ביותר , סטיית התקן , ו-fitness range לא משתנים ובפרט נראה שהאכלוסיה התכנסה לפתרון מסוים אך היא לא מצליחה להמשיך לאזורי חיפוש שונים על מנת להגיע לפתרון מיטבי.

ניתן לראות ש- fitness range = 0 וזה מצביע על כך שכל האוכלוסיה התכנסה לפתרון זהה שאינו הפתרון הממשי וכי אין גיוון באוכלוסייה.



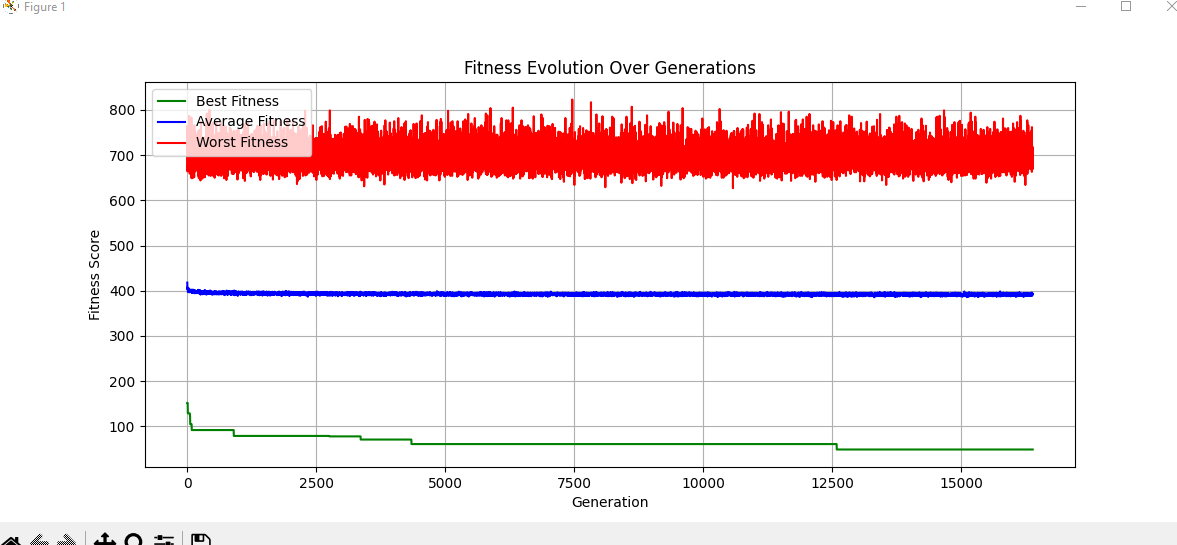
הפיטנס נשאר קבוע על הערך 4 לאורך דורות ללא שינוי.

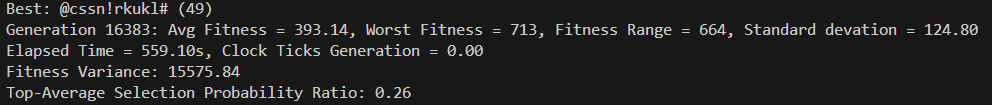


* רק עם מוטציות ללא שיחלוף:

האלגוריתם מתקדם באיטיות רבה - רמות הפיטנס גבוהות מאוד ולוקח הרבה זמן להתקדמות מבחינת הפרט הטוב ביותר, כלומר האלגוריתם מתקשה למצוא איזורי חיפוש חדשים ולכן הפרט הטוב ביותר לא משתנה במשך הרבה דורות.

האלגוריתם לא מגיע לפתרון המיטבי בזמן שניתן למקסימום דורות. אולי עם יותר דורות בסופו של דבר האלגוריתם יגיע לפתרון המיטבי אבל זה לא יעיל.

