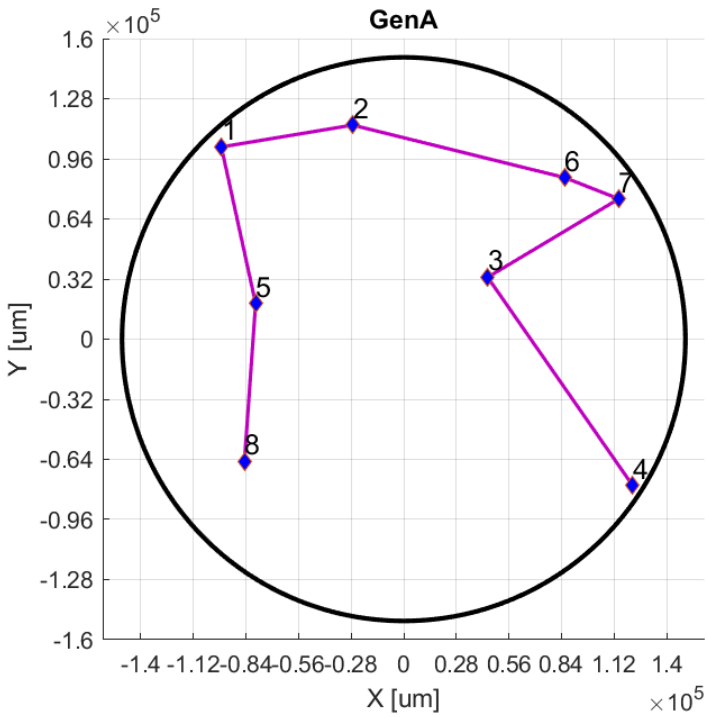
Multi-Objective Optimization

Traveling Salesperson Problem Project Report



מרצה – מר אלירן פרחי

מגישים:

שי גולדשטיין 060517232

אייל צימרמן 204092654

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

תוכן עניינים

1. מבוא

מערכת SEM (Defect Review) DR היא חלק אינטגרלי וקריטי במפעלי ייצור שבבים אלקטרונים.

המערכת משתלבת בפס הייצור בתהליכי הבקרה כ"עיניים" של המפעל. המערכת מקבלת מפת מיקומים חשודים לפגמים בייצור, מנווטת אליהם ומצלמת אותם במיקרוסקופ אלקטרוני.

המערכת נדרשת לטעות ניווט הקטנה מכמה מאות ננומטרים ולבצע פעולה זו בזמן הקצר ביותר.

2. הגדרת הבעיה

המערכת צריכה לנווט בזמן מינימלי ובדיוק מקסימלי לכל מיקומי הפגמים הפוטנציאלים על מנת לתת ערך מקסימלי ללקוח ולשפר את תהליך הייצור. על ידי השגת יעדים אלו, המערכת תוכל לדגום יותר פגמים בתמונות ברזולוציה משופרת וכך לשפר את תהליך הייצור.

ישנן דרכים רבות לשפר פרמטרים אלו, אך לעבודה זו התמקדנו באופטימיזציה של מסלול הניווט בין הפגמים. בצורה זו הבעיה שקולה לבעיית הסוכן הנוסע.

להגדרת הבעיה הנחנו את ההנחות הבאות:

- מיקומי הפגמים וכמותם משתנים בכל פעם ופעם ואינם ידועים מראש
  - לתוצאות המוצגות בדו"ח זה השתמשנו במיקומים קבועים של 50 פגמים על מנת לבצע ניתוח סטטיסטי והשוואה בין שני אלגוריתמים שונים
- כמות הפגמים יכולה לנוע מפגם בודד ועד 50 אלף פגמים
- נקודת ההתחלה אינה קבועה ויכולה להיות כל פגם מהרשימה
- אין צורך לחזור בסוף המסלול לנקודת ההתחלה ולכן המסלול אינו מעגלי
- הפגמים יכולים להיות בכל מקום בפרוסת הסיליקון הנבדקת ברדיוס של 150 מ"מ
- קורדינטות הפגמים נתונים ביחידות של ננומטרים
- ביצועי אלגוריתם הניווט הקיים במערכת, Nearest Neighbor (NN), הינם בסיס השוואתי לביצועי האלגוריתמים המוצגים בדו"ח זה

3. הגדרת בעיית האופטימיזציה

נבחרו שתי פונקציות מטרה שברצוננו למזער:

$$f_1(\overline{d}) = \sum d_i$$

$$f_2(\overline{d}) = \sqrt{\frac{\sum(d_i-\mu)^2}{N}}$$

כאשר:

$$\overline{d} = (d_1,\ d_2, \dots,\ d_N)$$

d_i - distance between defect i and defect $i+1$

N – number of defects

$$\mu = \frac{\Sigma d_i}{N}$$

כלומר, הפונקציה הראשונה היא סכום המרחקים והפונקציה השנייה היא מזעור סטיית התקן של המרחקים.

