**פרוייקט אופטימיזציה**

8) For BIPs and Bounded IPs, can heuristic search (A\*) work better than the classical solver?

a. focus only a subclass of problems (e.g., setcover, VC, knapsack, TSP…)

b. propose good H functions

c. Test GAs

בפרוייקט שלנו בחרנו להתרכז בבעית ה- Traveling Salesman .

יצרנו שלושה אלגוריתמים שונים שפותרים את הבעיה:

1. הפותר הקלאסי, שפותר את הבעיה בעזרת branch and bound .
2. הפותר ההיוריסטי, שפותר את הבעיה בעזרת אלגוריתם A\* .
3. הפותר הגנטי, שפותר את הבעיה בעזרת אלגוריתם גנטי.

האלגוריתמים שלנו מקבלים את הקלט שלהם בפורמט של מטריצת מרחקים בקובץ טקסט.

חילקנו את הבעיות שנשלח לאלגוריתמים למספר קטגוריות:

1. עלויות שוות הלוך וחזור – הבעיות הן מטריצות סימטריות.
2. מחירים שווים בין כל הערים – הבעיות כאן בצורה של מטריצות מרחקים בעלות עלות אחידה בין מעבר בין עיר לעיר.
3. הבדלים עצומים בין מחירים של ערים – הבעיות כאן בצורה של מטריצות עם טווחים גדולים מאוד של מספרים, יהיו מסלולים עם ערכים מאוד כבדים בהשוואה לאחרים.
4. גודל מספר הערים שנידרש לבקר – כאן נשווה לפי גודל המטריצות, שמסמן את מספר הערים

שאנחנו נדרשים לבקר.

יצרנו בסה"כ 3 בעיות עבור קטגוריות 1-3 ו-5 עבור קטגוריה 4 .

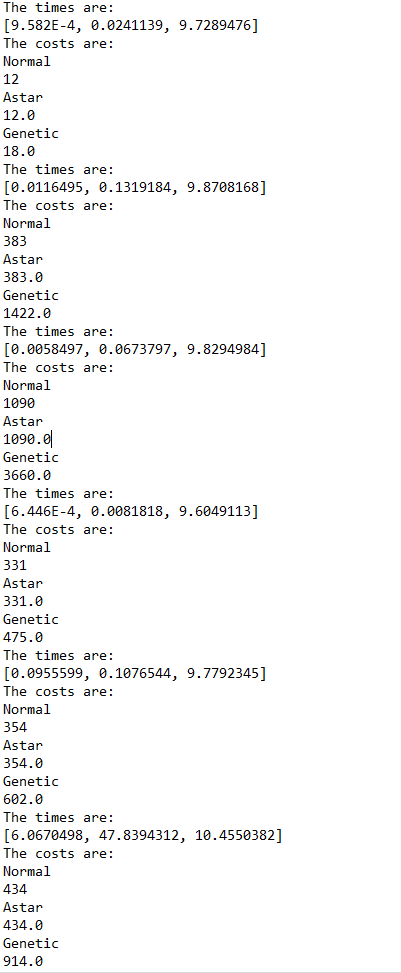
הרצנו את בעיות אלה על האלגוריתמים, והשוונו את הזמנים שלהם על מנת להחליט בממוצע איזה אלגוריתם יותר טוב על איזה סוג של בעיה.

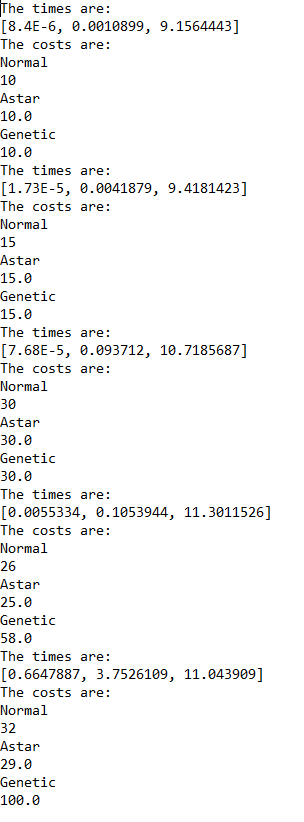
הנה התוצאות שקיבלנו:

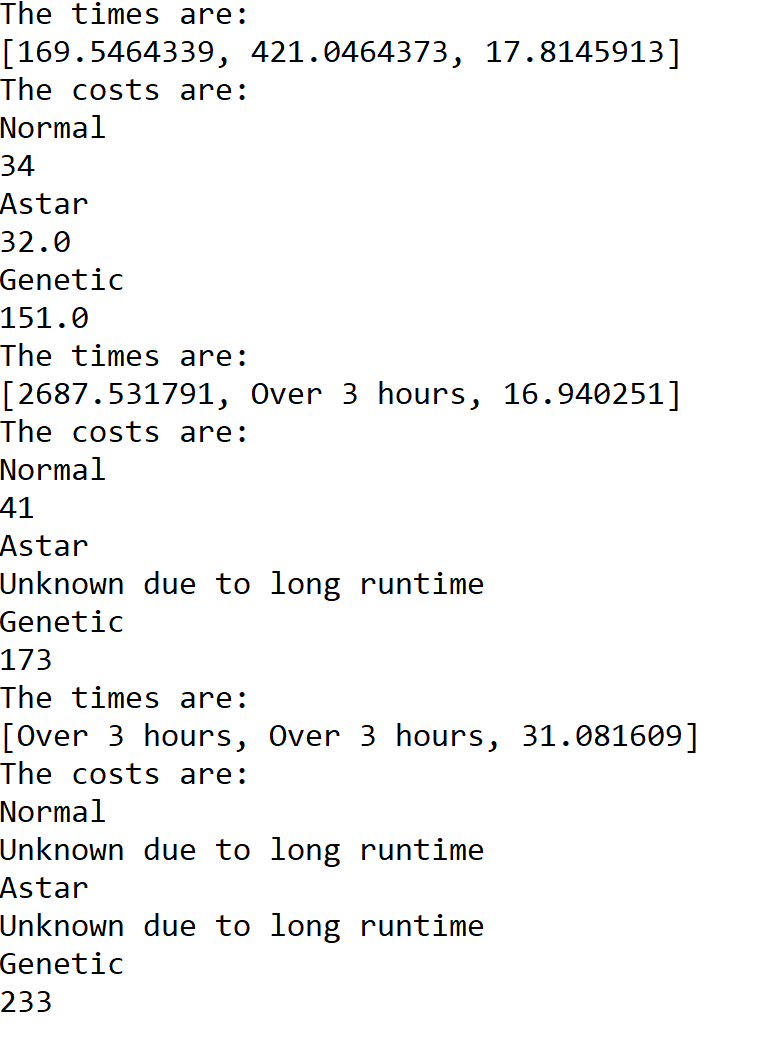
בצד שמאל אלגוריתם Branch and bound

באמצע אלגוריתם A\*

בצד ימין האלגוריתם הגנטי.







כפי שניתן לראות, האלגוריתם של ה- Branch and bound פעל בצורה מהירה יותר משאר האלגוריתמים על כלל הבעיות פרט לבעיות עם מספר רב של קדקודים. שם דווקא האלגוריתם הגנטי היה הכי מהיר ו- A\* הכי איטי.

מבחינת מחיר, Branch and bound ו- A\* מוצאים את המסלול האופטימלי (עם סטייה קטנה בזוג בעיות לטובת A\* ובעיות שלא נמצא המחיר עקב זמן ריצה רב) והאלגוריתם הגנטי משער פתרון אופטימלי שלרוב יצא גרוע יותר מהפתרון האופטימלי האמיתי.

A\* הוא סוג של הרחבה על Branch and bound. ב-Branch and bound, בכל איטרציה, מרחיבים את המסלול הכי טוב שמצאנו עד עכשיו. ב-A\*, במקום לבחור את המסלול הכי קצר כרגע, נבחר את המסלול עם המחיר המשוער הכי קצר, פלוס תוספת מהפונקציה ההיוריסטית.

Branch and bound תמיד ימצא פתרון אופטמלי (מסלול הכי קצר).

A\* תמיד ימצא פתרון אופטימלי אמ"מ הפונקציה ההיוריסטית אדמיסבילית, כלומר לעולם לא עושה overestimate. בחירת הפונקציה ההיוריסטית היא החלק הקשה ביותר באלגוריתם. היוריסטיקה טובה יכולה לייעל את החיפוש, ופונקציה לא טובה יכולה להגריע אותו (מהניסיונות שלנו על היוריסטיקות שונות זאת הייתה הכי טובה).

ובניגוד להם, אלגוריתם גנטי הוא אלגוריתם שההשראה אליו באה מביולוגיה. הרעיון הוא לתאר מרחב של פתרונות בתור genome, מערך סופי של גנים, ואז לברר אילו מהם הוא הפתרון הנכון. עושים זאת על ידי דירוג הפתרונות בצורה מסויימת, ואז צירוף הפתרונות הטובים ביותר ויצירת דור הבא של פתרונות אפילו יותר טוב, עד שהדירוג הזה מקובל על ידינו. דירוג זה נקרא fitness, בעוד ששילוב הפתרונות נקרא crossover . בגלל שהאלגוריתם מסתמך הרבה על אקראיות, יש סיכוי טוב שהוא ישלב פתרונות לא טובים. כדי לנסות לצמצם את תופעה זאת, אנחנו עושים משהו שנקרא mutation על אחוז קטן מה-genome שלנו. ניתן לייצר אלגוריתמים גנטיים עבור כל בעיה, אך הם אינם יהיו תמיד יעילים ומדויקים בכלל כמו אלגוריתמים שנוצרו ספציפית על מנת לפתור בעיות מסוימות.

בתור תשובה לשאלה:

For BIPs and Bounded IPs, can heuristic search (A\*) work better than the classical solver?

התשובה היא כן, אבל זה יקרה אך רק במקרים מסויימים, ותלוי ברובו על איכות הפונקציה ההיוריסטית.

ואלגוריתם הגנטי פחות מתאים לפתירת סוג הבעיה הזאת פרט לגרפים בעלי מספר רב של קודקודים .