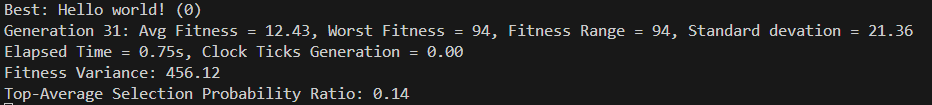


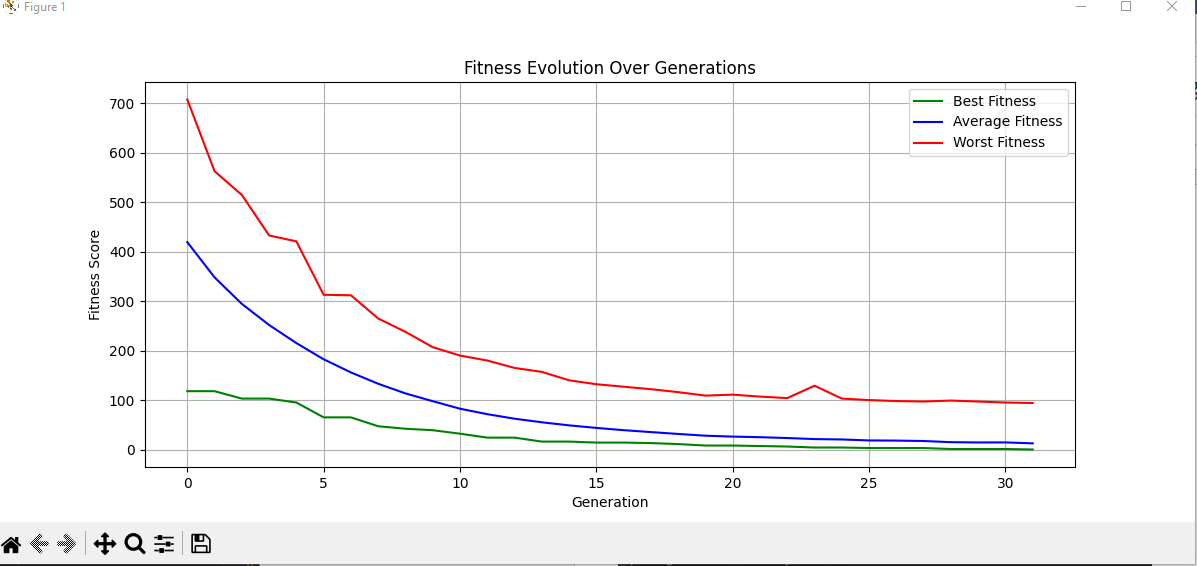


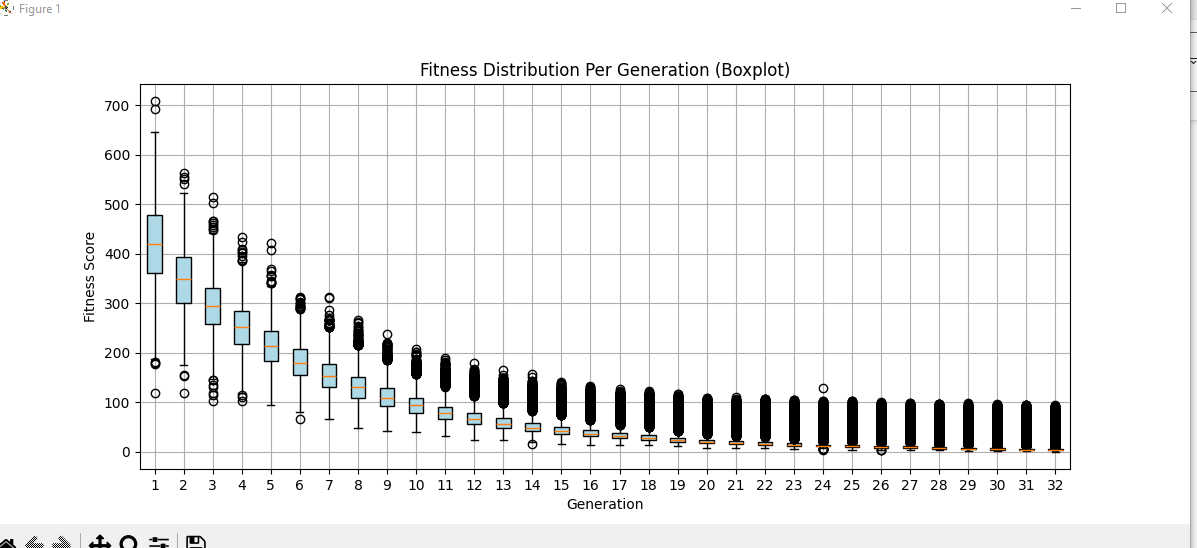
מהגרף ניתן לראות שהקו הכחול (הממוצע) נשאר כמעט קבוע וזה מעיד על חוסר גיוון באוכלוסיה, הפרטים הגרועים משתנים באותו טווח ערכים אבל לא מראים ירידה משמעותית בפיטנס. והקו הירוק (הפרט הכי טוב) מראה ירידה איטית מאוד שבה הרבה דורות נשארים באותו טווח.

מסקנה : מוטציה בלבד יוצרת חיפוש רנדומלי יתר על המידה ולכן קשה להתכנס לפתרון בזמן יעיל.

* עם שניהם:





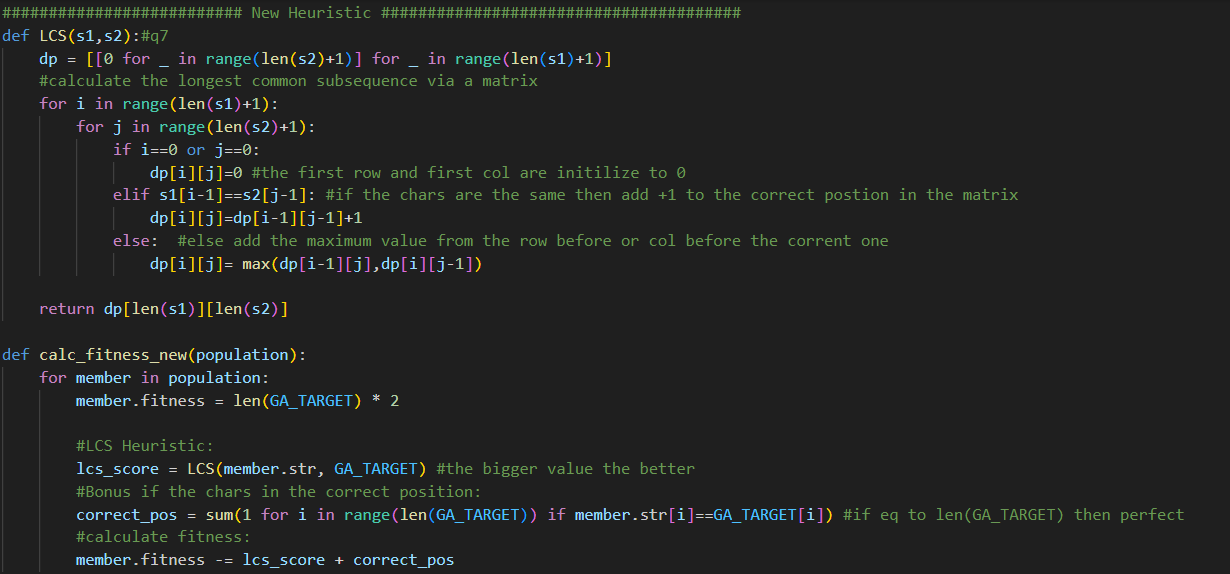


האלגוריתם הגיע לפתרון המיטבי תוך 31 דורות.

הגרפים מראים שהאוכלוסיה התכנסה לפתרון המיטבי בסוף הריצה, כפי שניתן לראות הקו הכחול(הממוצע) והקו הירוק (הפרט הטוב ביותר) מצמצמים פערים במהלך הריצה.

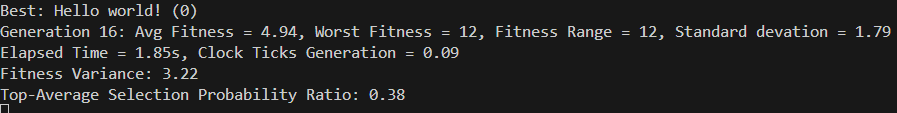
נראה שיש חשיבות למוטציה ולשחלוף על מנת לשמור על גיוון באוכלוסיה , לגלות איזורי חיפוש שונים ולאזן בין זה לבין שמירה על הפתרונות הטובים ביותר. כלומר יש איזון בין exploitation לבין exploration .