על ידי מזעור סכום המרחקים אנחנו מוודאים שהמרחק הכולל קצר ככל הניתן, ואנחנו יודעים שישנו קשר לינארית בין זמן התנועה למרחקו.

על ידי מזעור סטיית התקן של התנועות אנו מוודאים שהתנועות אחידות ככל הניתן אחת לשנייה במרחקן וכך מוודאים שהתנועה האופיינית תהיה קצרה ככל הניתן. אנו יודעים שתנועה קצרה בעלת קורלציה חזקה לטעות ניווט קטנה יותר ביחס לתנועה ארוכה.

מרחב ההחלטות מוגדר לפי קורדינטות הפגמים:

$x_i,\ y_i$ - defect coordinates

ומוגבל לפי:

$$\sqrt{x_i^2 + y_i^2} \leq 150{,}000,\ \forall i$$

4. האלגוריתמים

בחרנו בעבודה זו בשני אלגוריתמים גנטיים:

- NSGAII – נבחר לפי סקירת מאמרים לבעיית TSP עם שתי פונקציות מטרה. ראינו שזהו אלגוריתם פופולרי לבעיה זו בספרות ויחסית מוצלח. כמו כן, אלגוריתם זה נלמד בכיתה.
- MOEA/D – נבחר רנדומלית בשבילנו ולמדנו, מן הספרות, שהוא אפקטיבי גם כן ובמיוחד אפקטיבי יותר ביחס לNSGAII בבעיות עם יותר משני ממדים

4.1 NSGAII

אלגוריתם זה, הוא אלגוריתם גנטי שייחודו הוא האליטיזם ותהליך תיעדוף הכרומוזומים.

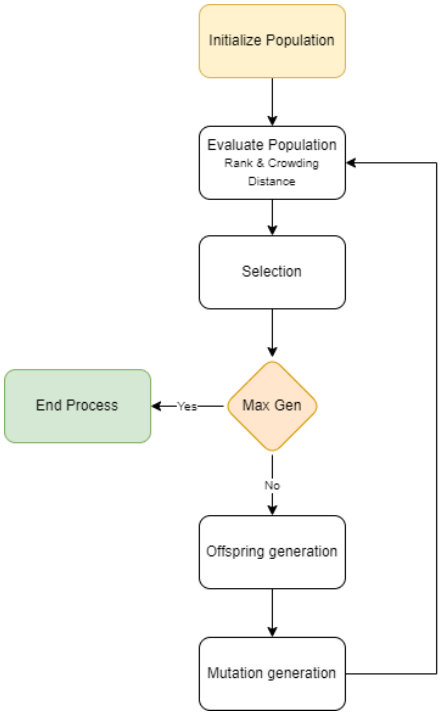
האלגוריתם מתחיל על ידי יצירת מספר פתרונות התחלתיים. כל פתרון הוא כרומוזום וכל גן בכרומוזום מייצג מיקום במסלול. סדר הגנים בכרומוזום מייצג את המסלול.

בשלב הבא, מדרגים את הפתרונות לפי ה- rank שלהם ודירוג משני של Crowding Distance.

בשלב הבא, אנו יוצרים דורות ומוטציות אשר מדורגות שוב, יחד עם אוכלוסיית ההורים (שלב האליטיזם) ובוחרים את הפתרונות המדורגים גבוה ביותר מבניהם (כך שנשאר עם אוכלוסייה קבועה בסוף כל מחזור).

במקרה שלנו, לא הטמענו תנאי עצירה אלא קבענו מספר דורות לתהליך.

