מציאת ערך הזרימה המירבי ברשת זרימה

באמצעות אלגוריתם Ford-Fulkerson

בשילוב שיטת ה-Bit-Scaling של Gabow

מגיש	ליאור וונש
מס׳ זהות	
סמסטר	סמסטר א׳ 2020
מחלקה	הנדסת תוכנה
מנחה	ד״ר ילנה קליימן

תוכן עניינים

תקציר המחקר

נשתמש בשיטת ה-Bit Scaling של Gabow. (ראה מקור 3)

אלגוריתם שמשתמש בשיטה זו נקרא אלגוריתם Bit-Scaling שצורת פתרון הבעיה שהוא מציע היא על ידי התחשבות בתחילה רק בסיבית המשמעותית ביותר של כל ערך קלט (כגון קיבול קשת) בייצוגו כמספר בינארי.

לאחר מכן, הוא מרחיב את הפתרון הראשוני על ידי התבוננות בשתי הסיביות המשמעותיות ביותר. האלגוריתם מסתכל בהדרגה על יותר ויותר סיביות לפי חשיבותן, משכלל את הפתרון בכל פעם, עד שבדק את כל הסיביות ומחשב פיתרון סופי.

נשתמש בשיטה זו כדי לתכנן אלגוריתם Ford-Fulkerson שמשתמש באלגוריתם Bit-Scaling לפתרון בעיית מציאת ערך זרימה מירבי ברשת זרימה.

האלגוריתם נועד לשיפור זמני הריצה של אלגוריתם Ford-Fulkerson עבור קלטים מסוימים.

נוכיח את נכונות האלגוריתם, נבצע ניסויים תוך יישומו בתוכנה ונסיק מסקנות לפי התוצאות שיתקבלו.

בנוסף, נשתמש בשיטה על מנת לפתור בעיה של שיבוץ משימות במעבד.

רקע כללי

במתמטיקה וליתר דיוק בתורת הגרפים, גרף הוא מבנה המסתכם בקבוצת אובייקטים שבהם זוגות מהאובייקטים במובן מסוים "קשורים".

האובייקטים תואמים להפשטות מתמטיות הנקראות קודקודים או צמתים וקשרים בין זוגות הקודקודים שנקראים צלעות או קשתות.

בדרך כלל, גרף מתואר בצורה דיאגרמתית כסט של נקודות או עיגולים עבור הקודקודים וקווים או עקומות לקשתות. גרפים הם אחד ממושאי הלימוד במתמטיקה דיסקרטית.

הקשתות עשויות להיות מכוונות או לא מכוונות, מה שקובע אם הוא הגרף נקרא לא מכוון או שמא גרף מכוון בהתאמה.

בגרפים שאעבוד איתם, הקשתות יהיו מכוונות ולכן גם הגרף ובו בין כל שני צמתים יהיו עד שתי קשתות, אחת מכל צומת למשנהה.

נעבוד גם עם גרפים ממושקלים, כלומר גרפים שבהם לכל קשת יש משקל שהוא ערך שמייצג עלות, אורך או כל מדד אחר.

רשת זרימה היא גרף מכוון בו לכל קשת יש קיבולת, שזו הגבלה על כמות הזרימה שיכולה לזרום בקשת זו. רשתות זרימה משמשים בדרך כלל למידול בעיות הכרוכות בהובלת פריטים בין מיקומים, באמצעות רשת נתיבים בעלי קיבולת מוגבלת. דוגמאות לכך כוללות תעבורה ברשת דרכים, נוזל ברשת צינורות, חשמל ברשת רכיבי מעגל ורשת של העברת תוכן מדיה.

ברשתות הזרימה שאעבוד איתם, יהיו שני קודקודים מיוחדים, האחד s ייקרא מקור (source) שממנו "תצא" כלל הזרימה והשני t בור (sink) שאליו בסופו של דבר תזרום.

אלגוריתם Ford-Fulkerson הוא אלגוריתם שמתמודד עם בעיית מציאת זרימה מקסימלית ברשת זרימה.

כלומר, כמות הזרימה המקסימלית החוקית שיכולה לעבור ברשת בכל יחידת זמן.

רקע תיאורטי

רשת זרימה (ראה מקור 4,2)

נתון גרף מכוון עם צמתים וקשתות וסימונו $G = (V, E)$ בהתאמה.

לכל קשת מצומת u לצומת v, שסימונה (u, v), יש משקל $c(u, v) > 0$ שהוא הקיבול (Capacity) שלה. אם הקשת היא לא חלק מהגרף, אז $c(u, v) = 0$.

הנחה: הגרף קשיר, כלומר ניתן להגיע מכל צומת לכל צומת בעזרת הקשתות, ולכן מספר הקשתות בגרף גדול ממספר הצמתים בגרף, $|E| \geq |V| - 1$.

בעיית הזרימה המקסימלית:

מהו הקצב המקסימלי שבו אפשר להזרים חומר ברשת מ-s ל-t מבלי לחרוג מקיבולי הקשתות?

פירמול הבעיה (ראה מקור 1)

פונקציה $f: VxV \rightarrow R$ נקראת פונקצית זרימה אם מתקיימים התנאים הבאים:

- אילוץ הקיבול: על אף קשת לא מוזרם יותר מהקיבול שלה.

$$\forall u, v \in V: f(u, v) \leq c(u, v)$$

- אנטי-סימטריה: עבור כל קשת עם זרם, יש קשת נגדית עם הזרם שלילי.

$$\forall u, v \in V: f(u, v) = -f(v, u)$$

- שימור הזרימה: לכל קודקוד u השונה מ-s ו-t, כמות הזרימה היוצאת מ-u שווה לכמות הזרימה הנכנסת אליו.

$$\forall u \neq s, t: \sum_{v \in V} f(u, v) = 0$$

הגודל $f(u, v)$ נקרא הזרימה נטו מצומת u לצומת v.