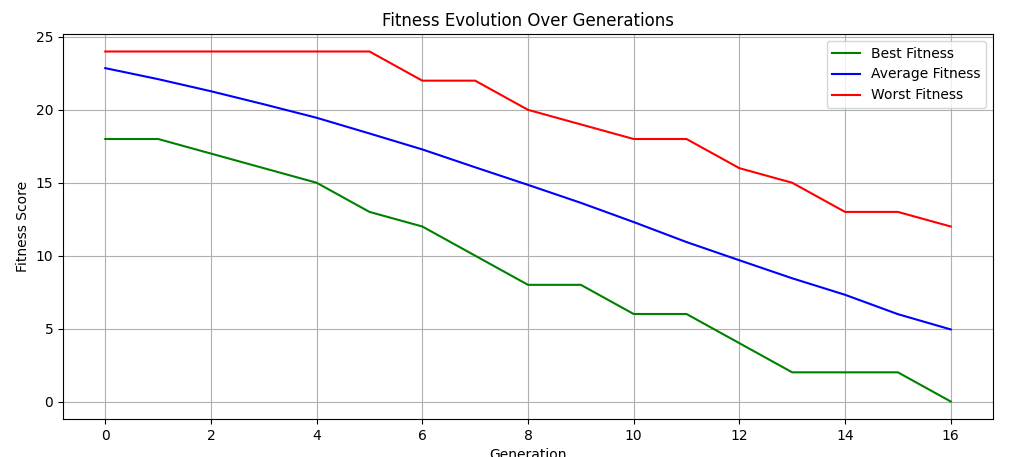


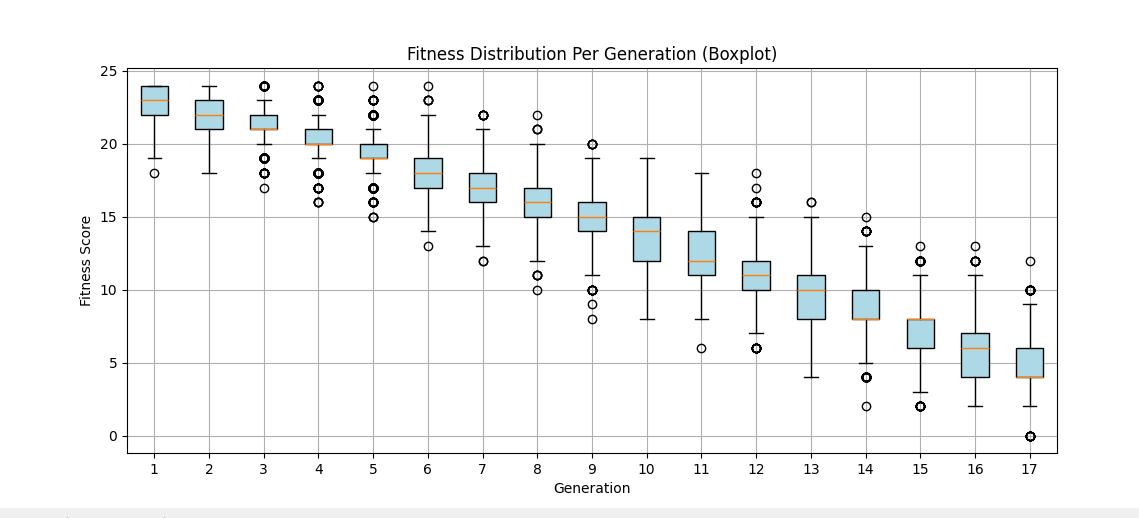


1. ההיורסיטיקה של LCS בודקת מהו התת רצף הארוך ביותר שתואם בין ה-string של הפרט לבין ה-string של המטרה, ככל שהפרט קרוב יותר לרצף המטרה בסדר האותיות כך ה-LCS SCORE גדול יותר. כאשר מבצעים שחלוף , אפשר לשמר תתי-רצפים של הורה אחד ואם אלו מובלים לתוצאות LCS SCORE גבוהות האלגוריתם מחשב פיטנס טוב יותר עבור הפרט שנוצר . כלומר, ההיוריסטיקה מעודדת את השחלופים ליצור פרטים עם סדר נכון של תווים – ולא רק אותיות נכונות במיקום מדויק .
2. נשתמש בהרצה האחרונה מסעיף 7 לצורכי השוואה. (הפרמטרים אצל שתי ההרצות זהות חוץ מהשימוש השונה בהיוריסטיקה) .

הריצה של LCS:







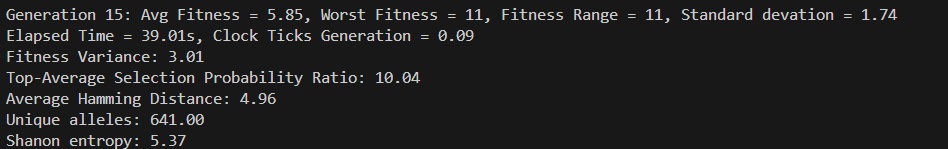
השוואה:

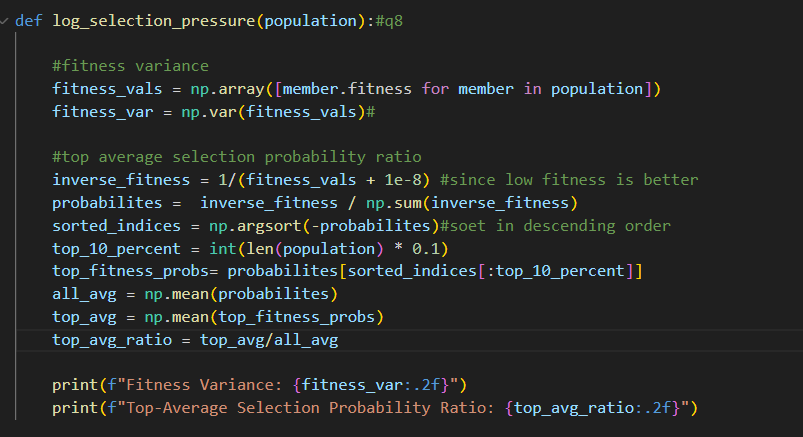
|  |  |  |
| --- | --- | --- |
|  | היורסיטיקה מקורית | LCS |
| מספר דורות | 31 | 16 |
| זמן ריצה | s0.75 | s1.85 |
|  |  |  |

במקרה של LCS האלגוריתם מגיע לפתרון המיטבי במחצית ממספר הדורות אך זמן הריצה ארוך יותר כי חישוב LCS ארוך יותר מהחישוב של פונקציית הפיטנס הקודמת.

איכות הפתרונות זהה בשני המקרים מאחר והאלגוריתם מגיע לפתרון המיטבי בשניהם. למרות זאת כן ניכר שבמקרה של LCS יש פער בין הפיטנס הממוצע לפיטנס הטוב ביותר, ולפי הגרף של BOXPLOT נראה שרוב האוכלוסיה לא התכנסה לפתרון המיטבי (אם כי ערך הפיטנס שלהם קרוב לכך) ויש פרטים חריגים שהערכים שלהם מעידים על כך שהם הגיעו לפתרון המיטבי.

לסיכום, הוספת ההיוריסטיקה של LCS תרמה לשיפור משמעותי במהירות ההתכנסות של האלגוריתם לפתרון מיטבי . ראוי לציין שבמקרים בהם נעדיף זמן ריצה יעיל יותר עדיף להשתמש בהיורסטיקה הקודמת. כמו כן, האוכלוסייה שומרת על גיוון רחב יותר – מה שיכול להועיל במקרים בהם הפתרון אינו ידוע מראש, אך פחות נדרש כאשר ידוע שיש פתרון אידיאלי ברור.

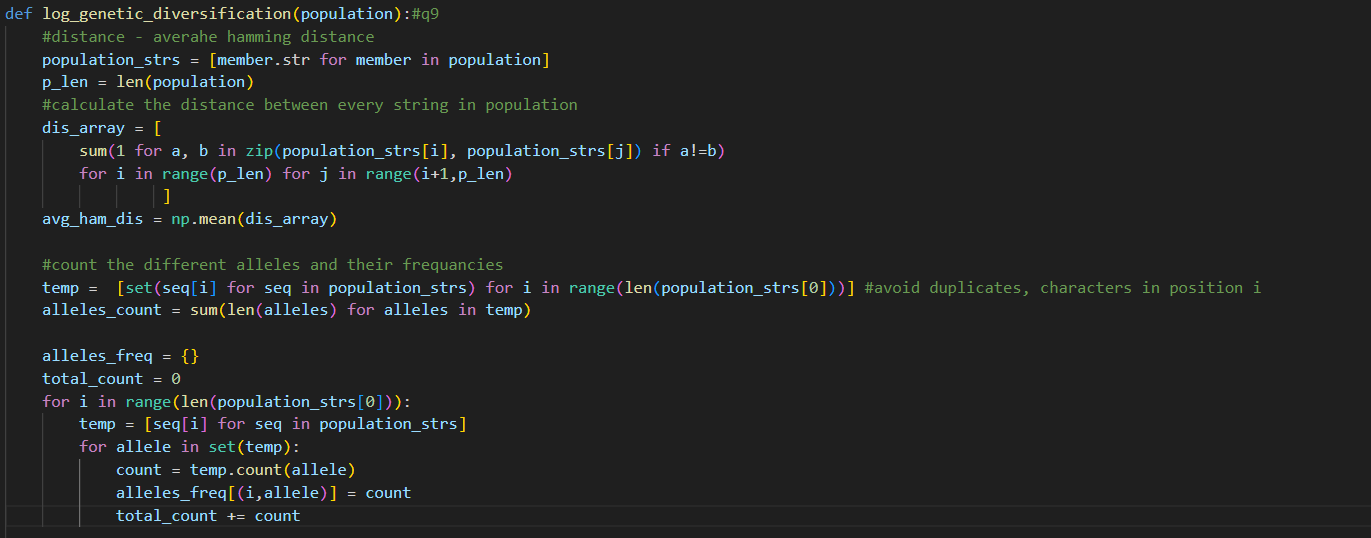
1. 

