נושא 8 - מסלול קצר ביותר

א) הגדרת הבעיה

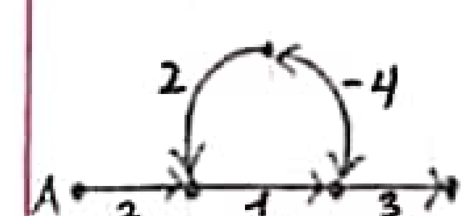
בהינתן גרף ממושקל $G=(V,E)$, כך שלכל קשת בגרף יש משקל המבטא את העלות של המעבר בקשת, וצומת u כלשהי בגרף, נרצה אלגוריתם שיחשב לנו מהו המסלול הקצר ביותר מצומת u לכל אחד מהצמתים האחרים בגרף, ומהו המשקל הכולל בכל אחד מהמסלולים. יש שני אלגוריתמים שעושים פעולות אלו: אלגוריתם דייקסטרה ואלגוריתם בלמן-פורד ועליהם נלמד בפרק זה. אך קודם נלמד כמה מושגים בסיסיים לצורך אלגוריתמים אלו.

ב) תכונות מסלול קצר ביותר

(1) מסלול קצר ביותר מורכב מתתי מסלולים קצרים ביותר. לדוגמא, המסלול הקצר ביותר מ-u ל-v העובר בצומת x, אזי מסלול זה מורכב מהמסלול הכי קצר מ-u ל-x והמסלול הכי קצר מ-x ל-v. ההוכחה לכך שאם היה מסלול קצר יותר מ-u ל-x או מ-x ל-u, יכולנו לבנות מסלול נוסף העובר בתת-מסלול הכי קצר ובכך לקבל מסלול קצר יותר מ-u ל-v. סתירה!

(2) אי-שוויון המשולש - נסמן $\delta(u,v)$ המסלול הקצר ביותר מ-u ל-v. עבור כל צומת בגרף $x\in V$ מתקיים: $\delta(u,v)\leq\delta(u,x)+\delta(x,v)$. זאת מפני ש-$\delta(u,v)$ לא יכול להיות ארוך יותר מכל מסלול אחר. דרך נוספת לתאר אי-שוויון המשולש: $\delta(u,v)\leq\delta(u,x)+\omega(x,v)$. כלומר $\delta(u,v)$ קטן מכל מסלול מ-u ל-x ועוד משקל הקשת מ-x ל-v.

(3) משקלים שליליים - בגרף שבו יש מעגל שהמשקל הכולל בו שלילי, חלק מהמסלולים הקצרים ביותר לא יהיו קיימים, משום שתמיד נוכל להיכנס למעגל שוב ושוב ובכך להוריד את המשקל הכולל של המסלול כך שיתקבל $-\infty$. במקרה שיש מעגל שלילי בגרף נחזיר "אין פתרון". ההבדל בין אלגוריתם דייקסטרה לבלמן-פורד, שדייקסטרה לא עובד בגרף עם משקלים שליליים (גם אם אין מעגל שלילי), ואילו בלמן-פורד כן.



ג) הרפיה - Relaxation

במהלך ריצת האלגוריתמים שנתאר אנו שומרים בכל צומת v אומדן $d[v]$ על הגבול העליון של משקל המסלול הכי קצר מ-u ל-v. אם מצאנו כי $d[v]>d[x]+\omega(x,v)$, אזי לפי תכונה 2 נעדכן את $d[v]=d[x]+\omega(x,v)$. פעולה זו נקראת "הרפיה".

24/12 בס"ד

33

3) אלגוריתם דייקסטרה

אלגוריתם זה: המשקלים צריך להיות אי-שליליים, אחרת אלגוריתם לא יזהה. צומת במשקל במעגלת.

<u>פעולת האלגוריתם:</u> באלגוריתם זה נעבור על כל צמתי הגרף, לפי סדר מסוים כמו שנסביר בהמשך. ועבור כל צומת נעבור על כל הקשתות היוצאות ממנו. בכל צומת v נתחזק אומדן d[v] שהוא הגבול העליון של משקל המסלול הכי קצר מ-u ל-v. כמו כן נשמור עבור כל צומת מיהו הצומת שהביאה לגילויה במערך π[v]. כאשר במהלך המעבר על הגרף נגיע לצומת, נבדוק האם המסלול הנוכחי שהלכנו בו קטן מהאומדן של הצומת. אם כן נבצע הרפיה ונעדכן את π[v], ואם לא, נשאיר ללא שינוי.

