אלגוריתמים קומבינטוריים
סיכומים של תרגילי כיתה מסמסטרים קודמים בנושא

מרחקים בגרפים

חישוב המרחקים בין מצומת מסוים לשאר הצמתים:

1. נתון גרף (לא מכוון או מכוון) $G(V,E)$ עם פונקצית מרחק על הקשתות: $w : E \to \mathbf{R}$. נתון גם מקור $s \in V$ ורוצים לחשב את המרחק ממנו לכל צומת אחר. המרחק מ- s ל- v, שמסומן $\delta(s,v)$, הוא המינימום על כל המסלולים מ- s ל- v של סכום הרחקים על הקשתות בו. במקרה שאין מסלול מ- s ל- v נגדיר $\delta(s,v) = \infty$. במקרה שיש אפשרות לעבור על מסלול שלילי בדרך מ- s ל- v נגדיר $\delta(s,v) = -\infty$.

2. האלגוריתם של Dijkstra פותר את הבעיה בהנחה שהמרחקים על הקשתות (w) הם אי-שליליים. ניתן בקלות לתת אימפלימנטציה של $\text{DIJKSTRA}(G,w,s)$ שרצה בזמן $O(|V|^2)$. ניתן לקבל זמן ריצה $O(|E| + |V|\log|V|)$ על ידי שימוש במנה נתונים מתאים.

3. האלגוריתם של Bellman-Ford פותר את הבעיה בהנחה שאין מעגל שלילי בגרף. אם יש - הוא מגלה את המצב ומחזיר תשובה שנכשל. זמן הריצה הוא $O(|V||E|)$.

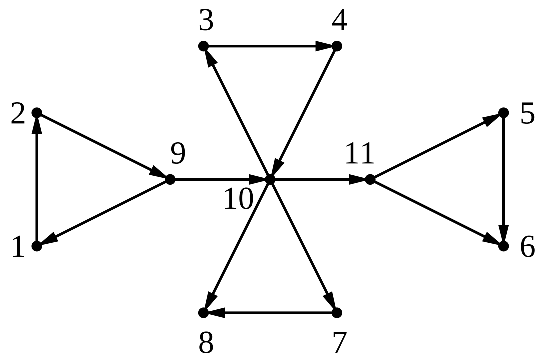
```
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex v ∈ V
     do d[v] ← ∞
        π[v] ← NIL
d[s] ← 0
```

```
RELAX(u, v, w)
if d[v] > d[u] + w(u, v)
     then d[v] ← d[u] + w(u, v)
          π[v] ← u
```

```
DIJKSTRA(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
S ← ∅
Q ← V[G]    ▷ Q is a priority queue with priorities d[v].
while Q ≠ ∅
     do u ← EXTRACT-MIN(Q)    ▷ Get a vertex from Q with minimum d[v].
        S ← S ∪ {u}
        for each v ∈ Adj[u]
             do RELAX(u, v, w)
```

```
BELLMAN-FORD(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
for i ← 1 to |V| − 1
     do for each edge (u, v) ∈ E
          do RELAX(u, v, w)
for each edge (u, v) ∈ E
     if d[v] > d[u] + w(u, v)
          then return FALSE
return TRUE
```

4. נריץ את האלגוריתמים על הגרף הבא:

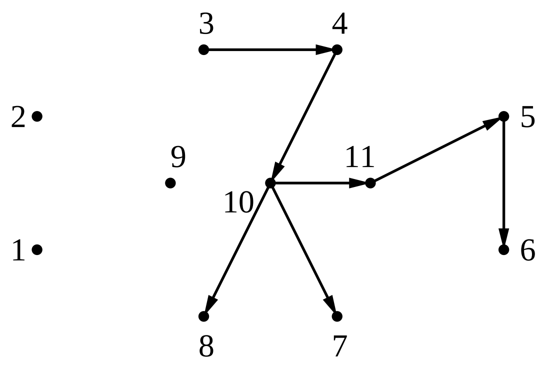


5. נריץ את DIJKSTRA$(G,3,w)$ אם המשקלות הבאות. (נניח כי $Adj[u]$ מסודר בסדר עולה.)