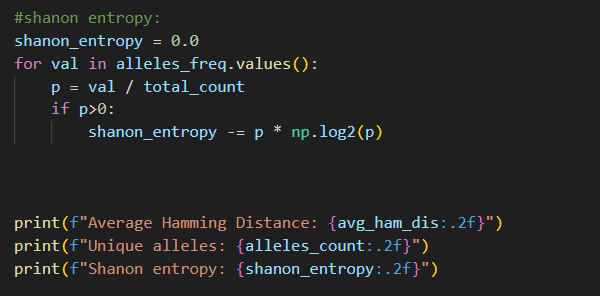


Fitness variance – שונות גבוהה מעידה שיש גיוון רחב בערכי הפיטנס באוכלוסייה ואילו שונות נמוכה מעידה שיש ערכי פיטנס דומים. כאשר השונות דועכת במהלך הדורות זה מעיד שקיים לחץ בחירה , אם יש ירידה מהירה בשונות אז לחץ הבחירה חזק. כאשר השונות נמוכה האוכלוסייה מתכנסת לפתרון ולחץ הבחירה חלש.

Top average selection probability ratio – לוקחים אחוז מסוים (במקרה הזה לקחנו 10%) של הטובים ביותר באוכלוסייה ומחשבים את הממוצע של סיכויי הבחירה שלהם ואז מחשבים את זה של כלל האוכלוסייה. יחס גבוה מעיד על כך שהמובחרים (10% הטובים ביותר) מקבלים עדיפות בבחירה ויחס נמוך מעדי שכל הפרטים מקבלים סיכויי בחירה דומים . ניתן לראות בדוגמא שבדור האחרון היחס עלה דרמטית (מקודם הוא היה 1.76) והסיבה לכך היא שהאלגוריתם גילה אזור חיפוש מבטיח והתכנס לשם (מעבר מ-exploration ל-exploitation).







* מרחק בין פרטים – average hamming distance – ערכים גבוהים מעידים על גיוון רחב באוכלוסייה בין הפרטים.
* מספר אללים שונים – ערכים גבוהים מעידים על חומר גנטי מגוון , כלומר הגיוון פה מעיד על האותיות ב-string מאשר על פתרון , כלומר ה-string עצמו .
* האנטרופיה של שנון – מודדים את התדירות והגיוון של ה-characters בכל פתרון (string) . ערך גבוה מעיד על גיוון גדול בחומר הגנטי.

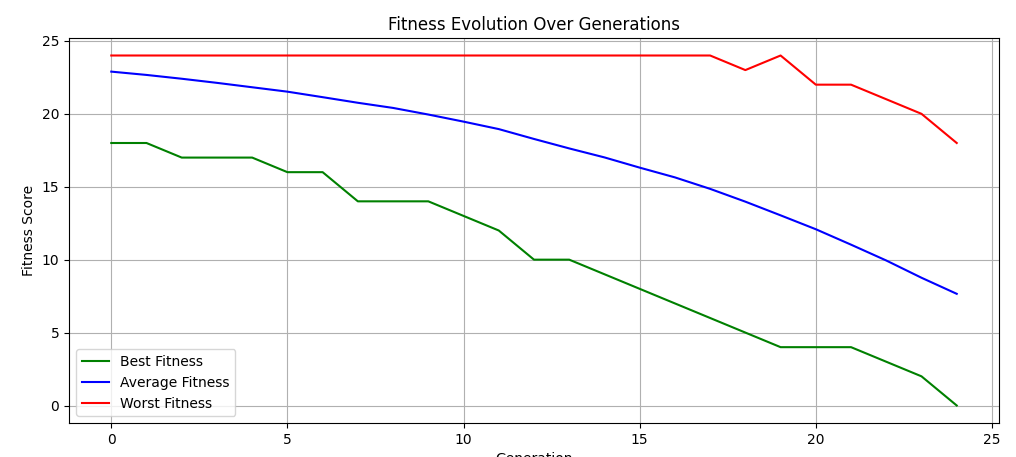
ההבדלים:

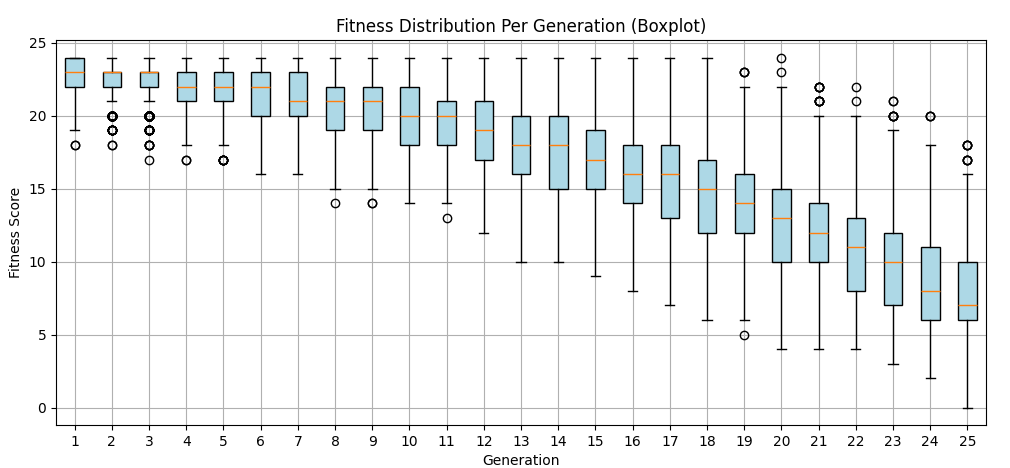
* המרחק בין הפרטים - מתמקד רק על הפתרונות השונים, כלומר ה-string המלא.
* מספר האללים השונים – הגיוון של characters בכל פוזיציה בפתרון (כלומר בכל אינדקס של ה-stirng) .
* האנטרופיה של שנון – מתייחס לגיוון ולתדירות.

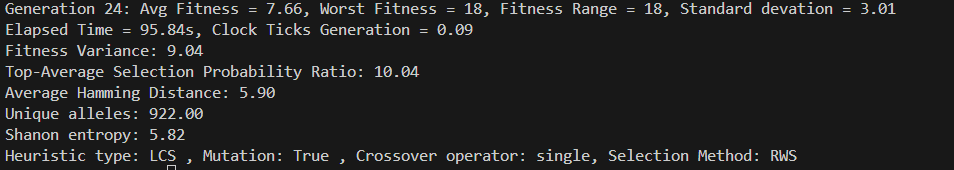
1. השתמשתי בהיורסטיקה LCS בכל הריצות כדי שהאלגוריתם יתכנס לפתרון המיטבי מהר יותר.

ההבדלים ב-RUN TIME בין הריצות בסעיפים הקודמים לריצות בסעיף הזה הוא השימוש בפונקציה log\_genetic\_diversification שלוקחת זמן חישוב ניכר.

* RWS +LINEAR SCALING:



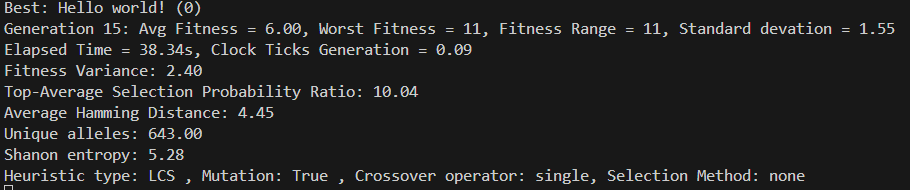




אלה התוצאות של הדור האחרון.