הגדרה: ערך הזרימה |f| הוא הזרימה שיוצאת מ-s והיא שווה לזרימה שנכנסת ל-t.

$$|f| = \sum_{v \in V} f(s,v) = \sum_{v \in V} f(v,t)$$

הוכחה:

$$0 = \sum_{u} \sum_{v} f(u,v) = \sum_{v} f(s,v) + \sum_{v} f(t,v) = \sum_{v} f(s,v) - \sum_{v} f(v,t)$$

אנטי-סימטריה שימור הזרימה אנטי-סימטריה

<u>בעיית הזרימה המירבית</u>

בהינתן רשת זרימה G, קודקוד מקור s וקודקוד יעד t, מהי הזרימה מ-s ל-t המביאה את ערך הזרימה |f| למקסימלי?

קשת רוויה: קשת שבה עוברית זרימה מירבית אפשרית $f(u,v) = c(u,v)$.

קשת משפרת: קשת (u,v) שאפשר "להזרים" לאורכה עוד זרימה $f(u,v) < c(u,v)$.

מסלול משפר: מסלול מ-s ל-t שמורכב כולו מקשתות משפרות.

<u>הרשת השיורית</u>

קיבול שיורי c_f: כמות הזרימה שאפשר עדיין להזרים לאורך קשת.

$$c_f(u,v) = c(u,v) - f(u,v)$$

קשת משפרת: קשת (u,v) משפרת אם $c_f(u,v) > 0$.

מסלול משפר: מסלול מ-s ל-t שמורכב כולו מקשתות משפרות.

הקיבול השיורי של מסלול P: הקיבול השיורי המינימלי לאורך P.

$$c_f(P) = \min_{(u,v) \in P} c_f(u,v)$$

הגרף השיורי: הגרף G_f שמכיל את כל הצלעות המשפרות של G, כאשר משקלי הצלעות הם הקיבולים השיוריים.

מסקנה: כל מסלול מ-s ל-t ב-G_f הוא מסלול משפר ב-G ולהיפך.

למה: יהיו f ו-f' פונקציות זרימה בגרפים G ו-G_f בהתאמה, אז הפונקציה $f + f'$ היא פונקציית זרימה עבור G, וכמו כן $|f + f'| = |f| + |f'|$.

הוכחה:

- אנטי-סימטריה:

$$(f + f')(u, v) = f(u, v) + f'(u, v) = -f(v, u) - f'(v, u) = -(f + f')(v, u)$$

- אילוצי קיבול:

$$\begin{aligned}(f + f')(u, v) &= f(u, v) + f'(u, v) \leq f(u, v) + c(u, v) \\ &= f(u, v) + \big(c(u, v) - f(u, v)\big) = c(u, v)\end{aligned}$$

- שימור הזרימה:

$$\sum_{v \in V}(f + f')(u, v) = \sum_{v \in V}\big(f(u, v) + f'(u, v)\big) = \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) = 0$$

- ערך הזרימה:

$$\begin{aligned}|f + f'| &= \sum_{v \in V}(f + f')(s, v) \\ &= \sum_{v \in V}\big(f(s, v) + f'(s, v)\big) = \sum_{v \in V} f(s, v) + \sum_{v \in V} f'(s, v) = |f| + |f'|\end{aligned}$$

הבחנה: בהינתן פונקציית זרימה f ומסלול שיפור P נגדיר את הפונקציה הבאה:

$$f_p(u, v) = \begin{cases} c_f(P) & (u, v) \in P \\ -c_f(P) & (v, u) \in P \\ 0 & \text{otherwise} \end{cases}$$

f_p היא פונקציית זרימה עבור G_f כך שמתקיים $|f_p| = c_f(P) > 0$.

מסקנה: תהי f פונקציית זרימה עבור G ויהי P מסלול שיפור ב-G_f, אז $f^* = f + f_p$ היא פונקציית זרימה עבור G ו- $|f^*| = |f| + |f_p| > |f|$.

<u>אלגוריתם Ford-Fulkerson</u>

- אתחול:

זרימה 0 בכל הקשתות: $f(u,v) = f(v,u) = 0$.

הגרף השיורי G_f שווה לגרף המקורי G: $c_f(u,v) = c(u,v)$.

- כל עוד יש מסלול מ-s ל-t בגרף G_f:

א. נמצא מסלול P בגרף G_f ונחשב את קיבולו השיורי $c_f(P)$.

ב. נגדיל את הזרימה ב-G לאורך P ב-$c_f(P)$:

$$\forall(u,v) \in P: f(u,v) = f(u,v) + c_f(P), f(v,u) = -f(u,v)$$

ג. נעדכן את הגרף השיורי לאורך המסלול P:

$$\forall(u,v) \in P: c_f(u,v) = c(u,v) - f(u,v), c_f(v,u) = c(v,u) - f(v,u)$$

<u>סיבוכיות זמן ריצה</u>

סימון: F = ערך הזרימה המירבית.

טענה: אם כל הקיבולים שלמים אז ייתכנו לכל היותר F איטרציות.

הוכחה: בכל איטרציה הזרימה גדולה לפחות ב-1.

מסקנה: זמן הריצה הוא $O(mF)$, כאשר $|E| = m$ ומשתמשים באלגוריתם BFS שמתואר בעמוד הבא.

החסם הדוק, כלומר ניתן לבחור מסלולים משפרים באופן כזה שיגרום לכך שאכן יהיו F איטרציות.

<u>בחירת מסלול משפר</u>

אלגוריתם BFS :

אלגוריתם חיפוש לרוחב הוא אלגוריתם המשמש למעבר על צומתי גרף, לרוב תוך חיפוש צומת המקיים תכונה מסוימת. צומת כלשהו V בגרף נקבע להיות הצומת ההתחלתי והאלגוריתם עובר על כל הצמתים במרחק צלע אחת מ-V, ואז על כל הצמתים במרחק 2 צלעות וכן הלאה. צורת חיפוש זו היא חיפוש לרוחב הגרף, בניגוד לחיפוש לעומק הגרף.