בטבלה השניה רשומים הערכים $d[v](\pi(v))$ במשך האלגוריתם, כאשר $\pi(v)=\mathrm{NIL}$ בכל מקום שהוא לא רשום. רשמנו רק את העידכונים בכל שורה. שורות ללא שינויים לוקטו ביחד. לאחר הטבלאות מופיע העץ המושרה.

e	$w(e)$	e	$w(e)$	e	$w(e)$	e	$w(e)$
$(1,2)$	5	$(3,4)$	4	$(7,8)$	3	$(5,6)$	2
$(2,9)$	2	$(4,10)$	3	$(10,7)$	5	$(11,5)$	2
$(9,1)$	2	$(10,3)$	1	$(10,8)$	4	$(11,6)$	4
$(9,10)$	1					$(10,11)$	1

Stage v	1	2	3	4	5	6	7	8	9	10	11
Init	∞	∞	0	∞	∞	∞	∞	∞	∞	∞	∞
$u=3$				$4(3)$							
$u=4$										$7(4)$	
$u=10$							$12(10)$	$11(10)$			$9(10)$
$u=11$					$11(11)$	$13(11)$					
$u=5$						$12(5)$					
$u=8,6,7,1,2,9$											
Final	∞	∞	0	$4(3)$	$11(11)$	$12(5)$	$12(10)$	$11(10)$	∞	$7(4)$	$9(10)$

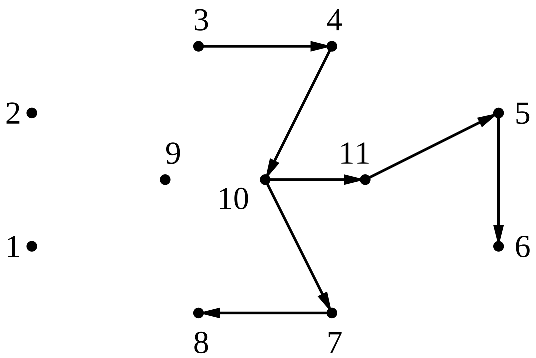


6. נריץ את BELLMAN-FORD$(G,3,w)$ אם המשקלות הבאות. (נניח שעוברים על הקשתות לפי הו-פעתם בטבלה, קודם כל מהעמודה הראשונה מלמעלה ללמטה וכו'.)

בטבלה השניה רשומים הערכים $d[v](\pi(v))$ במשך האלגוריתם, כאשר $\pi(v)=\mathrm{NIL}$ בכל מקום שהוא לא רשום. רשמנו רק שבהם יש עידכונות ואז רק את העידכון. לאחר הטבלאות מופיע העץ המושרה. שים לב לזאת שהאלגוריתם מצליח (מחזיר TRUE) אף על פי שיש מעגל שלילי בגרף! למה?

e	$w(e)$	e	$w(e)$	e	$w(e)$	e	$w(e)$
$(1,2)$	-5	$(3,4)$	4	$(7,8)$	-3	$(5,6)$	-2
$(2,9)$	2	$(4,10)$	-3	$(10,7)$	5	$(11,5)$	4
$(9,1)$	2	$(10,3)$	1	$(10,8)$	4	$(11,6)$	4
$(9,10)$	1					$(10,11)$	-2

Stage v	1	2	3	4	5	6	7	8	9	10	11
Init	∞	∞	0	∞	∞	∞	∞	∞	∞	∞	∞
$i=1, e=(3,4)$				$4(3)$							
$i=1, e=(4,10)$										$1(4)$	
$i=1, e=(10,7)$							$6(10)$				
$i=1, e=(10,8)$								$5(10)$			
$i=1, e=(10,11)$											$-1(10)$
$i=2, e=(7,8)$								$3(7)$			
$i=2, e=(11,5)$					$3(11)$						
$i=2, e=(11,6)$						$3(11)$					
$i=3, e=(5,6)$						$1(5)$					
Final	∞	∞	0	$4(3)$	$3(11)$	$1(5)$	$6(10)$	$3(7)$	∞	$7(4)$	$-1(10)$



7. תרגיל: נתון גרף מכוון $G(V,E)$ ולכל צלע מכוונת (u,v) נתון ערך $0 \le r(u,v) \le 1$ המיצג את אמינות ערוץ התקשורת מקודקוד u לקודקוד v. $r(u,v)$ היא ההסתברות שלא תהיה טעות במעבר בין u ל- v. נניח שהסתברויות אלה בלתי תלויות. תן אלגוריתם יעיל שבהינתן שני קודקודים s ו- t מוצא את המסלול האמין ביותר מ-s ל-t.

פתרון: יהי P מסלול מ- s ל- t. אזי ההסתברות שאין טעות במעבר דרך P הוא $r(P) = \prod_{(u,v)\in P} r(u,v)$.

אם נקח $\log$ של זה נקבל

$$\log r(P) = \log\left(\prod_{(u,v)\in P} r(u,v)\right) = \sum_{(u,v)\in P} \log r(u,v).$$

בביטוי שקיבלנו ייתכן שיופיע $\log 0 = -\infty$. כדי למנוע בעיות כאלו נמחוק כל קשת שבו $r(u,v) = 0$. במקרה שזה מנתק את כל המסלולים מ- s ל- t אז בהמשך נגלה שלא נשאר אף מסלול מ- s ל- t. אם זה יקרה, אז נחפס מסלול כלשהו מ- s ל- t (שבו $r(P) = 0$) ונחזיר אותו. (למה זה אופטימלי במקרה כזה?)