(1) נבנה שתי קבוצות $Q=V$ ו- $S=\emptyset$. בכל איטרציה נשלוף מ-Q ונכניס ל-S. $O(n)$

(2) נעבור על כל הצמתים ונאתחל כל צומת $d[v]=\infty$, $\pi[v]=null$. $O(n)$

(3) נאתחל את הצומת הראשונה $d[u]=0$. $O(1)$

(4) נעבור בלולאת while כל עוד Q לא ריק. נשלוף את הצומת שהאומדן שלו מינימלי ונכניס ל-S. בפעם הראשונה הצומת שתישלף תהיה u. $O((m+n)$

(5) עבור כל צומת מינימלי, נעבור על כל הקשתות היוצאות ממנו אל צמתים שעדיין ב-Q. לכל צומת שנגיע נבדוק האם צריך לבצע הרפיה ולעדכן את π[v]. בסיום לולאת ה-while האלגוריתם ייפסק.

<u>הערה</u> - בכל פעם שאנו מוחקים צומת מ-Q, לא נוכל לעדכן את האומדן שלה. משמע מכך שהאומדן המחושב בצומת זו מתאים למסלול הקצר ביותר. לכן, אם אנו מחפשים מסלול ספציפי מצומת לצומת אחרת, נוכל בשלב זה לעצור את האלגוריתם.

<u>פסאודו - קוד:</u>

```
Dijkstra(G, u) {
    Q = MinHeap(V) , S = Ø
    for each v∈V {
        d[v] = ∞ , π[v] = null
    }
    d[u] = 0
    while (Q ≠ Ø) {
        x = ExtractMin(Q)
        S = S U {x}
        for each v∈adj[x] {
            if (d[v] > d[x] + w(x,v)) {      // Relaxation
                d[v] = d[x] + w(x,v)
                π[v] = x
} } } }
```

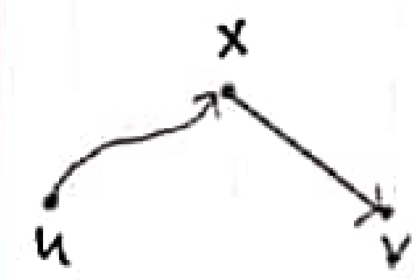
24/12

(34)

הוכחת נכונות האלגוריתם:

- טענת עזר - במהלך האלגוריתם בכל שלב שנגיע לצומת v, האומדן השמור בה $d[v]$ (שבהתחלה הוא ∞ והתעדכן במהלך האלגוריתם) תמיד גדול או שווה למרחק המינימלי בין u ל-v, שאותו האלגוריתם רוצה לחשב. כלומר מתקיים: $\delta(u,v) \le d[v]$.

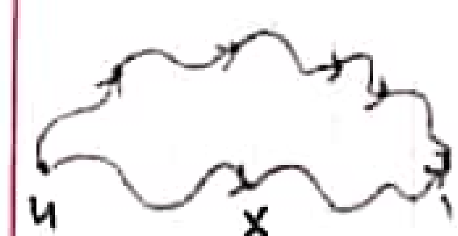
הוכחה - נניח בשלילה שהטענה לא נכונה, יהי v הצומת הראשונה שהתגלתה שבה $d[v] < \delta(u,v)$. יהי x הצומת שהובילה אל v. כלומר $\pi[v]=x$. אזי מתקיים $d[v]=d[x]+\omega(x,v)$. אמנם נשים לב כי גם מתקיים:



סתירה! שהרי יוצא: $\underbrace{d[v] < \delta(u,v)}_{\text{לפי הנחה בשלילה}} \underbrace{\le \delta(u,x) + \omega(x,v)}_{\text{לפי אי-שוויון המשולש}} \underbrace{\le d[x] + \omega(x,v)}_{\substack{\text{v הראשון שאינו} \\ \text{מקיים המשפט לכן} \\ d[x] \ge \delta(u,x)}} \Rightarrow$ ש- $d[v] < d[x] + \omega(x,v)$.

- הוכחת האלגוריתם - נוכיח שכאשר צומת כלשהו v מוסר מ-Q הוא מקבל את הערך של המסלול המינימלי מ-u אליו. כלומר $d[v]=\delta(u,v)$.

