**מימוש אלגוריתמים בתוכנה –**

**חלוקה מינימאלית של גרף**

**מגיש:** יובל אלפסי, 318401015, yuvalalfassi@gmail.com

**תאריך הגשה:** 19 יוני 2018

**מנחה:** פרופ' לאה אפשטיין

**תוכן עניינים**

מבוא ותיאור הבעיה – עמוד 3

גרפי קלט בהם השתמשתי – עמוד 4

פיתוח התוכנה וממשק משתמש – עמוד 5

פתרון באמצעות חיפוש מקומי – עמוד 6

פתרון באמצעות אלגוריתם גנטי – עמוד 7

פתרון באמצעות Branch and Bound – עמוד 10

סיכום – עמוד 12

**מבוא ותיאור הבעיה**

במסגרת הקורס מימוש אלגוריתמים בתוכנה, נתבקשנו לחקור מהאספקט הפרקטי היוריסטיקות לפתרון בעיות NP קשות במדעי המחשב. כל סטודנט קיבל בעיה שהאלגוריתם לפתרונה הוא אקספוננציאלי מבחינת זמן ריצה, והתבקש לממש היוריסטיקות שיפתרו את הבעיה.

חשוב לציין, שמדובר בבעיות שהן NP קשות, כך שניתן להאמין שלא קיים אלגוריתם שפותר אותן באופן יעיל. כמו כן, הבעיה שאני פותר היא בעיה בגרפים, בה כמות הקשתות היא **ריבועית** בכמות הצמתים. לדוגמא, עבור כמות של 100 צמתים, יהיו 10,000 קשתות, דבר המכביד על זמן הריצה. לצורך העניין – באלגוריתם גנטי של אוכלוסייה עם 100 פרטים, כאשר בגרף יש 100 צמתים, מדובר בכמות של מיליון משקלים מסוג double שיש לתחזק – כמות לפחות של 2Mega-Byte זיכרון, וזאת עוד מבלי לדבר על שאר המרכיבים.

אני קיבלתי את הבעיה של מציאת חלוקה מינימאלית של גרף לשלושה חלקים.

בהינתן גרף עם משקלים, עלי למצוא חלוקה של קדקודיו לשלוש קבוצות זרות, כך שהקבוצות הראשונה והשנייה ימנו רבע ממס' הקדקודים, והקבוצה השלישית תמנה מחצית ממס' הקדקודים. כמו כן, סכום משקלי הקשתות של כלל הקשתות שבכל אחת מהקבוצות צריך להיות מינימאלי.

מדובר בבעיית **מינימיזציה**.

ומתמטית:

Given Graph, find disjoint graphs,

Minimize:

לצורך פתרון הבעיה השתמשתי באלגוריתם גנטי כדי לפתורו וכן באלגוריתם חיפוש מקומי.

שיטות מציאת ה-Upper Bound וה-Lower Bound על בעיה זו ב-Branch & Bound לא היו מספיק טובות, ולפיכך פתרתי עם Branch & Bound בעיה קלה יותר – מציאת חלוקה של הגרף אך ורק לשתי קבוצות שוות בגודלן:

Given Graph, find disjoint graphs,

Minimize:

**גרפי קלט בהם השתמשתי**

לצורך בדיקת נכונות האלגוריתמים שפיתחתי, הרצתי את התוכנה על מספר סוגי גרפים שונים, בעלי מספר צמתים לבחירת המשתמש:

1. גרף סכום:
2. גרף מינימום:
3. גרף מקסימום:
4. גרף רנדומלי (משקלי הקשתות – מס' רציונלי רנדומלי בין 0 ל-1):
5. גרף זוגיות:
6. גרף הבנוי מ-4 קליקות (n – מספר הצמתים בגרף):

כפי שניתן לראות, השתמשתי בכמות מגוונת של סוגי גרפים, מה שמאושש את נכונות הריצה של האלגוריתמים והיוריסטיקות.

**פיתוח התוכנה וממשק משתמש**

את התוכנה יישמתי בשפת התכנות C#, שפת תכנות עילית, תוך שימת דגש על תכנות מודולרי ופונקציונאלי.