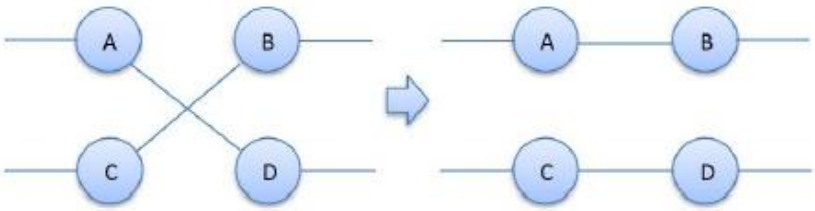
את תרשים האלגוריתם הקלאסי ניתן לראות באיור 1.



איור 1 - דיאגרמה כללית לNSGAII

ביצענו מספר שינויים לאלגוריתם הקלאסי של NSGAII על מנת למקסם את ביצועיו לבעיית TSP מכיוון שאינו מותאם בצורתו הבסיסית לבעיה זו:

- בחרנו בפונקציית Crossover Nearest Neighbor (CNN) (Satyanananda, 2015). פונקציה זו מניחה שהמרחק הקצר ביותר בין שני אתרים יוביל לפתרון האידיאלי ביותר. כך, מתוך שני הורים, נבחר גן אחר אקראית, ולאחריו נבחר הגן הבא בשרשרת של כל הורה הקרוב ביותר אליו. כך עוברים על כל הגנים עד שנוצר הצאצא
- לפונקציית המוטציה בחרנו בלוגיקת invert, אשר לוקחת חלק אקראי באורך אקראי בשרשרת הגנים והופכת את סדרו
- במהלך העבודה ראינו שאנו מתכנסים שוב ושוב, במספר פגמים נמוך יחסית של 20, למינימום מקומי ולא מצליחים "לנצח" את תוצאות אלגוריתם הבסיס (NN) בזמן סביר. לפיכך, חקרנו בספרות ולמדנו על אלגוריתם 2Opt אשר עקרונו הוא "התרת" קשרים במסלול תחת הנחת מוצא ש"אלכסון הוא אסון". כלומר, כאשר לוקחים שתי דרכים אקראיות, אם נבדוק את סכום מסלולם במצב המקורי ונשווה לסכום הדרך כאשר מחליפים בין נקודות ההתחלה והסוף (כפי שניתן לראות באיור 2) נקבל את המסלול הקצר ביותר. את הלוגיקה הזו הוספנו כפונקציית מוטציה שנייה(D. Nuraiman, 2018)



איור 2 - איור של לפני (משמאל) ואחרי (מימין) של אלגוריתם 2opt (D. Nuraiman, 2018)

4.2 MEOA/D

אלגוריתם זה לוקח את בעיית האופטימיזציה ומפרק אותה לN תת בעיות אופטימיזציה של סקלר, כל הבעיות נפתרות במקביל והאוכלוסייה מתפתחת בהתאם לפתרונות הטובים ביותר. כל תת בעיה עוברת אופטימיזציה בשימוש של מידע מהפתרונות השכנים.

לMOEA/D מספר תכונות:

- עושה אופטימיזציה לN בעיות סקלר מאשר אופטימיזציה לכל הבעיה, גישה קלה יותר למציאת ההתאמה (fitness) ומגוון.
- סיבוכיות חישוב נמוכה יחסית
- מתמודד עם בעיות של אופטימיזציה סקלר באופן טבעי
- מוצא "פשרה" בין פונקציות המטרה

ישנם מספר שיטות להמיר את ההערכת קו הפרטו למספר בעיות סקלר, הגישה שהשתמשנו בפרויקט היא Tchecycheff Approach:

$$\max g^{te}(x|\lambda, z^*) = \min\,(\lambda_i|f_i - z_i^*)$$

$$z^* = (z_1^*, \ldots\ldots, z_N^*)$$

$$\lambda = (\lambda_1, \ldots\ldots\ldots, \lambda_N)$$

Step 1) Initialization:

Step 1.1) Set $\mathrm{EP} = \emptyset$.

Step 1.2) Compute the Euclidean distances between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.

Step 1.3) Generate an initial population $x^1, \ldots, x^N$ randomly or by a problem-specific method. Set $FV^i = F(x^i)$.

Step 1.4) Initialize $z = (z_1, \ldots, z_m)^T$ by a problem-specific method.

Step 2) Update:

For $i = 1, \ldots, N$, do

Step 2.1) Reproduction: Randomly select two indexes k, l from $B(i)$, and then generate a new solution y from x^k and x^l by using genetic operators.

Step 2.2) Improvement: Apply a problem-specific repair/ improvement heuristic on y to produce y'.