פלט האלגוריתם, המכונה עץ החיפוש לרוחב (BFS), מקיים את התכונה שהמסלול משורש העץ לכל אחד מהצמתים הוא המסלול בעל מספר הצלעות הנמוך ביותר בגרף המקורי, ובגרף שאינו גרף ממושקל הוא גם המסלול הקצר ביותר.

סיבוכיות הזמן של האלגוריתם בגרף קשיר היא $O(V + E)$, כאשר V מייצג את קבוצת הצמתים בגרף ו-E מייצג את קבוצת הקשתות בגרף ומתייחסים לגודל הקבוצות.

אלגוריתם Edmonds-Karp :

אם בוחרים מסלול משפר בעזרת BFS, כלומר מוצאים מסלול קצר ביותר, אז מספר האיטרציות הוא $O(nm)$ ולכן זמן הריצה הכולל הוא $O(nm^2)$.

משפט זה תקף תמיד (גם עבור קיבולים שאינם שלמים), כאשר $|V| = n, |E| = m$.

מסקנה :

אם הקיבולים שלמים ובוחרים מסלולים משפרים על ידי BFS אז זמן הריצה הוא

$O(\min\{nm^2, mF\})$

<u>הוכחה שהזרימה המתקבלת מירבית</u>

חתך ברשת הזרימה: חתך (S,T) הוא חלוקה של V לשתי קבוצת זרות, S ו- $T = V\backslash S$ כאשר $s \in S, t \in T$.

קיבול חתך: סכום הקיבולים החוצים את החתך.

$$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

זרימה בחתך: סכום הזרימות החוצות את החתך.

$$f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v)$$

למת החתך: תהי f זרימה ברשת ויהי (S,T) חתך, אז $|f| = f(S,T)$.

הוכחה:

$$f(S,T) = f(S,V) - f(S,S) = f(S,V) = f(s,V) + f(S-s,V) = f(s,V) = |f|$$

0 (שימור זרימה)

מסקנה $(\text{MaxFlow} \le \text{MinCut})$: $|f| \le c(S,T)$.

כלומר, הקיבול של חתך כלשהו ברשת מגביל את ערך הזרימה המקסימלית ובפרט זה נכון עבור חתך מינמלי ברשת.

הוכחה: לכל חתך (S,T) ולכל זרימה f מתקיים:

$$|f| = f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v) \le \sum_{u \in S} \sum_{v \in T} c(u,v) = c(S,T)$$

למת החתך

אילוץ הקיבול

לכן, גם הקיבול של החתך קיבולו מינימלי מגביל את הזרימה המקסימלית.

משפט Max-Flow-Min-Cut

תהי f זרימה ברשת G. התנאים הבאים שקולים:

1. f זרימה מירבית ב-G.
2. בגרף השיורי G_f אין מסלול מ-s ל-t (אין מסלול משפר ב-G).
3. קיים חתך (S,T) שעבורו $c(S,T) = |f|$.

הוכחה:

$1 => 2$: נניח בשלילה ש-f זרימה מקסימלית אבל ב-G_f יש מסלול משפר P, אז אפשר להגדיל את הזרימה f בעזרתו ולכן f איננה מקסימלית.

$2 => 3$: נניח שב-G_f אין מסלול מ-s ל-t ונגדיר:

$$S = \{v \in V \mid G_f - \text{ב } v - \text{ל } s - \text{יש מסלול מ}\}, \quad T = V \setminus S$$

- (S,T) חתך כי $s \in S$ וגם $t \notin S$ (כי אין מסלול מ-s ל-t ב-G_f).
- לכל $u \in S, v \in T$ מתקיים $f(u,v) = c(u,v)$, אחרת הקשת (u,v) שייכת ל-G_f ואז גם $v \in S$ וזו סתירה.

$$|f| = f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v) = \sum_{u \in S} \sum_{v \in T} c(u,v) = c(S,T)$$

$3 => 1$: לכל חתך (S,T) מתקיים: $|f| = f(S,T) \le c(S,T)$.

לכן, אם קיים חתך (S,T) שעבורו $|f| = c(S,T)$ אז f זרימה מקסימלית.

מסקנה: אם אלגוריתם Ford-Fulkerson עוצר, אז הזרימה |f| ברשת מירבית.

הוכחה:

אם האלגוריתם עוצר אז אין בגרף השיורי G_f מסלולים משפרים.

לכן, לפי משפט Max-Flow-Min-Cut, הזרימה ברשת מירבית.

יישום שיטת ה-Bit-Scaling של Gabow עבור Ford-Fulkerson

יהי $G = (V, E)$ גרף מכוון עם פונקציות קיבול $c : E \to N_{\geq 0}$, שני צמתים מיוחדים s ו-t שמייצגים את המקור והיעד.

נניח שהגרף הוא קשיר, מכאן ש - $|E| \geq |V| - 1$.

בהינתן שכל קיבול c(e) מיוצג בתור מספר בינארי בעל K סיביות. (עם אפסים מובילים במידת הצורך על מנת שכל הקיבולים יהיו בעלי אותו K סיביות)

נגדיר סדרה של בעיות $P_1, P_2, P_3, \ldots, P_k \ldots, P_K$, כאשר P_k מייצג את בעיית הזרימה המקסימלית באותה רשת זרימה $G = (V, E)$ והקיבולים הקטועים $c_k(e)$ בעלי k ספרות.

כלומר, לכל קשת, הקיבול שלה $c_k(e)$ הוא המספר שמיוצג ע״י k הביטים המובילים של c(e).

יהי f_k^* הזרימה המקסימלית עבור בעיה P_k, לכל $1 \leq k \leq K$.

לשם אתחול, נגדיר גם f_0^* בתור הזרימה האפס, כלומר $\forall e \in E : |f_0^*(e)| = 0$.