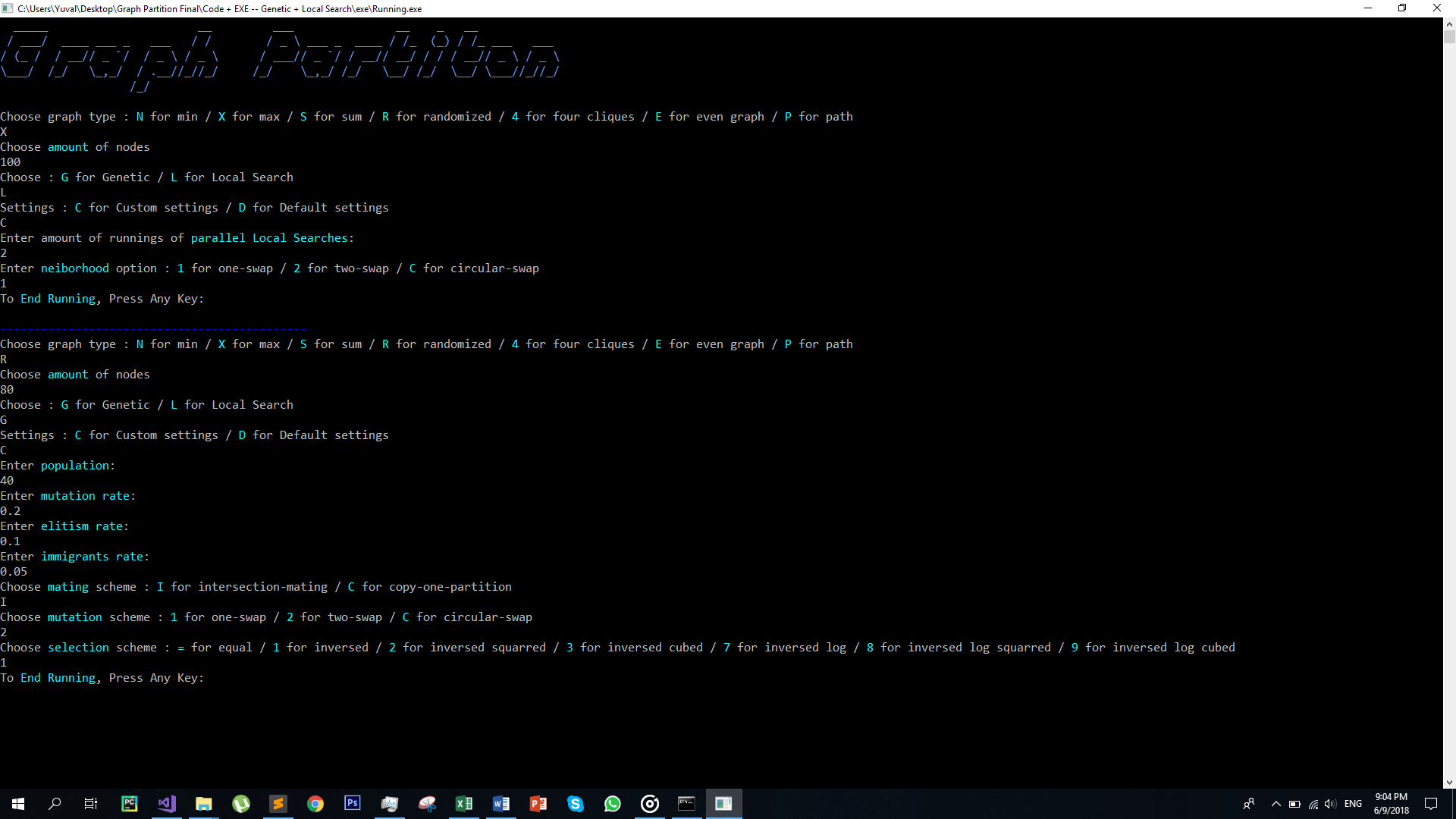
את התוכנה פיתחתי על מערכת הפעלה Windows 10 בעזרת סביבת העבודה Visual Studio.