הוכחה - נניח בשלילה שהטענה לא נכונה, יהי v הצומת הראשונה שהוסרה ובה מתקיים $d[v] \ne \delta(u,v)$. לפי טענת עזר מתקיים $d[v] > \delta(u,v)$. כתוצאה מכך נוכל להסיק שהגענו ל-v לא דרך המסלול הכי קצר, וגם שיש לפחות צומת אחת במסלול הכי קצר מ-u ל-v שעוד לא הסרנו. שהרי, אם היינו מסירים את כל הצמתים במסלול האמיתי הכי קצר מ-u ל-v, היינו עושים הרפי"ה לכל אחת מהקשתות היוצאות מהם וכך בוודאי מגיעים ל-v במסלול הכי קצר. נסמן x הצומת הראשונה במסלול הכי קצר שלא הוסרה. מתקיים $d[x]=\delta(u,x)$, שהרי כל הצמתים לפני x הוסרו במסלול הכי קצר.



נשים לב כי מתקיים: $\underbrace{d[v] > \delta(u,v)}_{\text{לפי הנחה בשלילה}} \underbrace{= \delta(u,x) + \delta(x,v)}_{\text{הגדרת מסלול הכי קצר}} \underbrace{= d[x] + \delta(x,v)}_{\substack{\text{והסברנו מדוע} \\ d[x]=\delta(u,x)}} \ge d[x]$

הגענו לסתירה! כי יוצא $d[v] > d[x]$ ולכן לא היינו צריכים להסיר את v מהערימה אלא את x.

סיבוכיות האלגוריתם: כל הפעולות מחוץ ללולאת while מתבצעות ב-$O(n)$ או $O(1)$. בלולאת while עוברים על כל הצמתים והקשתות וגם שולפים את האיבר עם האומדן המינימלי. נחלק לארבעה מקרים: גרף דליל/צפוף ומימוש Q בערימה/מערך.

גרף דליל - גרף שבו מספר הקשתות לינארי למספר הצמתים $m=\Theta(n)$.

גרף צפוף - גרף שאינו דליל. מניחים $m=n^2$.

גרף \ מימוש	דליל	צפוף
ערימה	$O(n\log n)$	$O(n^2\log n)$
מערך	$O(n^2)$	$O(n^2)$

מסקנה - כאשר הגרף צפוף עדיף להשתמש במערך.

בגרף צפוף מימוש בעזרת ערימה כל עדכון של מפתח הוא ב- $\log(n)$. עבור כל צומת יש n קשתות ונעדכן כל קשת בכל עשיית הרפי"ה נקבל $O(n^2)$. במערך עדכון של $O(1)$ גם עדכון יש n צמתים נקבל $O(n^2)$.

31/12

בס"ד

(35)

ה) אלגוריתם בלמן - פורד

פעולת האלגוריתם: אלגוריתם בלמן-פורד בא לפתור את אותה הבעיה כמו אלגוריתם דייקסטרה, אך הוא גם מאפשר שבגרף יהיו משקלים שליליים בקשתות. אמנם כתוצאה מכך הזמן ריצה שלו פחות טוב. גם באלגוריתם זה עבור כל צומת v נתחזק אומדן $d[v]$ שהוא הגבול העליון של משקל המסלול הכי קצר מ-s ל-v. וכן נשמור מיהו הצומת שהביאה לגילוי במערך $\pi[v]$.

(1) נעבור על כל הצמתים ונאתחל כל צומת $d[v]=\infty$, ואת $\pi[v]=null$. — $O(n)$

(2) נאתחל את הצומת הראשונה $d[s]=0$. — $O(1)$

(3) עוברים בלולאה חיצונית $n-1$ פעמים. בכל איטרציה i עוברים על כל הקשתות בגרף, הסדר לא חשוב אך כן חשוב שהוא ישאר זהה בכל איטרציה. בכל קשת בודקים האם האומדן לצומת היעד גדול מהאומדן בצומת המקור פלוס משקל הקשת. אם כן - מבצעים הרפיה, ואם לא - לא משנים כלום. באיטרציה הראשונה נעדכן רק את הצמתים שהם השכנים של צומת המקור, ובכל איטרציה נגדיל את הטווח של הצמתים שאנו מעדכנים. בשיטה זו אנו בעצם עוברים על כל מסלול אפשרי בגרף היוצא מצומת המקור s, ומעדכנים את האומדנים בצמתים במסלול זה. מספיק $n-1$ איטרציות משום שהמסלול הפשוט הכי ארוך מ-s לכל צומת אחרת הוא לכל היותר $n-1$. בסיום הלולאה החיצונית יתקבלו המסלולים הכי קצרים לכל צומת. — $O(n \cdot m)$