לכן כדי לקבל את $r(P)$ מקסימום אז צריכים לקבל מינימום של $\sum_{(u,v)\in P} w(u,v)$, כאשר הערכים $w(u,v) = -\log r(u,v)$ הם אי-שליליים. זה מסלול קצ ביותר ביחס למשקלות w וניתן למצוא אחד כזאת בעזרת האלגוריתם של Dijkstra. (משחזרים מסלול אופטימלי דרך הערכים $\pi(v)$).

8. תרגיל: נתון גרף אם משקלות אי-שליליות על הקשתות. נתונות קבוצות $A, B \subseteq V$ לא ריקות וזרות. לכל צומת $a \in A$ מצא את המסלול הקצר ביותר ל- B.

פתרון: דרך פשוטה לפתור את הבעיה היא להתחיל להריץ את $\mathrm{DIJKSTRA}(G, w, a)$ לכל $a \in A$ ולעצור כל ריצה כאשר מוציאים לפעם הראשונה מ- Q צומת מ-B (שהוא הצומת ב- B האכי קרוב ל- a).

דרך יותר מתוחכמת: נבנה גרף מכוון חדש $G' = (V', E')$ בצורה הבאה. ראשית נוסיף את כל צמתי V ל- V'. במקרה ש- G גרף לא-מכוון נשים זוג קשתות בכיוונים הפוכים ב- E' במקום כל צלע לא מכוונת מ- E. במקרה ש- G גרף מכוון לכל קשת $e = (u, v) \in E$ נשים קשת בכיוון ההפוך $e' = (v, u)$ ב- E'. עכשיו נוסיף שני צמתים חדשים s ו- t ונוסיף את הקשתות $(s, a) \forall a \in A$ ו- $(b, t) \forall b \in B$. נקבע $w'(e) = w(e) \forall e \in E$ ו- $w'(e) = 0 \forall e \in E' \setminus E$. נריץ את $\mathrm{DIJKSTRA}(G', w', t)$. המסלול הקצר ביותר מ- t לכל צומת חייבת לעבור דרך B (לפי מבנה G'). לכן לכל $a \in A$ נגלה את המסלול הקצר ביותר מ- t אליו, המסלול המתאים בכיוון ההפוך ומבלי t בגרף המקורי G היא המסלול הקצר ביותר מ-a ל- B.

למה הרצנו את $\mathrm{DIJKSTRA}(G', w', t)$ בגרף שבו הפכנו את כיווני הקשתות המקוריות ולא את $\mathrm{DIJKSTRA}(G', w', s)$ מבלי להפוך את הכיוונים?!? זה לא יעבוד! ייתכן שהרצת $\mathrm{DIJKSTRA}(G', w', s)$ לא ישרה מסלול ל- B לחלק מצמתי A! (אולי קצר יותר לקחת קשתות לחבר את B מבפנים ורק לחבר אותו לצומת יחיד ב- A.)

הערה: באמת אין צורך בצומת s - מטרתה היא רק להבליט את הפתרון הנכון מול פתרון שגוי.

חישוב המרחקים בין כל זוג של צמתים:

9. $\mathrm{FLOYD\text{-}WARSHALL}$ מחשב את המרחק בין כל זוג צמתים בזמן $O(|V|^3)$ בהנחה שאין מעגל שלילי בגרף. עבור האלגוריתם הזו מניחים ש- $V = \{1, 2, \ldots, n\}$ ומשתמשים בצמתים כאינדקסים לשורות ועמודות מטריצה. בקלט נתון המטריצה $W = (w_{i,j})$ שבו

$$w_{i,j} = \begin{cases} 0 & i = j \\ w(e) & e = (i,j) \in E \\ +\infty & (i,j) \notin E \end{cases}$$

האלגוריתם מחשב מטריצה $D = (d_{i,j})$ שבו $d_{i,j}$ הוא המרחק המינימלי בין צומת i ל- j.

```
FLOYD-WARSHALL(W)
1  n ← row[W]
2  D^(0) ← W
3  for k ← 1 to n
4        do for i ← 1 to n
5              do for j ← 1 to n
6                    d_ij^(k) ← min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
7  return (D^(n))
```

באיתחול מתחשבים רק במסלולים שאין להם צמתי ביניים. בכל שלב (שורה 2) מוסיפים צומת k לקבוצת הצמתים שיכולים להופיע כצמתי ביניים ומעדכנים את כל המרחקים. בסוף התהליך מקבלים את התשובה הנדרשת. היסוד באלגוריתם הוא שכל תת-מסלול של מסלול מינימלי הוא גם מינימלי (בין קצותיו).