בנוסף, נגדיר $\forall e \in E : c_0(e) = 0$.

<u>האלגוריתם</u>

1. מציאת מספר הסיביות המינימלי K הדרוש לייצוג כל הקיבולים כמספרים בינאריים.

זהו גם מספר הסיביות בקיבול המקסימלי מבין הקיבולים על הקשתות בגרף.

2. איפוס הקיבולים $c_0(e)$ ואיפוס ערך הזרימה $f_0(e)$ על הקשתות בגרף.
3. החל מ-k=1 ועד K סיביות : (כל עוד נותרו עוד סיביות "לגלות")

3.1. מעבר על כל הקשתות בגרף:

גישה לספרה במקום של הסיבית ה-k מההתחלה בקיבול המקורי של הקשת, כלומר במרחק k-1 ספרות מהסיבית המשמעותית ביותר וחישוב הקיבול החדש על הקשת:

$$c_k(e) = 2 \cdot c_{k-1}(e) + \text{bit}_k(e)$$

חישוב הזרימה החדשה על הקשת:

$$|f_k(e)| = 2 \cdot |f^*_{k-1}(e)|$$

3.2. הפעלת אלגוריתם Ford-Fulkerson על רשת הזרימה החדשה עם הזרימה המעודכנת למציאת הזרימה המקסימלית f^*_k עבור הבעיה P_k.

הערה: אתחול הזרימה על כל קשת הוא הזרימה המעודכנת ושלב השיפור ייעשה על רשת הזרימה עם הזרימות המעודכנות.

4. החזרת ערך הזרימה המקסימלי שהתקבל מההרצה של האלגוריתם באיטרציה האחרונה, כלומר על רשת הזרימה עם הקיבולים בעלי K סיביות.

__הוכחת נכונות__

1. טענה: ניתן לחשב את מספר הסיביות המינימלי הדרוש לייצוג כל הקיבולים כמספרים בינאריים.

הוכחה: מס׳ הסיביות הדרוש לייצוג מס׳ כלשהו n הוא $\lfloor \log_2 n \rfloor + 1$, ניעזר בעובדה זו על מנת למצוא את המקסימום מבין כמויות הסיביות שהתקבלו לכל הקיבולים.

2. טענה: בהנחה ש-${f_{k-1}}^*$ זרימה מירבית חוקית עבור בעיה P_{k-1}, מתקיים כי הזרימה $2{f_{k-1}}^*$ חוקית לשימוש עבור בעיה P_k.

הוכחה: בהתחלת האיטרציה הראשונה $\forall e \in E : c_0(e) = 0$ ולכן $|f_0^*| = |f_0|$.

לפי ההנחה, עבור בעיה P_{k-1}, הזרימה על קשת הייתה חוקית עבור הקיבול עליה.

לכל קשת, הקיבול $c_{k-1}(e)$ גדל פי 2 ויותר (תלוי בסיבית מס׳ k בקיבול המקורי על הקשת) וערך הזרימה ${f_{k-1}}^*(e)$ גדל רק פי 2.

מכאן מתקיים: $|f_k(e)| = 2 * |{f_{k-1}}^*(e)| \leq 2 * c_{k-1}(e) + bit_k(e) = c_k(e)$

אין הפרה של אילוץ הקיבול גם עבור הזרימה החדשה $2 * {f_{k-1}}^*(e)$ ולכן היא חוקית לשימוש עבור בעיה P_k.

3. טענה: התוצאה של בעיה P_K היא אכן הזרימה המירבית עבור רשת הזרימה.

הוכחה: $c_K(e) = 2 * c_{K-1}(e) + bit_K(e)$ ו- $|f_K(e)| = 2 * |f_{K-1}{}^*(e)|$,

כאשר $c_K(e)$ הוא הקיבול של קשת עם התייחסות ל-K סיביות ו-$f_K(e)$ היא הזרימה המירבית על הקשת עם התייחסות ל-K סיביות.

לכן מתקיים $c_K(e) = c(e)$ ו- $|f_k(e)| = |f(e)|$, הקיבול והזרימה המקוריים על הקשת.

נפעיל את אלגוריתם פורד-פולקרסון על הרשת עבור הבעיה P_K (שהיא גם הרשת המקורית) ונקבל את ערך הזרימה המירבית $|f_K{}^*|$ עבור רשת הזרימה.

4. טענה: הזרימה המירבית המקסימלית בגרף הבעיה P_1 היא מס׳ הצמתים m. (בסיס)

הוכחה: הזרימה המירבית בגרף הבעיה P_0 היא 0 כי הקיבולים על כל הקשתות הם 0.

בגרף הבעיה P_1, כאשר יש התייחסות לביט אחד בלבד עבור כל קיבול ברשת, הביט הראשון בכל אחד מהקיבולים הוא מקסימום 1 ולכן הזרימה המירבית שיכולה לצאת מהמקור היא מקסימום מס׳ הקשתות שיכולות לצאת ממנו O(m).

5. טענה: $|f_k{}^*| - 2 * |f_{k-1}^*| \leq m$. (צעד)

הוכחה:

עבור הקשתות היוצאות מהמקור s:

$$|f_{k-1}^*(e)| = c_{k-1}(e)$$

$$c_k(e) = 2 * c_{k-1}(e) + bit_k(e) = 2 * |f_{k-1}^*(e)| + bit_k(e) \leq 2 * |f_{k-1}^*(e)| + 1$$

$$|f_k^*(e)| \leq c_k(e) = 2 * |f_{k-1}^*(e)| + 1$$

מס׳ הקשתות שיוצאות מ-s לא עולה על מס׳ הקשתות m בגרף ולכן:

$$|f_k{}^*| \leq 2 * |f_{k-1}^*| + m \qquad => \qquad |f_k{}^*| - 2 * |f_{k-1}^*| \leq m$$

6. טענה: מציאת מסלולים קצרים ביותר עבור בעיה P_k, שמתחילה עם $2f_{k-1}{}^*$ כפתרון התחלתי, רצה בזמן O(nm).