את התוכנה מימשתי תוך שימת דגש על יעילות ושימוש חסכוני בזיכרון באמצעות Data Sharing. רבים ממבני הנתונים ייצגתי על ידי מבני נתונים פונקציונאליים, מה שהיטיב עם זמן הריצה של הבעיה האקספוננציאלית.

לצורך בחירת סוג הגרף עליו נרוץ, כמות הצמתים שלו, סוג היוריסטיקה ומאפייני היוריסטיקה, תכנתי REPL – read-eval-print-loop שמתקשר עם משתמש הקצה ועושה את אשר המשתמש מבקש.

כפי שהזכרתי מקודם, פיתחתי תוכנות נפרדות עבור ריצה של אלגוריתם Local Search ו-Genetic Algorithm לעומת ריצת Branch & Bound.

להלן צילומי מסך עם הסבר על אופן ההרצה:



בתחילת ההרצה מתבקשים לבחור את סוג הגרף שירוץ ואת כמות הצמתים שנרצה בו. לאחר מכן נבחר האם נרצה פתרון באמצעות חיפוש מקומי או אלגוריתם גנטי (התוכנה של Branch & Bound נפרדת, ולכן שם הבחירה תהיה אך ורק Branch & Bound). לאחר מכן בוחרים את הפרמטרים של הריצה: לדוגמא, בחיפוש מקומי את 'סביבת החיפוש' ובאלגוריתם גנטי את 'סוג הזיווג' ואת 'פונקציית המוטציה'; לאחר מכן האלגוריתם ירוץ ויכתוב כפלט קבצים לתיקייה בשולחן העבודה במחשב. ניתן להפסיק את הריצה באמצע באמצעות לחיצה על מקש כלשהו – אלגוריתם גנטי לא עוצר אף פעם, Local Search עוצר ו-Branch & Bound לוקח זמן אקספוננציאלי ולכן עוצר רק על קלטים קטנים.

**פתרון באמצעות Local Search**

בפתרון באמצעות חיפוש מקומי התחלתי מפתרון שרירותי רנדומלי של חלוקה של הגרף, וממנו התקדמתי כאשר סביבה הינה החלפה של שני צמתים או יותר מבחינת קבוצת Partition.

אפשרתי למשתמש להריץ במקביל מספר חיפושים מקומיים במקביל, ובכך לא להיות תלויים בבחירת פתרון התחלתי לא טוב שיתקע את האלגוריתם במינימום מקומי.

אפשרתי למשתמש לבחור בין מספר סביבות אפשריות:

1. החלפה אחת:

סביבה של פתרון הינה החלפה של שני צמתים בלבד, הם עוברים partition מאחד לשני. אפשר לצפות כאן שהאלגוריתם עשוי להיכנס למינימום מקומי, אך לא ניצפו מיקרים כאלה בהרצות רבות שערכתי.

1. שתי החלפות:

סביבה של פתרון הינה החלפה של שני זוגות של צמתים בין ה-Partitions שלהם/

1. החלפה מעגלית:

ישנם שלושה partitions. ההחלפה כאן הינה החלפה מעגלית של שלושה צמתים כאשר כל אחד ב-partition אחר.

1. כולם ביחד:

כל הסביבות ביחד. Mashup של שלושת השיטות שלעיל.