Step 2.3) Update of z: For each $j = 1, \ldots, m$, if $z_j < f_j(y')$, then set $z_j = f_j(y')$.

Step 2.4) Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y'|\lambda^j, z) \leq g^{te}(x^j|\lambda^j, z)$, then set $x^j = y'$ and $FV^j = F(y')$.

Step 2.5) Update of EP:

Remove from EP all the vectors dominated by $F(y')$.

Add $F(y')$ to EP if no vectors in EP dominate $F(y')$.

Step 3) Stopping Criteria: If stopping criteria is satisfied, then stop and output EP. Otherwise, go to **Step 2**.

5. תוצאות

5.1 תיאור התוצאות

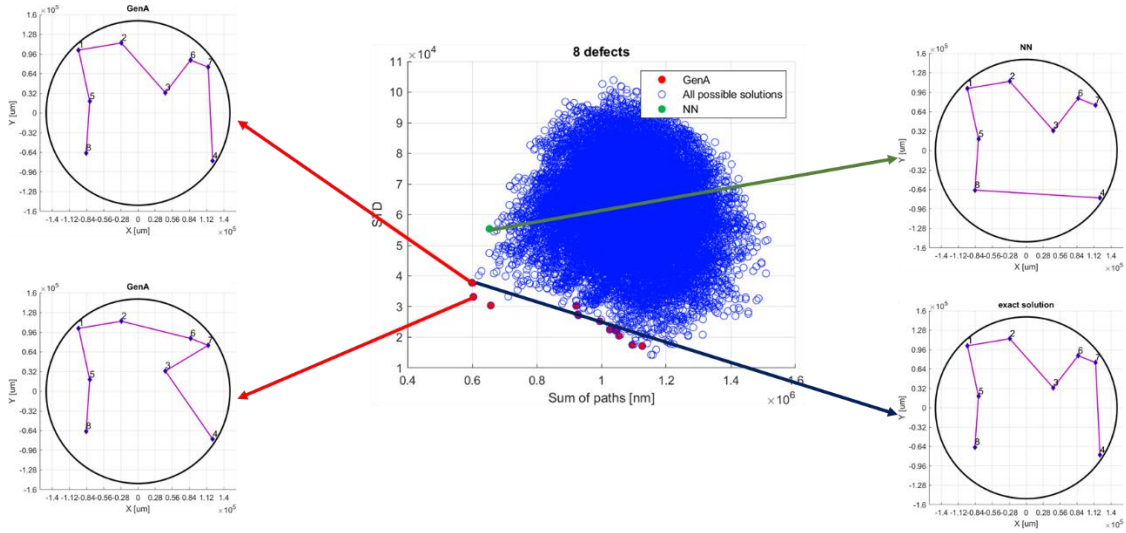
ראשית, רצינו לבדוק את ביצועי האלגוריתם למרחב הפתרונות המלא. מכיוון שמספר האפשרויות הוא $\frac{n!}{2}$, כאשר n זה מספר הפגמים, צמצמנו את הבדיקה עד 8 אתרים בלבד.

בטבלה 1, ניתן לראות השוואה של ביצועי NSGA2 ו-NN לעומת הפתרון האופטימלי האמיתי. המדד הנבחר להשוואה הוא הזמן המינימלי (אשר חושב ממרחקי התנועה של כל מסלול).

Travel Time - effeciency												
Locations	5			6			7			8		
Iteration	Exact	NN	GenA	Exact	NN	GenA	Exact	NN	GenA	Exact	NN	GenA
1	100%	89%	100%	100%	95%	100%	100%	94%	100%	100%	96%	100%
2	100%	100%	100%	100%	98%	100%	100%	86%	100%	100%	93%	100%
3	100%	100%	100%	100%	92%	100%	100%	97%	100%	100%	95%	96%
4	100%	93%	100%	100%	100%	100%	100%	97%	100%	100%	96%	100%
5	100%	95%	100%	100%	95%	100%	100%	100%	100%	100%	99%	100%
6	100%	97%	100%	100%	99%	100%	100%	93%	100%	100%	90%	100%
7	100%	100%	100%	100%	95%	100%	100%	93%	100%	100%	96%	100%
8	100%	94%	100%	100%	96%	100%	100%	99%	100%	100%	98%	100%
9	100%	90%	100%	100%	95%	100%	100%	97%	100%	100%	88%	100%
10	100%	92%	100%	100%	90%	100%	100%	95%	100%	100%	91%	100%
	Avg.	95%	100%		96%	100%		95%	100%		94%	100%