כפי שניתן לראות מהתוצאות לחץ הבחירה היה גבוה לקראת סוף הריצה (אבל זה לא יוצא דופן מריצות קודמות שלא השתמשו בשיטה לבחירת הורים).

האלגוריתם מסיים עם מספר רב של אללים שונים ולצורכי השוואה בריצה ללא בחירת הורים מספר האללים קטן ב-300 :

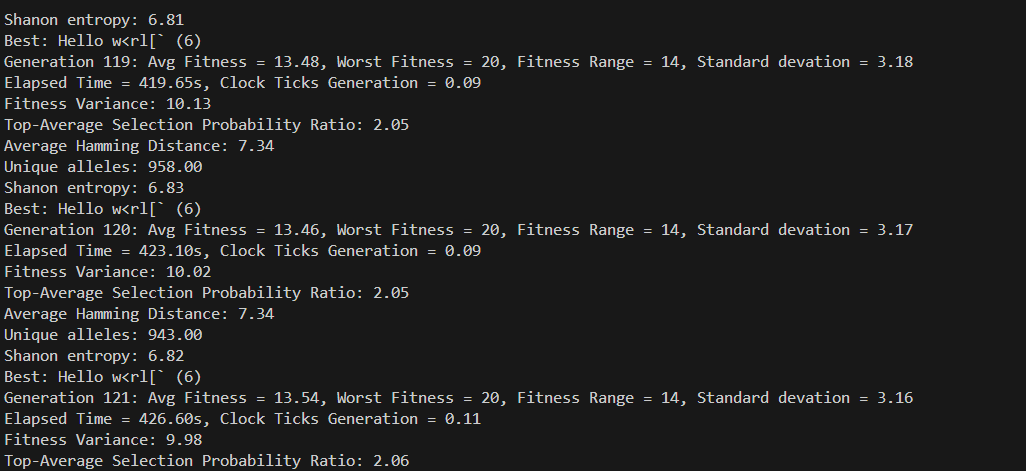


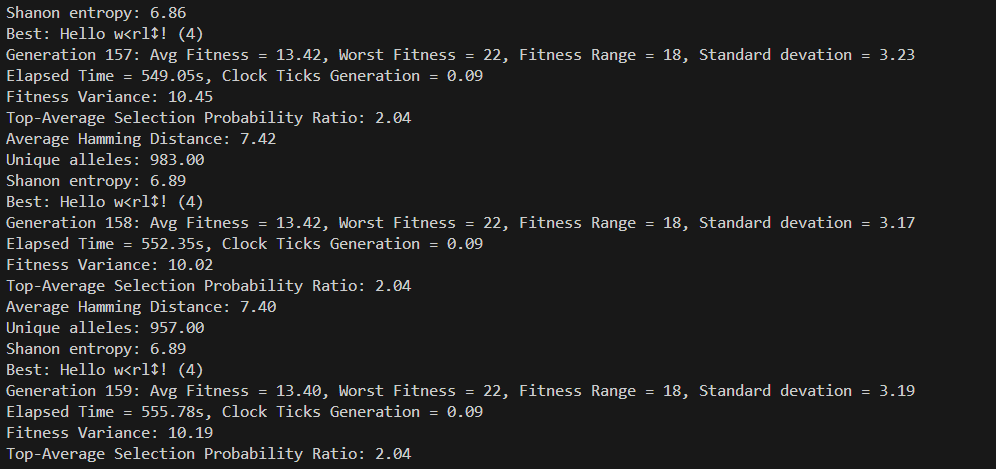
מבחינת שאר הריצות נראה מספר דומה ב-SUS אבל לא בטורנירים.

האלגוריתם מתכנס לפתרון המיטבי אחרי 24 ריצות ונמצא במקום ה-3 מתוך 4 .

ניתן לראות לפי הגרפים שיש גיוון באוכלוסייה ב-fitness ולא כולם התכנסו לפתרון המיטבי.

* SUS +LINEAR SCALING:





במשך דורות האלגוריתם תקוע באותו איזור חיפוש עד אשר הוא מגלה פתרונות חדשים כפי שניתן לראות בתמונות.

עצרנו את האלגוריתם בדור :

Best: Hello w<rl↕! (4)

Generation 233: Avg Fitness = 13.26, Worst Fitness = 22, Fitness Range = 18, Standard devation = 3.64

Elapsed Time = 809.21s, Clock Ticks Generation = 0.09

Fitness Variance: 13.25

Top-Average Selection Probability Ratio: 2.77

Average Hamming Distance: 7.38

Unique alleles: 950.00

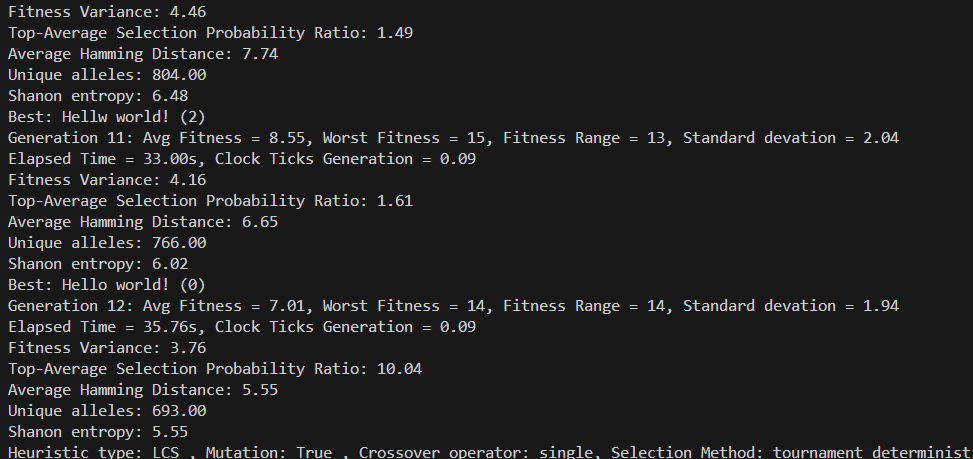
Shanon entropy: 6.95

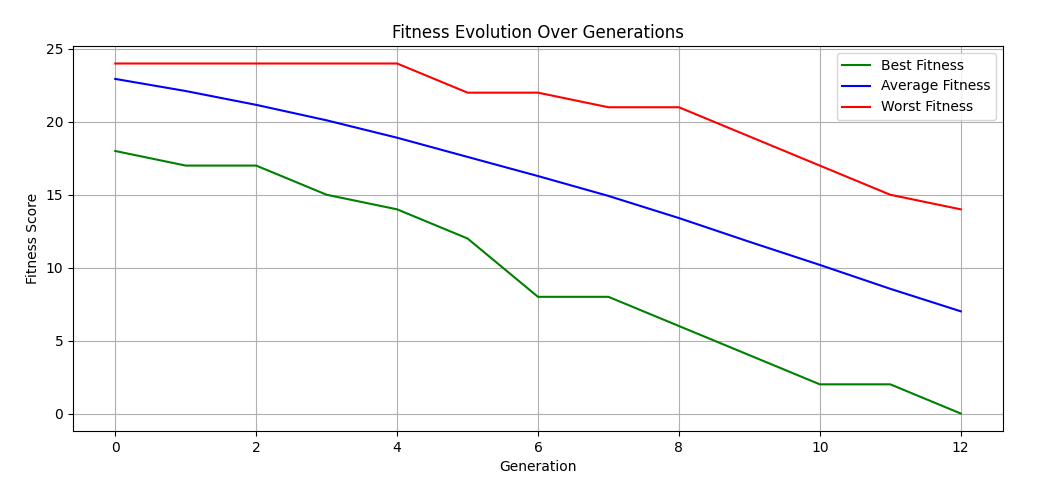
מאחר וזמן הריצה היה מאוד ארוך ונראה שיקח הרבה זמן לאלגוריתם להתכנס לפתרון המיטבי.