הוכחה: בכל איטרציה של הפעלת פורד-פולקרסון על רשת הזרימה של בעיה P_k, השיפור המקסימלי בזרימה על כל קשת הוא 1 ולכן השיפור המקסימלי בזרימה הכוללת הוא מס' הקשתות שיוצאות מהמקור s.

אנו ידועים שמס' הקשתות המקסימלי שיכולות לצאת מ-s לשאר הצמתים ברשת יהיה $|V| - 1$ ולכן מס' המסלולים המשפרים המקסימלי שניתן למצוא בגרף השיורי עבור רשת זרימה זו שווה ל-O(n).

זה גם מס' האיטרציות להרצת אלגוריתם BFS.

מכאן, אם נשתמש באלגוריתם BFS למציאת מסלול בזמן O(n + m) = O(m),

($|E| \geq |V| - 1$), מציאת כל המסלולים עבור בעיה P_k רצה בזמן O(nm).

<u>סיבוכיות זמן ריצה</u>

1. מעבר על כל הקיבולים שעל הקשתות O(m) ומציאת מס' סיביות לכל קשת O(1). סה"כ O(m). נובע מטענה 1.
2. איפוס קיבולים וזרימה על הקשתות O(m).
3. מס' האיטרציות כמס' הסיביות O(K), בכל איטרציה:
   3.1. מעבר על הקשתות וגישה לספרה O(m).
   3.2. הפעלת פורד-פולקרסון O(nm). נובע מטענה 6.

   סה"כ O(K) * (O(m) + O(nm)) = O(nmK).
4. החזרת ערך זרימה מקסימלי O(1).

סה"כ: O(mK) + O(m) + O(nmK) + O(1) = O(nmK).

השערת המחקר

אנו משערים שבאמצעות שיטת ה-Bit-Scaling של Gabow ניצור אלגוריתם למציאת ערך זרימה מירבית שיעילותו תלויה בכמות הסיביות בקיבול הכי גבוה בגרף, ובכך אם פונקציית הקיבול בקלט תהיה עם קיבול מקסימלי עם כמות סיביות נמוכה מספיק ביחס לכמות הקשתות, נוכל לשפר את יעילות ריצת האלגוריתם המקורי של Ford-Fulkerson.

הכנות לניסוי Bit-Scaling

- נשתמש בתוכנה Eclipse ובשפה Java, ונשתמש בספריה graphstream (ראה מקור 5) שמאפשרת לנו:

• יצירת אובייקטים כגון: גרף, קודקוד, קשת.

• יצירת גרף מכוון רנדומלי בעל n צמתים.

• הוספת תכונות לאובייקטים, בעיקר לשמירת קיבול וזרימה.

- נכתוב את הפונקציות העיקריות הבאות:

• getMaxFlow – פונקציה שמקבלת כקלט רשת זרימה ומחזירה את ערך הזרימה המירבי שיכול לעבור בה מהמקור לבור.

• initAdj – פונקציה שמקבלת כקלט גרף ושומרת לכל צומת את רשימת השכנים שלה.

• BFS – פונקציה שמקבלת כקלט גרף, משתמשת בתוצרי הפונקציה initAdj ומחזירה לכל צומת את המסלול הכי קצר מהמקור אליה ואת אורכו.

• getPath – פונקציה שמקבלת כקלט גרף ומספר צומת בצורה רקורסיבית, משתמשת בתוצרי הפונקציה BFS ומחזירה את רשימת הקשתות במסלול מהמקור לצומת. נשתמש בה כדי למצוא את המסלול מהמקור לבור בגרף השיורי.

• getRequiredBits – פונקציה שמקבלת אוסף ערכי קיבולים אי-שליליים ומחזירה את מספר הסיביות בקיבול המקסימלי באוסף.

מהלך ניסוי Bit-Scaling

השוואת זמני ריצה עבור קלטים זהים:

נריץ את שני האלגוריתמים על גרפים בגדלים שונים, לפי מקור שיתחבר לגרף ובור שהגרף יחובר אליו עם קיבולים אי-שליליים.

נשווה בין זמני הריצה של התוצאות עבור כל אחד מהם כאשר יש חשיבות לחסם עליון וחסם תחתון על הקיבול המקסימלי ברשת הזרימה.

חסם עליון – נבחר את מספר הקשתות $|E|$.

חסם תחתון – נבחר את $\log_{10}|E|$.

תוצאות ניסוי Bit-Scaling

500 צמתים ו-2646 קשתות:

Number of Bits = 2
Bit-Scaling running time : 0.3004179 seconds.
Ford-Fulkerson running time : 0.3249575 seconds.

Number of Bits = 4
Bit-Scaling running time : 0.3505846 seconds.
Ford-Fulkerson running time : 0.3710478 seconds.

Number of Bits = 5
Bit-Scaling running time : 0.4826958 seconds.
Ford-Fulkerson running time : 0.4275806 seconds.

Number of Bits = 7
Bit-Scaling running time : 0.5106278 seconds.
Ford-Fulkerson running time : 0.3260123 seconds.

Number of Bits = 11
Bit-Scaling running time : 0.9707791 seconds.
Ford-Fulkerson running time : 0.4112960 seconds.

1000 צמתים ו-5312 קשתות :

Number of Bits = 2
Bit-Scaling running time : 1.5677564 seconds.
Ford-Fulkerson running time : 1.9191646 seconds.

Number of Bits = 4
Bit-Scaling running time : 2.4520959 seconds.
Ford-Fulkerson running time : 2.4975598 seconds.

Number of Bits = 5
Bit-Scaling running time : 2.3725291 seconds.
Ford-Fulkerson running time : 2.2697169 seconds.

Number of Bits = 7
Bit-Scaling running time : 3.2206728 seconds.
Ford-Fulkerson running time : 2.5177996 seconds.