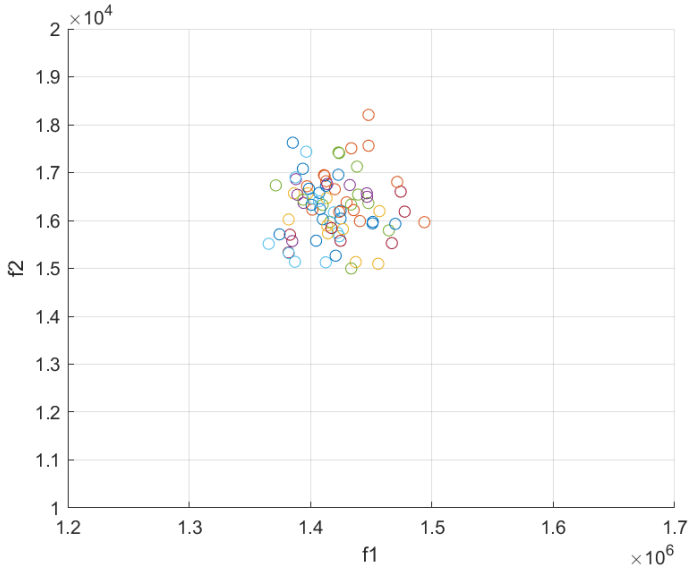
טבלה 1 – השוואת אלגוריתמים לפי זמן תנועה

באיור מספר 3 , ניתן לראות תצוגה גרפית של 8 פגמים עם כל הפתרונות האפשריים בכחול, פתרון NN בירוק, והחזית של NSGA2 באדום.

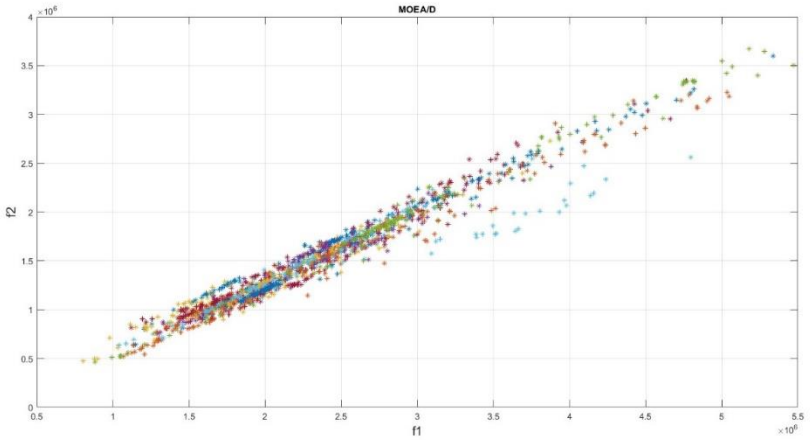


איור 3 -מרחב הפתרונות ל8 פגמים

לאחר וידוא ראשוני, עברנו לבעיה קשה יותר של 50 אתרים. כל אלגוריתם הורץ 30 פעמים, לצורך מדגם סטטיסטי מינימלי, על אותו סט של פגמים. בכל הרצה, בוצעו 75 חזרות (דורות) עם גודל אוכלוסייה של 50 פתרונות.
באיורים הבאים ניתן לראות את חזית הפרטו של כל אחת מ-30 הריצות לכל אלגוריתם ובנוסף חזית פרטו מאוחדת לכל אלגוריתם בנפרד.