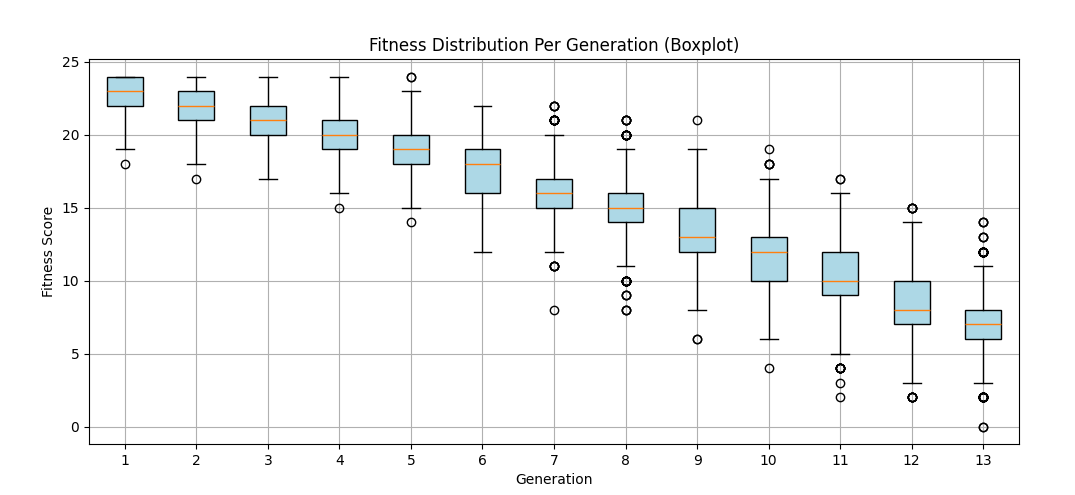
יש גיוון גנטי רחב , ולחץ הבחירה לפי top average selection probability ratio נמוך (כלומר הפרטים הטובים ביותר לא בעדיפות).

נראה שהאלגוריתם עסוק בחיפוש אחר כל מיני פתרונות ולא מצליח לנצל את הפתרונות הטובים ביותר ולפתח אותם עד להגעה לפתרון המיטבי.

* טורניר דטרמיניסטי ו- fitness ranking :

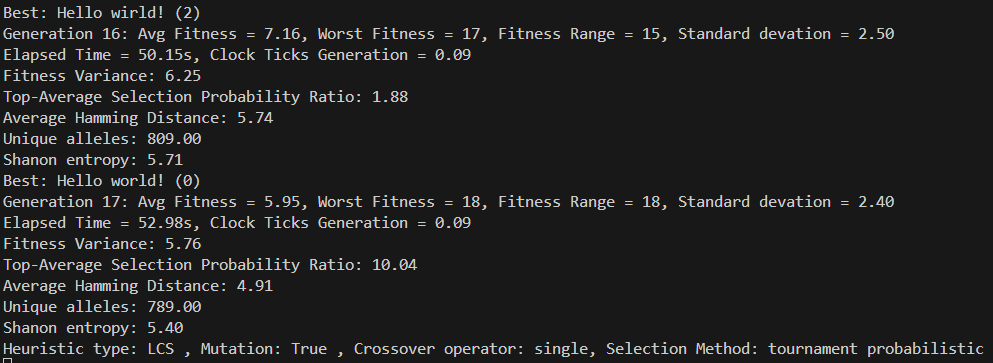


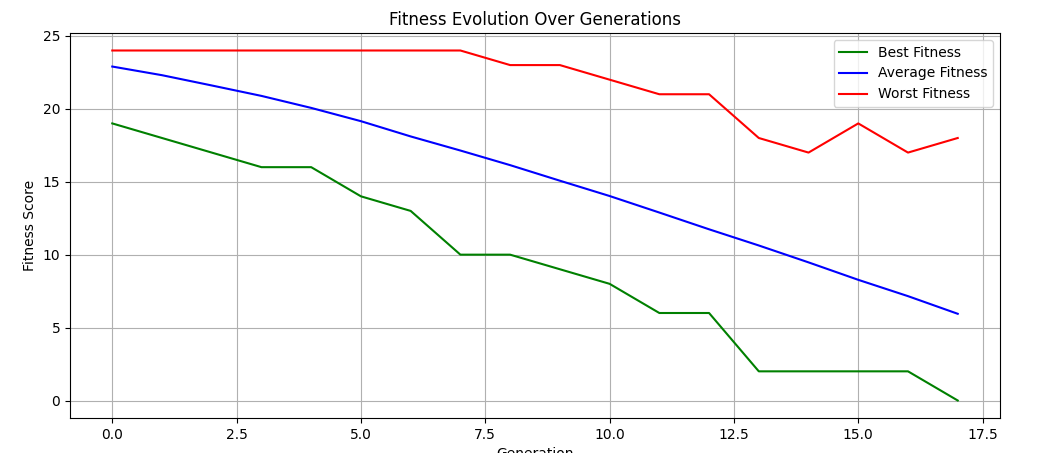


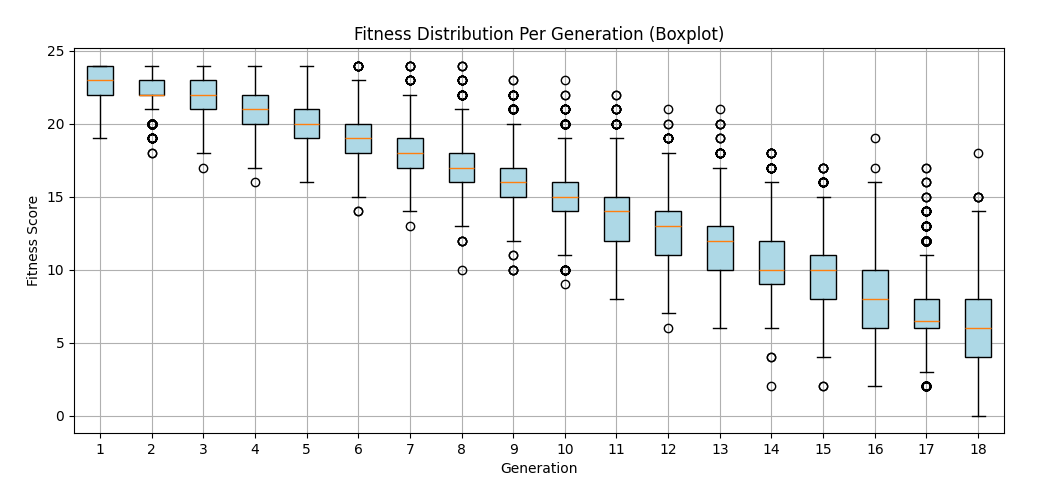


לחץ בחירה גבוה לפי top average selection probability ratio (כלומר הפרטים הטובים ביותר מועדפים) , התכנסות לפתרון המיטבי הכי מהירה מבחינת דורות וזמן ריצה. הגיוון הגנטי נמוך יותר ביחס לשאר השיטות ( יש מספר אללים נמוך יותר וניתן לראות בגרפים שהפיטנס של האוכלוסייה מתכנס לאותו פתרון , אם כי לא כולם מתכנסים לפתרון המיטבי).