Number of Bits = 13
Bit-Scaling running time : 7.6433509 seconds.
Ford-Fulkerson running time : 2.5203365 seconds.

2000 צמתים ו-10646 קשתות :

Number of Bits = 3
Bit-Scaling running time : 12.1556708 seconds.
Ford-Fulkerson running time : 14.3753815 seconds.

Number of Bits = 4
Bit-Scaling running time : 17.4734826 seconds.
Ford-Fulkerson running time : 19.0287221 seconds.

Number of Bits = 5
Bit-Scaling running time : 21.5961446 seconds.
Ford-Fulkerson running time : 20.4344744 seconds.

Number of Bits = 7
Bit-Scaling running time : 28.3133797 seconds.
Ford-Fulkerson running time : 20.4508253 seconds.

Number of Bits = 14
Bit-Scaling running time : 72.2035783 seconds.
Ford-Fulkerson running time : 21.4795002 seconds.

5000 צמתים ו-26646 קשתות :

Number of Bits = 3
Bit-Scaling running time : 227.7628831 seconds.
Ford-Fulkerson running time : 281.125669 seconds.

Number of Bits = 5
Bit-Scaling running time : 302.7419346 seconds.
Ford-Fulkerson running time : 340.510598 seconds.

Number of Bits = 6
Bit-Scaling running time : 376.6639475 seconds.
Ford-Fulkerson running time : 353.959483 seconds.

Number of Bits = 7
Bit-Scaling running time : 541.003154 seconds.
Ford-Fulkerson running time : 412.5484657 seconds.

Number of Bits = 15
Bit-Scaling running time : 1382.2356035 seconds.
Ford-Fulkerson running time : 536.3868125 seconds.

הערה : לעתים התוצאות בין שני האלגוריתמים יהיו שונים עם שגיאה יחסית נמוכה ביניהם, זאת מכיוון שסביבת Java מעגלת מספרים מסוימים ואין לנו שליטה על כך.

אנו יודעים שהאלגוריתמים אכן נכונים מהוכחת הנכונות שלהם.

מסקנות Bit-Scaling

עבור הגרף עם 500 הצמתים :

ניתן לראות שכאשר החסם העליון על הקיבולים הוא נמוך, יש עדיפות בזמן ריצה לשימוש בשיטת Bit-Scaling.

בנוסף, עבור חסם עליון גבוה יותר, ככל שהקיבול המקסימלי ברשת הזרימה גדל, כך גם גדל הפער בין זמני הריצה של האלגוריתמים וניתן לראות כי יש עדיפות לשימוש באלגוריתם המקורי. זאת מכיוון שזמן הריצה תלוי בעיקר ביחס שבין הקיבול המקסימלי ברשת לבין מספר הקשתות בה.

עבור הגרף עם 1000 הצמתים :

ניתן לראות שכאשר החסם העליון על הקיבולים הוא נמוך, יש עדיפות בזמן ריצה לשימוש בשיטת Bit-Scaling ושהפער בין זמני הריצה גדל מסדר גודל של מאית השנייה בגרף של 500 הצמתים לסדר גודל של עשירית השנייה בגרף זה.

בנוסף, ניתן לראות כי הפער בין זמני הריצה הוא בסדר גודל של שנייה כשמסתכלים על החסם העליון הגבוה ביותר מהפער שהתקבל בגרף הקודם עם 500 הצמתים שהוא בסדר גודל של עשירית השנייה ושיש עדיפות לשימוש באלגוריתם המקורי עבור חסם עליון זה.

עבור הגרף עם 2000 הצמתים :

גם כאן, יש עדיפות בזמן ריצה לשימוש בשיטת ה-Bit-Scaling אפילו יותר גבוהה מכיוון שהפער בין זמני הריצה גדל מסדר גודל של עשירית השנייה בגרף הקודם לסדר גודל של שנייה בגרף זה.

בנוסף, ניתן לראות כי הפער בין זמני הריצה הוא בסדר גודל של עשרות שניות כשמסתכלים על החסם העליון הגבוה ביותר מהפער שהתקבל בגרף הקודם שהוא בסדר גודל של שנייה ושיש עדיפות לשימוש באלגוריתם המקורי עבור חסם עליון זה.

עבור הגרף עם 5000 הצמתים :

גם כאן, יש עדיפות בזמן ריצה לשימוש בשיטת ה-Bit-Scaling אפילו יותר גבוהה מכיוון שהפער בין זמני הריצה גדל מסדר גודל של שנייה בגרף הקודם לסדר גודל של עשרות שניות בגרף זה.

בנוסף, ניתן לראות כי הפער בין זמני הריצה הוא בסדר גודל של מאות שניות כשמסתכלים על החסם העליון הגבוה ביותר מהפער שהתקבל בגרף הקודם שהוא בסדר גודל של עשרות שניות ושיש עדיפות לשימוש באלגוריתם המקורי עבור חסם עליון זה.

לסיכום :

האלגוריתם שמשתמש בשיטת ה-Bit-Scaling הצליח לשפר משמעותית את זמני הריצה, רק כאשר הקיבול המקסימלי בגרף היה נמוך מספיק ביחס למספר הקשתות בו. השערתנו הייתה נכונה.

בעיית שיבוץ משימות במעבד

נתונה רשימת משימות J ומכונה. לכל משימה $j \in J$ יש זמן שחרור r_j וזמן יעד d_j ולוקח בדיוק שעה אחת לעבד משימה.

אנחנו מניחים שזמני שחרור וזמני יעד הם תמיד התחלות של שעות, כמו 7AM, 5PM ועוד.

נניח שלמכונה יש קיבולי עיבוד c חיוביים שונים בשעות שונות ביום, למשל

$c(3AM \to 4AM) = 8$ משמעותו שבזמן שבין 3AM ו-4AM, המכונה יכולה לעבד עד 8 משימות.