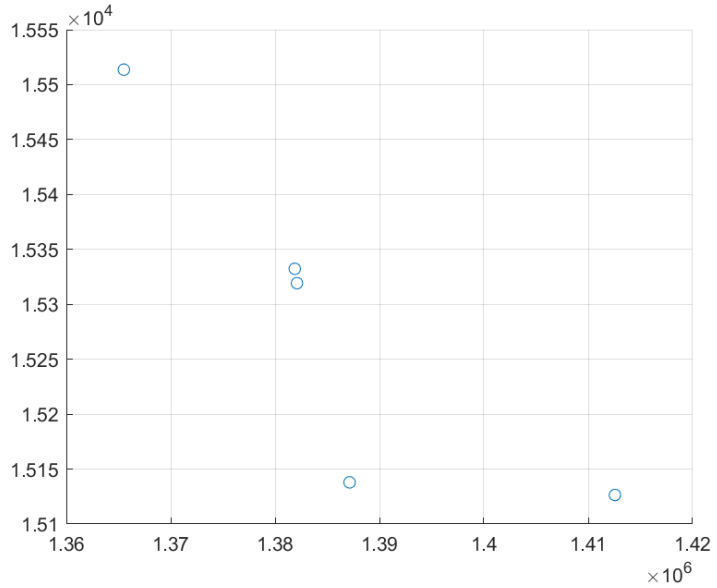


איור 4 - קבוצת האיחוד ל NSGA2

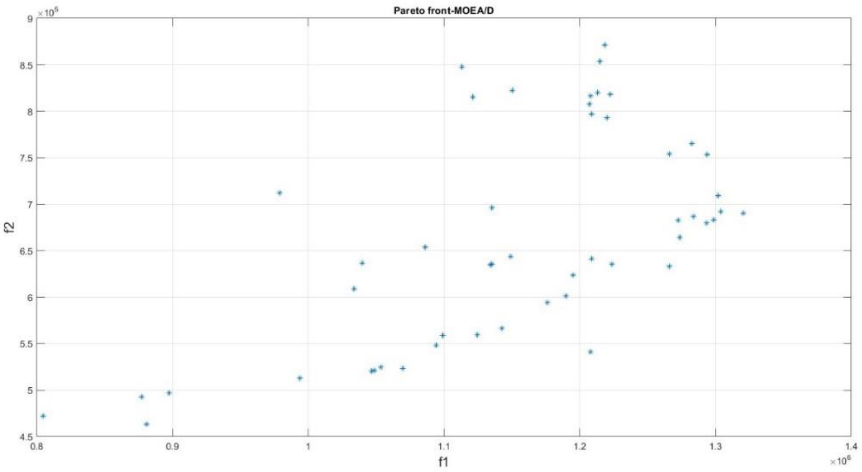


איור 5 - קבוצת האיחוד ל MOEAD/A

באיורים הבאים ניתן לראות חזית איחוד מכל 30 ההרצות לכל אלגוריתם:



איור 6 - חזית האיחוד לNSGA2



איור 7 - חזית האיחוד ל MOEA/D

5.2 מדדי השוואה

השתמשנו בשלושת מדדי ההשוואה כפי שנדרש וקיבלנו את התוצאות הבאות:

	MOEA/D			NSGAII		
Run	IGD	Spread	HV	IGD	Spread	HV
1	126082	0.7010	0.6823	8105	0.6667	0.2123
2	143564	0.6081	0.7981	4769	0.6236	0.4140
3	77383	0.7297	0.6141	11379	0.6550	0.4557
4	127738	0.6560	0.7178	3831	0.6667	0.2459
5	81700	0.9060	0.5585	17383	0.8987	0.3599
6	74342	0.8299	0.4142	5406	0.5995	0.3883
7	75550	0.5895	0.3465	0	0.6667	0.3356
8	74342	0.6247	0.2548	10164	0.7565	0.4901
9	75563	0.5660	0.3752	6233	0.5449	0.3572
10	137193	0.7250	0.7650	348	Inf	0.1423
11	74342	0.7042	0.5618	1115	Inf	0.1569
12	96509	0.4733	0.2709	4590	0.6667	0.2434
13	89216	0.6298	0.4985	0	0.7207	0.6644
14	74985	0.7890	0.5853	44824	0.6667	0.1816
15	74342	0.6807	0.4376	3604	0.7000	0.4129
16	90291	0.6677	0.7236	29437	0.6667	0.1848
17	136354	0.7178	0.7346	17256	0.7341	0.4067
18	81513	0.6280	0.5461	953	Inf	0.1101
19	76812	1.0259	0.4853	10679	0.6858	0.4501
20	75790	0.7429	0.7188	3012	0.6667	0.2274
21	75948	0.5917	0.4378	513	Inf	0.1531
22	104891	0.8550	0.5117	4525	0.6054	0.4525
23	80967	0.6688	0.4553	22871	0.9957	0.3976
24	101290	0.5380	0.5690	1180	0.6667	0.2693
25	83168	0.6937	0.5006	13003	0.6645	0.4000
26	74342	0.6846	0.4055	14808	0.6667	0.2597
27	92509	0.4714	0.5565	7308	0.6667	0.2061
28	157204	0.6025	0.7860	18659	0.5408	0.3862
29	74342	1.0276	0.6525	17592	0.6757	0.4437
30	95406	0.7158	0.4727	24994	0.6667	0.1201
AVG	93456	0.6948	0.5479	11019	0.6821	0.3176
Median	81606	0.6827	0.5513	7706	0.6667	0.3464
Variance	6.09E+08	0.0179	0.0223	1.04E+08	0.008228	0.01745