באנליזה שערכתי על הרצות עם שיטות סביבה שונות על גרף מסוג 'ארבע קליקות' התוצאות הינן:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Neighborhood** | **Graph Type** | **Running Time** | **Finishing Time** | **# Graph Nodes** | **Result Sum of Weights** | **Description** |
| One swap | 4 cliques | 0.123 minutes | 0.22 minutes | 40 | 100 | Global Minimum |
| Circular swap | 0.101 minutes | 0.32 minutes | 127 | Local Minimum |
| Two swaps | 0.113 minutes | 0.41 minutes | 100 | Global Minimum |
| All three together | 0.128 minutes | 0.43 minutes | 100 | Global Minimum |
| One swap | 4 cliques | 2.3 minutes | 3.7 minutes | 60 | 225 | Global Minimum |
| Circular swap | 7.5 minutes | 17.1 minutes | 263 | Local Minimum |
| Two swaps | 6.3 minutes | 8.8 minutes | 225 | Global Minimum |
| All three together | 5.0 minutes | 10.0 minutes | 225 | Global Minimum |
| One swap | 4 cliques | 19 minutes | 28 minutes | 80 | 400 | Global Minimum |
| Circular swap | 53 minutes | 80 minutes | 457 | Local Minimum |
| Two swaps | 41 minutes | 91 minutes | 400 | Global Minimum |
| All three together | 66 minutes | 92 minutes | 400 | Global Minimum |
| One swap | 4 cliques | 1.5 hours | 2.0 hours | 100 | 625 | Global Minimum |
| Circular swap | 3.1 hours | 6.8 hours | 645 | Local Minimum |
| Two swaps | 5.3 hours | 5.9 hours | 625 | Global Minimum |
| All three together | 2.2 hours | 5.1 hours | 625 | Global Minimum |

כפי שניתן לראות, אלגוריתם Circular Swap לא מגיע למינימום גלובלי כמו השאר.

**פתרון באמצעות אלגוריתם גנטי**

בפתרון שלי של האלגוריתם הגנטי שילבתי רעיונות מאלגוריתמים אבולוציוניים שונים.

הרעיון באלגוריתם גנטי שואב השראה מביולוגיה. אלגוריתם זה מתחיל מאוכלוסיית פתרונות, כך שבכל דור זוגות מהאוכלוסייה מזדווגים זה עם זה ויוצרים פתרונות עם גנים (תכונות) של שני ההורים. הפתרונות שמזדווגים הם הזוגות ה'טובים' וה'יפים', כאלה שמתאימים יותר למינימיזציה של הפונקציה הדרושה.

האלגוריתם הגנטי שתיכנתי מאפשר למשתמש מספר פרמטרים לבחירה:

1. גודל האוכלוסייה בדורות:

ניתן לקבוע מה גודל האוכלוסייה המבוקש. גודל אוכלוסייה קטן מדי ינוון את האלגוריתם הגנטי ולא יביא לידי ביטוי את הגיוון שבאוכלוסייה, בעוד שגודל אוכלוסייה גדול מדי יאט מאוד את קצב הרצת האלגוריתם וקצב ההתכנסות.

1. סוג מוטציה:

סוג המוטציה שתקרה – המוטציה עצמה מכילה רנדומיות בתוכה. ניתן לבחור בין החלפה של זוג צמתים אחד, שני זוגות צמתים או החלפה מעגלית.

1. אחוז סיכוי למוטציה:

כאשר שני הורים מזדווגים, בסיכוי מסוים תעבור 'מוטציה' על הגן של הולד שנוצר. ניתן לקבוע מה הסיכוי שדבר כזה יקרה בכל זיווג. לביטול מאפיין זה נבחר סיכוי של 0%.

1. אחוז אליטיזם:

זהו גם כן מאפיין מאלגוריתמים אבולוציוניים. מאפיין זה שומר אחוז מסוים מהאוכלוסייה לדור הבא. לדוגמא, אם אחוז האליטיזם יהיה 0.3, אזי 30% מהטובים ביותר של האוכלוסייה הנוכחית יישמרו לדור הבא. מטרת מאפיין זה היא לשמור מדור לדור את הגנים הטובים והמשובחים ביותר שלא נרצה שייעלמו.

1. אחוז מהגרים:

זהו גם כן מאפיין מאלגוריתמים אבולוציוניים. בכל אחוז מסוים מהאוכלוסייה יהיו גנים חדשים שהם מעין 'מהגרים' שהגיעו מארצות אחרות (באנלוגיה לביולוגיה). מאפיין זה מגביר את ה-diversity, מאפשר יציאה ממינימום מקומי במקרי קיצון, ומדמה באופן נאמן יותר אבולוציה. לביטול מאפיין זה נקבע את אחוז המהגרים להיות 0%.