(4) נעבור פעם נוספת על כל הקשתות ונבדוק אם צריך באחת מהן "הרפיה". אם כן סימן שיש בגרף מעגל שלילי ולכן נחזיר "אין פתרון". — $O(m)$

פסאודו - קוד:

```
BellmanFord (G, S) {
    for each v∈V {
        d[v]=∞ , π[v]=null
    }
    d[S]=0
    For (i=1 to n-1) {
        For each (u,v)∈E {
            IF ( d[v] > d[u] + w(u,v)) {        // Relaxation
                d[v]= d[u] + w(u,v)
                π[v] = u
            }
        }
    }
    For each (u,v) ∈ E {
        IF (d[v] > d[u] + w(u,v))  return "no Solution"
    }
}
```

31/12

הוכחת נכונות האלגוריתם:

נוכיח כי לאחר $n-1$ איטרציות האורך בכל צומת שווה לאורך המסלול הכי קצר.

כלומר מתקיים: $d[v]=\delta(s,v)$.

נניח שהמסלול הכי קצר מ-s ל-v הוא: $s\to v_1\to v_2\to\dots\to v$. נוכיח כי בכל איטרציה i מתקיים $d[v_i]=\delta(s,v_i)$, כלומר האלגוריתם יכניס ל-$d[v_i]$ את המסלול הקצר ביותר עד אליו. נוכיח באינדוקציה על i.

בסיס - עבור $i=0$, כלומר האיטרציה הראשונה, אכן $d[s]=0$ הוא המרחק הכי קצר.

הנחה - נניח שהטענה נכונה עבור k.

צעד - נוכיח עבור $k+1$. לפי הנחת האינדוקציה $d[v_k]=\delta(s,v_k)$ בתחילת האיטרציה ה-$k+1$.

לכן באיטרציה ה-$k+1$ כאשר נגיע לקשת (v_k,v_{k+1}) נבצע הרפיה על v_{k+1}, והוא אכן יחזיק את אורך המסלול המינימלי. $d[v_{k+1}]=\delta(s,v_{k+1})$.

מכיוון שהמסלול הקצר ביותר מ-s ל-v לא יכול להכיל יותר מ-$n-1$ צמתים (כי עבור מעגל שלילי שאז נחזיר "אין פתרון"), נקבל כי מספיק $n-1$ איטרציות כדי לעדכן את $d[v]$ להחזיק את אורך המסלול הקצר ביותר.

<u>סיבוכיות האלגוריתם</u>: כל הפעולות הם ב-$O(n)$ או ב-$O(1)$. בלולאה הראשונה עוברים $n\cdot m$ פעמים ובלולאה השנייה n פעמים. בתוך הלולאות כל הפעולות ב-$O(1)$. לכן סיבוכיות אלגוריתם בלמן-פורד היא $O(n\cdot m)$. ובמקרה הגרוע כאשר $m=n^2$ נקבל: $O(n^3)$.

1) סיכום מסלול קצר ביותר

לפי סוג המשקל של הגרף נחליט איזה אלגוריתם להשתמש למציאת מסלול הכי קצר.

סוג הגרף	אלגוריתם	זמן ריצה
ללא משקל	BFS	$O(m+n)$
משקל חיובי	Dijkstra	$O(n^2)$ צפוף / $O(n\log n)$ דליל
משקל כללי	Bellman-Ford	$O(m\cdot n)$

ב) מסלול קצר ביותר בגרף DAG

בגרף מכוון חסר מעגלים (DAG), נוכל למצוא דרך מהירה לחישוב מסלול קצר ביותר לכל צומת. במקום להשתמש באלגוריתם בלמן-פורד ולעבור $n-1$ פעמים על כל הקשתות, נוכל לעשות זאת במעבר אחד על כל הקשתות. וזאת אם נבצע קודם על הגרף מיון טופולוגי, כך שנעבור במעבר אחד על כל הקשתות לפי הסדר שיוצאים מהצמתים במיון טופולוגי, נעבור על כל המסלולים האפשריים, ולכן מספיק רק מעבר אחד על כל הקשתות. סיבוכיות האלגוריתם במקרה היא $O(m)\doteq O(m+n)$. מיון טופולוגי מתבצע גם כן ב-$O(m+n)$.