עלינו למצוא תשובה לשאלה : האם יש לוח זמנים אפשרי לעיבוד כל המשימות ב-J?

נתכנן אלגוריתם שרץ בזמן פולינומיאלי, נוכיח נכונות ונממש ונשווה ביצועים של 2 האלגוריתמים.

הגדרה

עבור גרף G, שידוך M הינה קבוצת קשתות זרות בקודקודים.

גודל השידוך הינו מספר הקשתות בקבוצה זו.

שידוך מקסימלי הוא שידוך שגודלו מקסימלי מבין כל השידוכים האפשריים.

הקדמה

לכל משימה נתאים קודקוד בקבוצת קודקודים שתיקרא J.

לכל שעה ביום נגדיר קודקוד בקבוצת קודקודים שתיקרא H.

נבנה גרף דו-צדדי G כאשר בצד אחד קודקודים מקבוצה J ובצד השני קודקודים מקבוצה H.

מכל משימה בקבוצה J, נמתח קשתות לשעות בה המעבד יכול להריץ אותה בקבוצה H.

השידוך M בגרף G הוא קבוצת הקשתות הזרות בקודקודים היוצאות מJ ונכנסות לH.

<u>האלגוריתם</u>

1. עבור הגרף הדו-צדדי $G = (J,H,E)$, כאשר $H = \{0,..,23\}$ התחלות של שעות ביום ו-$E = \emptyset$, נגדיר את רשת הזרימה $N = (G,s,t,c)$ באופן הבא:

$$V = J \cup H \cup \{s, t\}$$

$$E = \{ (s,v) \mid v \in J \} \cup \{ (u,t) \mid u \in H \} \cup \{ (u,v) \mid u \in J, v \in H, r_j \le v < d_j \}$$

לכל קשת נגדיר קיבול באופן הבא:

$$\forall j \in J: \quad c(s,j) = 1$$

$$\forall h \in H: \ c(h,t) = c(h \to h+1)$$

$$\forall j \in J: \quad c(j,h) = 1 \mid r_j \le h < d_j$$

2. הפעלת אלגוריתם פורד-פולקרסון.
3. נגדיר $M = \{ (u,v) \mid u \in J,\ \ v \in H,\ \ f(u,v) = 1 \}$ שידוך מקסימלי ב-G.
   אם $|M| = |J|$, יוחזר true, אחרת false.

<u>סיבוכיות זמן ריצה</u>

1. בניית רשת זרימה – $O(n+m)$,
   כאשר גודל שידוך מקסימלי הוא לכל היותר $O(n) = O(|V|) = |J| + 24 + 2$.
2. לכן יהיה צורך בהפעלת $O(n)$ איטרציות באלגוריתם Ford-Fulkerson, כי בכל איטרציה משפרים את הזרימה ב- 1 לפחות. (כל פעם מהמקור s ל-$j \in J$ אחר)
   אם מוצאים מסלולים משפרים ע"י BFS, סיבוכיות כל איטרציה שלו היא $O(m)$.

מכאן שהסיבוכיות הכוללת היא $O(n+m) + O(nm) = O(nm)$.

הוכחת נכונות

1. טענה: השידוך M בגרף G מתאים לשיבוץ משימות חוקי במעבד.

הוכחה: אנו יודעים כי הזרימה מכל שעה $v \in H$ לבור t שווה לקיבול המעבד הנתון עבור אותה שעה. מחוק שימור הזרימה ניתן לראות כי כל קודקוד שעה $v \in H$ משודך למס' צמתי משימות $u \in J$ כגודל זרימה זה.

בנוסף, לכל משימה $u \in J$ יכולה לזרום זרימה של 0 או 1 מהמקור s. מחוק שימור הזרימה ניתן לראות כי כל משימה תתואם לשעת מעבד כגודל זרימה זה.

מכאן שלכל שעה יש התאמה למס' המשימות שהמעבד יכול לבצע בה ולכל משימה יש התאמה לשעת המעבד בה המעבד יבצע אותה ולכן השידוך מתאים למערכת המכונה והמשימות שלנו.

הערה: הקשתות בשידוך זרות בקודקודים על מנת שההתאמה תהיה רק בין משימות לשעות מעבד ולא התאמה של משימות למשימות או שעות מעבד לשעות מעבד.

2. טענה: יהי G גרף דו-צדדי ו-N רשת זרימה מתאימה (כפי שבנינו).

אז אם M שידוך ב-G קיימת ב-N זרימה בשלמים f כך ש-$|M| = |f|$.

ולהיפך: אם f זרימה בשלמים ב-M, אז קיים ב-G שידוך M כך ש-$|f| = |M|$.

הוכחה:

<=: נניח ש-M שידוך חוקי ב-G (קבוצה כלשהי של קשתות זרות ב-G), אז נגדיר ב-N פונקציית זרימה בשלמים באופן הבא:

אם $v \in H$,$u \in J$ | $(u,v) \in M$, אז $f\big((s,u)\big) = f\big((u,v)\big) = f\big((v,t)\big) = 1$.

הזרימה על שאר הקשתות היא 0. נראה כי זוהי זרימה חוקית:

- נשמרים אילוצי הקיבול כי $\forall e \in E: c(e) \geq 1$.

- מתקיים שימור הזרימה (לכל הצמתים השונים מ-s,t), כי לכל צומת משודך $u \in J$ ערך הזרימה הנכנסת = ערך הזרימה היוצאת = 1 וכנ"ל לכל $v \in H$.

לכל יתר הצמתים, לא יוצאת או נכנסת אליהם זרימה ולכן $|M| = |f|$.

$<=$: בהינתן פונקציית זרימה בשלמים f ב-N, נגדיר שידוך M ב-G באופן הבא:

$$M = \{ (u,v) \mid u \in J, v \in H , f(u,v) = 1 \}$$

לכל צומת $u \in J$ יש בדיוק קשת נכנסת אחת שקיבולה 1.