1. סוג זיווג:

יישמתי שתי שיטות זיווג;

שיטת חיתוך (intersection):

בשיטה זו הגן הבן יכיל בכל partition תחילה את כל הצמתים שהיו **גם** אצל האב וגם אצל האם. לאחר מכן נעבור על שאר הצמתים שאצל האם ונמלא אותם באותם מקומות, אחרי זה את של האב, ואת השאר נמלא בהתאם לאיפה שיש.

שיטת ההעתקה (copy):

בשיטה זו תחילה נבחר partition רנדומלי אצל האם. נעתיק את הצמתים שאצלו לאותו partition של הבן. אחרי כן נעבור על הצמתים של האב שב-partitions האחרים ונשים אותם אצל הבן. את שאר הצמתים נמלא בהתאם לאיפה שיש.

כפי שניתן לראות, הבן שנוצר שומר על ה'מאפיינים' (הגנים) של הוריו.

1. פונקציית נרמול פונקציית הכשירות:

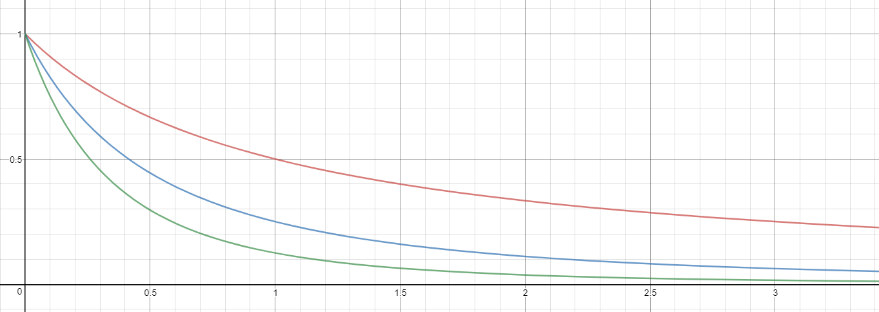
אחד העקרונות המנחים באלגוריתם גנטי שהופכים אותו ליותר טוב מאשר הילוך מקרי במרחב הוא שהפרטים באוכלוסייה שמזדווגים ומעבירים את הגנים שלהם הלאה לצאצאים שלהם הם אותם פרטים 'יפים' ו'טובים' – אלו שפונקציית המטרה שלהם היא מינימאלית. בחירת האב והאם נעשית על ידי מעין 'גלגל המזל' כך שלכל פרט יש סיכוי להיבחר באופן פרופורציונאלי לפונקציית הכשירות שלו, כאשר ככל שלפרט סכום משקלי הקשתות קטן יותר, כך סיכוייו להיבחר להזדווגות גדול יותר.

לצורך כך, בחרתי במספר פונקציות אותן המשתמש יכול לבחור, פונקציות שדועכות מהר יותר מביאות משקל חזק יותר לפרטים טובים ומעלים את סיכוייהם להזדווג, בעוד שפונקציות שדועכות לאט מעלות את אחוז ה-diversity בדורות.

נשים לב ש'גלגל המזל' בבחירת ההזדווגויות נעשה תוך נרמול סכום ההסתברויות ל-1 כך שתהיה פרופורציה בין ערך הפונקציה לסיכויי ההזדווגות.