כפי שניתן לראות בטבלה, ישנן שתי חריגות סטטיסטיות:

1. בNSGAII, מדד ה IGD מתאפס בשני מקרים. שני מקרים אלו, הם כאשר כל הפתרונות בחזית של הרצה זו נמצאים בחזית המאוחדת ולכן מרחקם מהחזית הוא אפס.
2. בNSGAII, מדד הSpread שווה לInf במקרים בהם ישנו רק פתרון אחד בחזית של אותה ההרצה ולכן אין אפשר לחשב את הפיזור והמספר שנקבע למקרה זה הוא inf

בכל המקרים החרגנו את אותה תוצאה חריגה במדד זה מחישוב הממוצע, חציון וסטיית התקן.

5.3 ניתוח התוצאות

בשני האלגוריתמים, לא התכנסו לחזית "קלאסית" וחלקה. זאת מכיוון שהמקרה של TSP, הפתרון הוא דיסקרטי. ניתן לראות באיור 3 כיצד ישנו "ענן" פתרונות (לכל הפתרונות האפשריים) עם "שלוחות" לכיוון המינימום ששם נמצאים הפתרונות האופטימליים. לכן, מבנה החזיתות אינו מפתיע.

ניתן לראות הבדל חזותי בין שני האלגוריתמים באיורים 4 ו-5. מצד אחד, בNSGAII, ישנו "ענן" אחיד ביחס הממדים שלו, בניגוד לMEOA/D, אך מצד שני, מספר הפתרונות בכל חזית בNSGAII קטן משמעותית ממספר הפתרונות בMEOA/D.

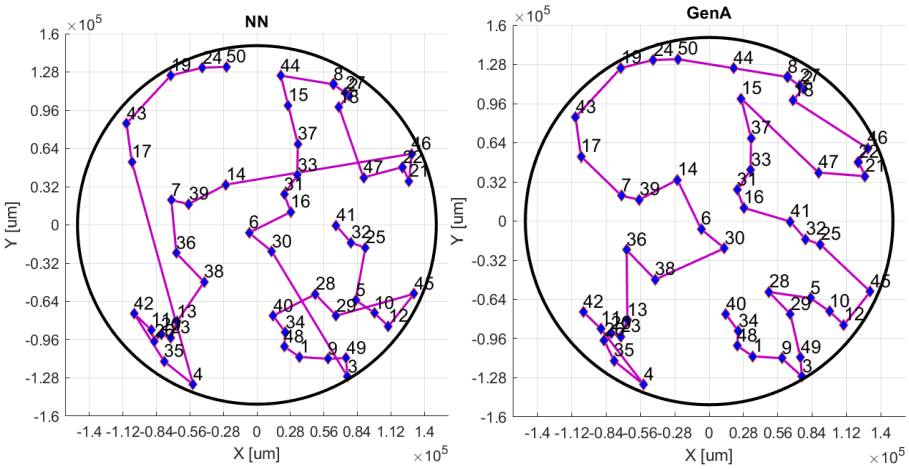
הבדל זה נראה, במידה מסוימת, גם בניתוח הסטטיסטי במדד הHV. מדד זה, מחשב את היחס בין התחום החוסם של חזית פתרונות של ריצה מסוימת לחזית האופטימלית (שמסונתזת מ30 הריצות יחדיו). במקרה שלנו, מדד זה ייתן אינדיקציה לצורת ה"ענן". בNSGAII, הענן סימטרי ואינו מתפרס ביחס לפתרונות בMEOA/D.

במבט על שאר המדדים, ניתן לראות שמדד הSpread דומה מאוד בין שני האלגוריתמים. דבר המצביע שלמרות המבנה השונה של הענן, הפיזור בין הפתרונות בכל הרצה דומה.
מדד הIDG שונה מהותית בין שני האלגוריתמים. כאשר בNSGAII הוא קטן פי 8 לערך ביחס לMEOA/D. כלומר, הפתרונות המתקבלים בNSGAII היו אחידים יותר (מדד זה מודד את ה"קרבה" של פתרון לחזית האיחוד) ומתיישב היטב עם הפריסה של הפתרונות בכל אחד מהאלגוריתמים. בנוסף, סטיית התקן במדד זה קטנה משמעותית גם כן בNSGAII ביחס לMEOA/D.