* טורניר לא דטרמיניסטי ו-fitness ranking:



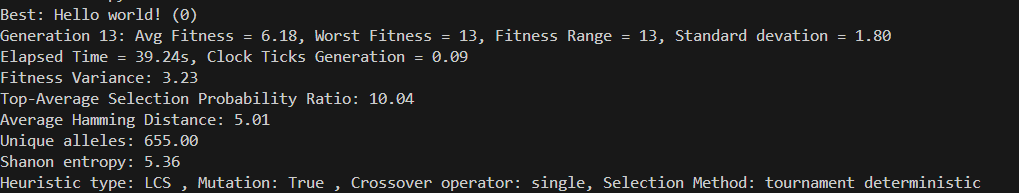




תוצאות דומות לטורניר דיטרמינסטי אך זמן ריצה ומספר דורות גדול יותר, ויותר גיוון גנטי (כפי שניתן לראות בגרפים ומספר האללים השונים).

* Aging

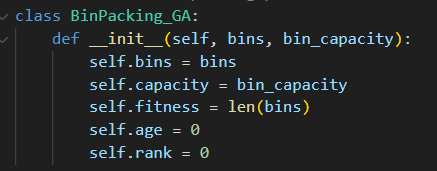
השתמשתי בשיטת טורניר דטרמיניסטי.



גיל מונע מפרטים להיות דומיננטיים ליותר מדי זמן ועל ידי כך לתת אפשרות לאלגוריתם לגלות פתרונות חדשים ולא להישאר תקוע באותו איזור חיפוש. הוא מוסיף גיוון גנטי כי נוצרים פרטים חדשים במקום הישנים שמוסיפים אפשרויות חדשות על ידי אתחול רנדומלי.

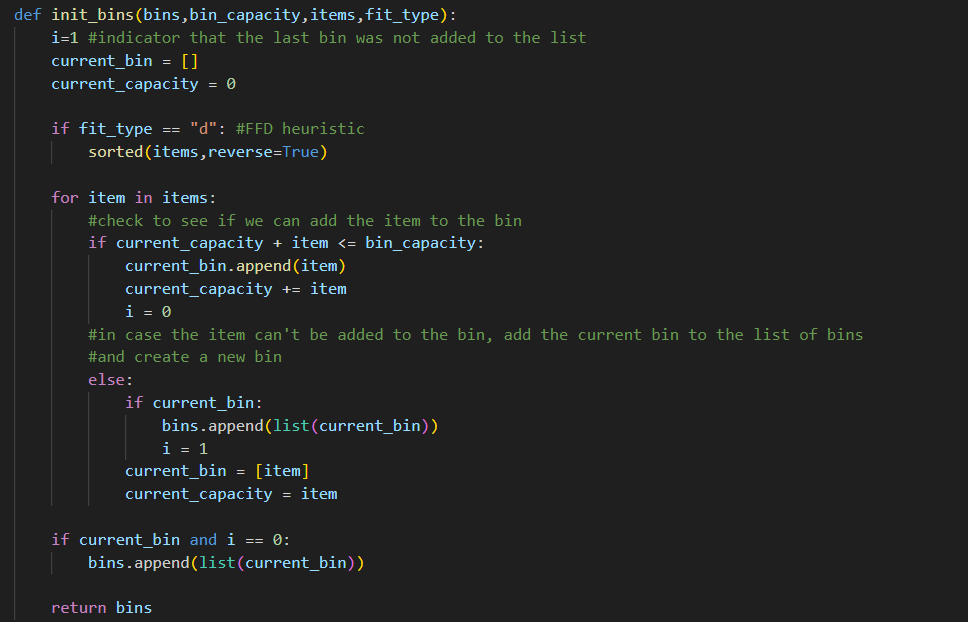


הייצוג של הבעיה :

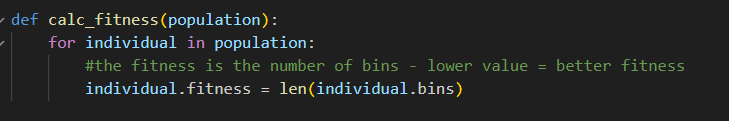


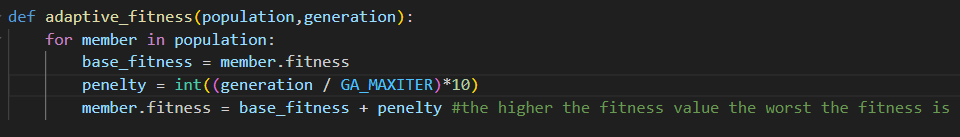
הכורומוזום הוא רשימה של רשימות ה-bins כאשר כל אחד מכיל את הגדלים השונים, כי המטרה היא לאחסן את המוצרים בכמה שפחות bins (כפי שבסעיפים הקודמים המטרה הייתה לצמצם את ההבדל בין הstring של הפרט לזה של המטרה) .

מאתחלים את ה-bins , אם נשתמש ב-FF אז נאתחל את המוצרים בצורה רנדומלית ונשתמש בהיוריסטיקה FF כדי למקם כל מוצר. אם נשתמש ב-FFD נסדר את המוצרים בסדר יורד ואז נמקם כל מוצר.



הפיטנס מחושב על ידי מספר ה-bins ,ובגלל שהמטרה היא שנאחסן את כל המוצרים בכמה שפחות bins ככל שהפיטנס נמוך כך הפתרון יותר טוב.

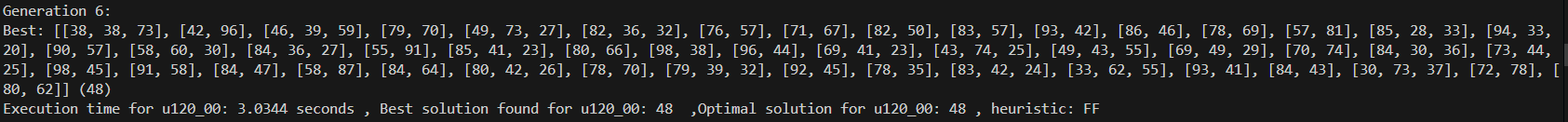




ב-adaptive fitness אנחנו רוצים לעודד מגוון של פתרונות ולכן נותנים penalty שגדל ככל שהאלגוריתם ממשיך מבלי להגיע לפתרון המיטבי (מה שנתון בשאלה).