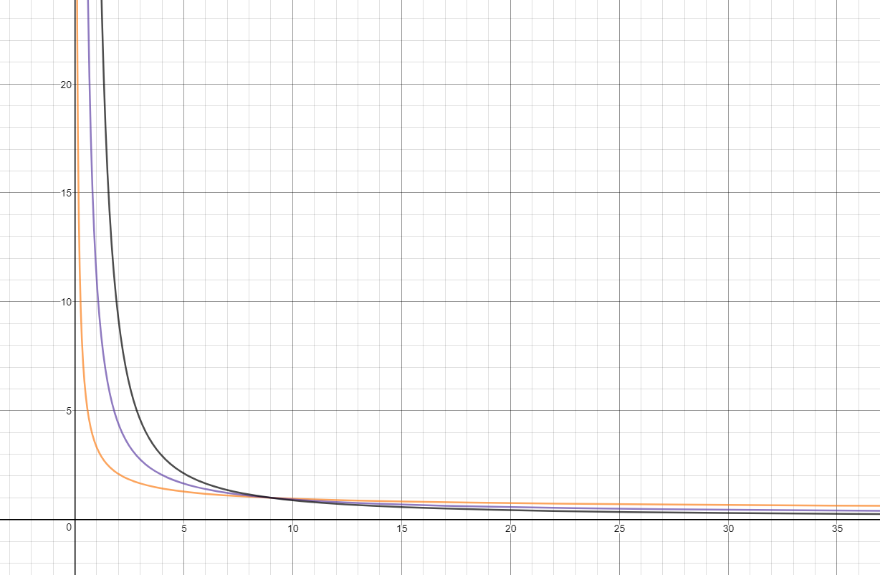
פונקציות המטרה הינן:

שרטוט של פונקציות אלה נמצא בתרשים שלהלן. נשים לב שהחזקות במכנה של הפונקציות גורמות לדעיכה חזקה יותר ובכך לחיזוק מאסיבי יותר של פרטים טובים אך בא בעת לפחות diversity.



פונקציה זו הביאה לכל הפרטים סיכוי שווה להזדווג. אפשרתי את פונקציה זו אך ורק לצורך בקרה. אכן, פונקציה זו הייתה פחות טובה, והיא למעשה שקולה למעין הילוך מקרי במרחב.

שרטוט של פונקציות אלה נמצא בתרשים שלהלן. גם כאן, החזקות במכנה של הפונקציות גורמות לדעיכה חזקה יותר ובכך לחיזוק מאסיבי יותר של פרטים טובים אך בא בעת לפחות diversity.



**פתרון באמצעות Branch & Bound**

פתרון באמצעות שיטת Branch & Bound יוצר למעשה עץ של אופציות שונות לבניית הכמות האקספוננציאלית של הפתרונות האפשריים. האינטליגנציה מאחורי אלגוריתם זה הוא שבכל שלב ביניים בבניית הפתרונות נקבע חסם עליון על ריצת האלגוריתם באמצעות בנייה חכמה של הפתרון, וחסם תחתון של הבעיה באמצעות אובזרבציות חכמות עליה. החסם העליון והתחתון מאפשרים לקטום אזורים בעץ מבלי צורך לפתח אותם מפאת תובנות שמסיקים ותכונות מתמטיות של הבעיה. כתוצאה מקטימת שלבי ביניים בעץ האפשרויות שנוצר, נצפה להפחתה בסדר גודל של זמן הריצה של האלגוריתם מ'אקספוננציאלי' ל-'אקספוננציאלי מסדר גודל קטן יותר', וכך לאפשר מציאת פתרונות בזמן ריצה סביר גם לפתרונות יותר גדולים מאשר כאשר בודקים את כל האפשרויות (brute force).

למרבה הצער, על הבעיה המקורית של חלוקת הגרף לשלוש קבוצות באופן אסימטרי, לא הצלחתי להגיע לפתרונות טובים ולחסמי Branch & Bound משביעי רצון. לפיכך, את שיטת ה-Branch & Bound הפעלתי על בעיה של חלוקת הגרף לשני חלקים בלבד, כאשר כל חלק בגודל של מחצית ממספר הצמתים. כך, כאשר כמות הקבוצות יותר קטנה, גודל עץ האפשרויות קטן יותר. כמו כן, כאשר גודל הקבוצות שווה, אין אסימטריה שמקשה על מציאת חסמים חכמים.

חסם עליון:

לצורך מציאת פתרון ספציפי 'טוב' בהינתן חלק מהצמתים שכבר הושמו, רציתי 'לבנות' משם פתרון חמדני 'טוב', שקרוב לוודאי 'יספק את הסחורה' ויהווה חסם עליון טוב לבעיה בהינתן אותו מידע מקדים של הצומת בעץ האפשרויות.