זמני החישוב, בשני האלגוריתמים זמן הריצה היה די דומה, היו בסביבות 30 עד 40 שניות לכל ריצה עם 75 דורות.

5.4 הצגת פתרונות אופייניים

בנוסף לאיור 3 המציג פתרונות אפשריים למקרה של 8 אתרים, באיור 8 ניתן לראות מסלול ניווט בין 50 אתרים בעזרת אלגוריתם NSGAII ולהשוואה בעזרת אלגוריתם NN.



איור 8 - מימין, פתרון 50 אתרים בעזרת NSGAII ; משמאל, פתרון לאותה הבעיה בעזרת NN

כפי שניתן לראות, המסלול בNSGAII "נקי" יותר, ללא הצלבות במסלול. זוהי הלוגיקה אשר דחפה אותנו להוסיף את המוטציה הנוספת המבוססת על אלגוריתם 2-opt.

6. דיון ומסקנות

בשני האלגוריתמים, לא קיבלנו חזית אחידה וקלאסית בגלל מבנה בעיית TSP שהיא בעיה דיסקרטית ולכן, בהינתן פונקציות המטרה הנבחרות, לא ניתן לקבל חזית חלקה. מכשול זה נראה חזותית כאשר ממפים את כל מרחב הפתרונות, כפי שרואים באיור 8. אי לכך, ישנה גם נטייה של האלגוריתם "ליפול" למינימום מקומי כאשר מתקרבים אל החזית מכיוון שענן הפתרונות יוצאות "שלוחות" אל עבר המינימום. חזותית, ניתן גם לראות הבדל בין שני האלגוריתמים. NSGAII התכנס לענן אחיד ואילו MEOA/D התכנס לרצועות צרות. הבדל זה ניתן לראות גם דרך המדדים הסטטיסטים. דרך הHV ניתן לראות את חוסר האחידות של MEOA/D ביחס לNSGAII, ובתוספת של IGD שנותן לנו אינדיקציה לגבי התכנסות הפתרונות. כלומר, ככל שערך מדד ה-IGD נמוך יותר, כך הפתרון "קרוב" יותר אל החזית האופטימלית. מכיוון שב-NSGAII הערך הממוצע וסטיית התקן קטנים, כמעט בסדר גודל, ביחס לMEOA/D, ניתן להסיק על ההבדל בין ביצועי האלגוריתמים גם ללא תצוגה חזותית.

הסיבה העיקרית להבדל בביצועי האלגוריתמים היא שיפורים שהכנסנו לNSGAII על מנת לשפר את ביצועיו (כפי שצוין בחלק 4.1). שיפורים אלו לא שולבו בMEOA/D ודאגנו שמספר הדורות יהיה זהה. אנו יכולים להעריך שאילו היינו מוסיפים שיפורים דומים או מגדילים את מספר הדורות היינו מגיעים לביצועים דומים.

יחד עם זאת, תוצאות NSGAII אינן אופטימליות ובמקרים רבים מתכנסות למינימום מקומי. ניתן לראות זאת בכך שבארבע מתוך שלושים ריצות קיבלנו רק פתרון אחד בחזית ובניסויים עם מספר רב יותר של פגמים (עד 1000) אמנם קיבלנו תוצאות טובות יותר מאלגוריתם הבסיס (NN) ובזמן יחסית סביר (מספר דקות). אנו מניחים שהשיפורים שהכנסנו באלגוריתם אמנם אגרסיביים ומכנסים את האלגוריתם מהר יחסית, אך באים על חשבון פיזור הפתרונות והתכנסות חזקה למינימום מקומי.

על מנת לשפר את הביצועים אנו מציעים לשנות את מוטציית הבסיס כך שנכריח אותה להיות שונה מהותית מהאוכלוסייה הקיימת. הנחת הבסיס שלנו היא שאנו נתכנס לרוב למינימום מקומי, ומכיוון שהחזית אינה "חלקה" יש צורך בשינוי דרסטי בפתרון על מנת ליצור פתרון שונה שיאפשר מרחב מקיף יותר של הפרטו.