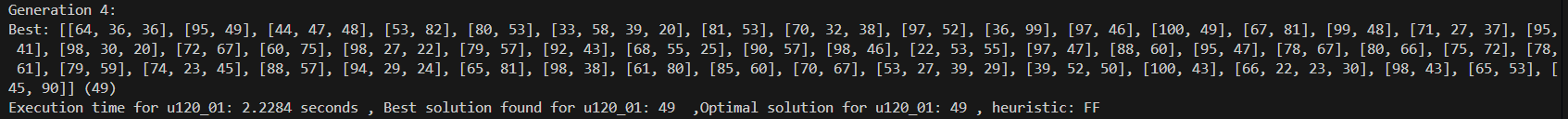
תוצאות עבור היוריסטיקה FF:

Gen = 6



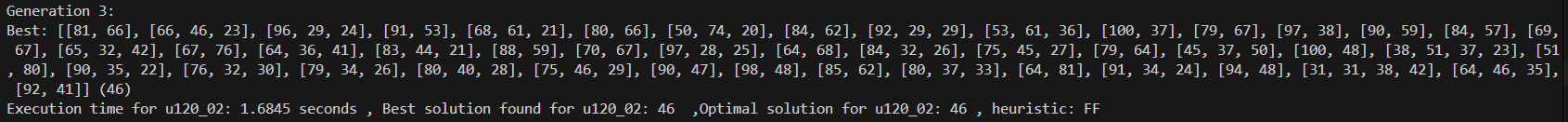
Execution time for u120\_00: 3.0344 seconds , Best solution found for u120\_00: 48 ,Optimal solution for u120\_00: 48 , heuristic: FF

Gen =4



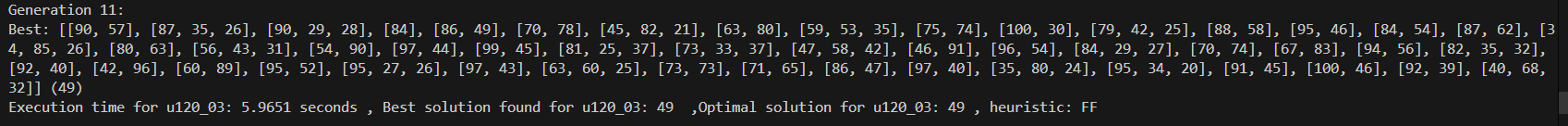
Execution time for u120\_01: 2.2284 seconds , Best solution found for u120\_01: 49 ,Optimal solution for u120\_01: 49 , heuristic: FF

Gen = 3



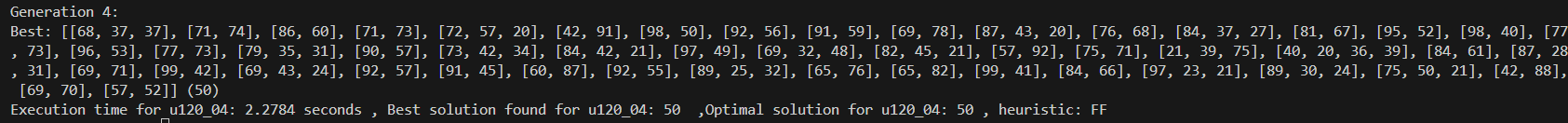
Execution time for u120\_02: 1.6845 seconds , Best solution found for u120\_02: 46 ,Optimal solution for u120\_02: 46 , heuristic: FF

Gen = 11



Execution time for u120\_03: 5.9651 seconds , Best solution found for u120\_03: 49 ,Optimal solution for u120\_03: 49 , heuristic: FF

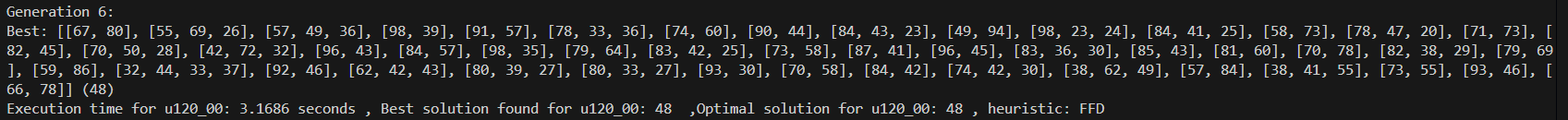
Gen = 4



Execution time for u120\_04: 2.2784 seconds , Best solution found for u120\_04: 50 ,Optimal solution for u120\_04: 50 , heuristic: FF

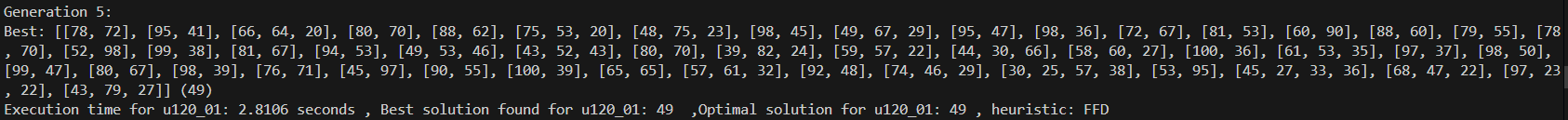
תוצאות עבור היוריסטיקה FFD:

Gen = 6



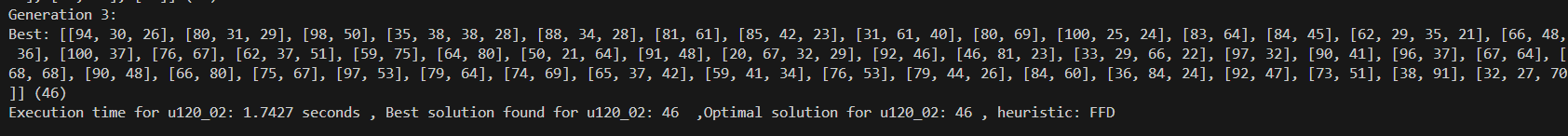
Execution time for u120\_00: 3.1686 seconds , Best solution found for u120\_00: 48 ,Optimal solution for u120\_00: 48 , heuristic: FFD

Gen = 5



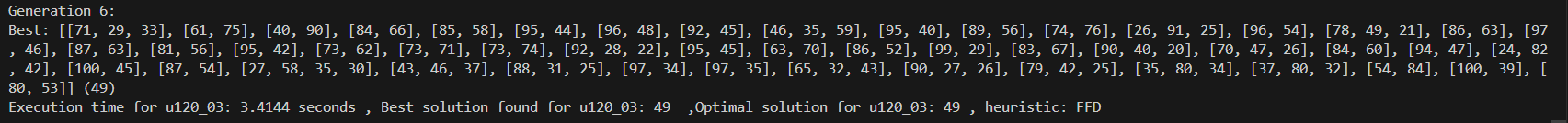
Execution time for u120\_01: 2.8106 seconds , Best solution found for u120\_01: 49 ,Optimal solution for u120\_01: 49 , heuristic: FFD

Gen = 3



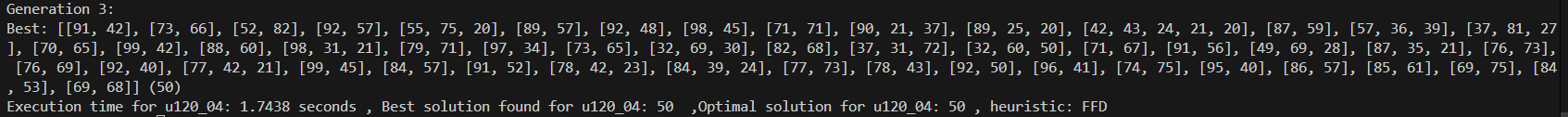
Execution time for u120\_02: 1.7427 seconds , Best solution found for u120\_02: 46 ,Optimal solution for u120\_02: 46 , heuristic: FFD

Gen = 6



Execution time for u120\_03: 3.4144 seconds , Best solution found for u120\_03: 49 ,Optimal solution for u120\_03: 49 , heuristic: FFD

Gen = 3



Execution time for u120\_04: 1.7438 seconds , Best solution found for u120\_04: 50 ,Optimal solution for u120\_04: 50 , heuristic: FFD

שתי הגישות הגיעו לפתרון האופטימלי בכל חמשת השאלות. עם זאת, FFD השיגה פתרונות במספר דורות נמוך יותר, כלומר מהירות התכנסות טובה יותר, ובחלק מהמקרים גם זמן ריצה קצר יותר, כנראה בגלל שמיון מוקדם של הפריטים מאפשר להקטין את כמות המכלים מהר יותר.