לצורך כך, תחילה רציתי לשים כל אחד מצמתי הגרף הנותרים ב-partition בו סכום המשקלים יהיה מינימאלי. רעיון זה לא עבד טוב, כיוון שהיה bias לשים צומת ב- partitionבו יש פחות צמתים, כיוון שכך יהיה סכום משקלים קטן יותר.

לפיכך, שמתי כל צומת ב-partition בו סכום המשקלים מנורמל ביחס לכמות הקשתות שיש בו, שהוא , יהיה מינימאלי (n – מספר הצמתים ב-partition). כך, דאגתי שהמשקל הממוצע בכל partition יהיה קטן ככל הניתן באופן היוריסטי, וטיפלתי בתלות בכמות הצמתים שב-partition בזמן נתון.

אפוא, כך בניתי באופן חמדני את הפתרון בהינתן צמתים שנמצאים כבר בכל אחד מה-partitions.

חסם תחתון:

לצורך קביעת חסם תחתון בהינתן צמתים בכל אחד מה-partitions, לקחתי בחשבון לכל צומת ב-partition את השכנים שלו באותו partition ואת שאר ה- הטובים ביותר. כמו כן, לקחתי את הצמתים בעלי סכום המשקלים נמוך ביותר כאשר המחיר שלהם חושב כמחיר ביחס למשקלים של הקשתות עם הצמתים שב-partition ולקשתות ל-

הצמתים בעלי המשקל הנמוך ביותר.

נשים לב שלמעשה לא ביצענו חלוקה ספציפית של הגרף, אלא לקחנו מקרה שלא יכול לקרות. לכל צומת חישבנו את ה- הצמתים הטובים ביותר שיהיו שכנים שלו, אותם שיקללנו למחיר כולל. לבסוף, את כל הסכום חילקתי בשתיים, כי כל מחיר חושב פעמיים 'בגדול'.

תוצאות:

את תוצאות ה-Branch & Bound ריכזתי בטבלה שלהלן, שמציגה את תוצאות ההרצה ביחס לפיתוח העץ כולו (כאשר החסם התחתון הינו 0 תמיד). כפי שניתן להסיק, ה-Branch & Bound מוריד בסדר גודל מעריכי את זמן הריצה של האלגוריתם.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Graph Nodes** | **B&B Tree Nodes** | **MinBound = 0 Tree Nodes** | **B&B Sec Time** | **MinBound = 0 Sec Time** | **Result Price** |
| 8 | 3 | 251 | 0.0019552 | 0.0802556 | 8 |
| 12 | 139 | 3431 | 0.1079612 | 2.3058725 | 18 |
| 16 | 183 | 48619 | 0.2912597 | 49.1454813 | 32 |
| 20 | 913 | 705431 | 2.7009588 | 1128.224792 | 50 |
| 24 | 8943 | 9876034 | 31.4022695 | 10605.31305 | 72 |
| 28 | 12207 | 108636374 | 65.7160494 | 131505.8818 | 98 |
| 32 | 62789 |  | 467.6546647 |  | 128 |
| 36 | 502312 |  | 2805.927988 |  | 162 |
| 40 | 4269652 |  | 14029.63994 |  | 200 |
| 44 | 21348260 |  | 64536.34373 |  | 242 |

כמות הצמתים בעץ ה-B&B כתלות בכמות הצמתים בגרף, **log-scale**:

זמן ריצה בשניות כתלות בכמות הצמתים בגרף, **log scale**:

**סיכום**

לסיכום, ארצה לציין שמטלות הקורס היו מעניינות ומאתגרות. נחשפתי לעולם מעניין של יישום פרקטי של בעיות NP-קשות עם שיטות שונות. התנסיתי הלכה למעשה בתיאוריות ופרקטיקות לא קונבנציונאליות במדעי המחשב. ארצה להודות תודה מיוחדת להנחיה של פרופ' לאה אפשטיין, שליוותה את תהליך יישום התוכנה והכווינה בהתאם לתבונות שלה על בעיות מסוג זה.