סך הזרימה הנכנסת ל-u היא לכל היותר 1. מתקיים חוק שימור הזרימה ולכן סך כל הזרימה היוצאת מ-u היא גם לכל היותר 1.

מכיוון ש-f היא זרימה בשלמים, הזרימה היא בדיוק 1 ולכן קיים צומת יחיד $v \in H$ כך שהזרימה על הקשת (u,v) שווה ל-1. (כפי שנתון בתרגיל כל משימה צריכה שעת מעבד אחת)

מכאן נובע ש-$u \in J$ משודך בדיוק לצומת אחד ב-H.

מסקנה: בפרט זרימה מקסימלית מתאימה לשידוך מקסימלי.

3. טענה: הזרימה המירבית ברשת הזרימה שבנינו שווה ל-$|J|$ אם ורק אם קיים לוח זמנים חוקי שבו לכל משימה יש זמן מעבד מתאים במערכת.

הוכחה:

$=>$: נניח קיים לוח זמנים חוקי, אז לכל משימה מוקצית שעת מעבד אחת ו-$|J| = |M|$.

כאשר $|M|$ גודל השידוך בין משימות לשעות מעבד.

נסתכל על החתך שבצידו האחד המקור s ובצידו השני קבוצת המשימות J.

מחוק שימור הזרימה, הזרימה החוצה אותו היא בדיוק גודל השידוך, כלומר ההתאמה שנעשתה בשידוך M בין כל משימה לשעת מעבד ולכן $|M| = |f|$ ו- $|J| = |M| = |f|$.

=> : נניח כי $|J| = |f|$, נסתכל על החתך שבצידו האחד המקור s ובצידו השני קבוצת המשימות J. מחוק שימור הזרימה, הזרימה החוצה אותו היא בדיוק גודל השידוך, כלומר ההתאמה שנעשתה בשידוך M בין כל משימה לשעת מעבד ולכן $|f| = |M|$. לכן $|J| = |M|$, כלומר לכל משימה יש בדיוק שעת מעבד אחת שמתאימה לה ויש לוח זמנים חוקי למערכת הנתונה.

ציפיות לניסוי שיבוץ

אנו מצפים שאלגוריתם פורד-פולקרסון ללא שיטת Bit-Scaling יתאים יותר לבעיה מכיוון

שזמן הריצה שלו יהיה נמוך יותר מזה שמשתמש בשיטת Bit-Scaling.

השערה זו נובעת מהעובדה שלכל משימה יש צורך בשעת מעבד אחת כדי לבצעה וכל הקשתות היוצאות מהמקור s יהיו בעלות קיבול 1, מכאן ששימוש ב-Bit-Scaling יגרור בדיקות לא נחוצות עבור אפסים מובילים של קיבולים אלו, שלא יביאו להתקדמות בפתרון ובמציאת ערך הזרימה המירבי ברשת הזרימה שלנו.

מהלך ניסוי שיבוץ

השוואת זמני ריצה עבור קלט זהה:

נריץ את שני האלגוריתמים על רשת הזרימה המבוססת על גרף G (כמתואר בעמוד 24) עם 30 משימות כאשר זמן התחלתן וזמן סיומן יוגרל לכל אחת.

נשווה בין זמני הריצה של התוצאות עבור כל אחד מהם כאשר יש חשיבות לחסם העליון על הקיבול המקסימלי ברשת הזרימה.

חסם עליון – נבחר את $\log_e|J|$.

תוצאות ניסוי שיבוץ

The flow graph has 56 nodes and 522 edges
Number of Bits = 3
Bit-Scaling running time : 0.028974 seconds.
Ford-Fulkerson running time : 0.0080431 seconds.
Flow = 29
false

The flow graph has 56 nodes and 584 edges
Number of Bits = 3
Bit-Scaling running time : 0.0297737 seconds.
Ford-Fulkerson running time : 0.0093077 seconds.
Flow = 30
true

מסקנות שיבוץ

בבעיית שיבוץ משימות למעבד, מכיוון שכל משימה צריכה שעת מעבד אחד, ערך הקיבול על הקשתות החוצות מצמתי המשימות לצמתי השעות יהיה 1.

לכן, רוב הקשתות בגרף G יהיו בעלות ערך קיבול 1 ויהיו בגודל סיבית אחת בלבד.

מכיוון שמצומת המקור s יצאו קשתות לכל המשימות ועל כל קשת יהיה ערך קיבול 1, מספר האיטרציות שאלגוריתם פורד-פולקרסון יעשה יהיה |J| כמספר המשימות.

לעומת זאת, מספר האיטרציות שהאלגוריתם פורד-פולקרסון בשיטת Bit-Scaling יעשה יהיה |J|*1+1*(K-1) (כאשר K הוא מספר הביטים בקיבול המקסימלי בגרף G) מכיוון שבמקרה הגרוע מוסיפים K-1 אפסים מובילים לכל הקיבולים עם ערך 1 בגרף ומכאן שב-K-1 האיטרציות הראשונות של הפעלת פורד-פולקרסון, האלגוריתם לא יעשה דבר אלא יבזבז זמן בחיפוש מסלולים בגרפים השיוריים המתאימים לכל איטרציה, ורק באיטרציה האחרונה K הוא יבצע את |J| השיפורים בגרף G.

מכאן שנעדיף להשתמש בפורד-פולקרסון ללא שיטת Bit-Scaling עבור הבעיה הספציפית הנתונה.

מקורות

1. אורט בראודה. מצגת בנושא רשתות זרימה במסגרת קורס אלגוריתמים.
2. טכניון. מצגת בנושא ערך זרימה מירבי במסגרת קורס אלגוריתמים 1.
3. [10] H.N Gabow. Scaling Algorithms for Network Problems. J. of Comp. and Sys. Sci., 31: 148-168, 1985.
4. https://brilliant.org/wiki/flow-network/
5. http://graphstream-project.org/