מסלולים קצרים ביותר בגרף עם אורכים שליליים

הערה: אין מעגלים שאורכם שלילי (אחרת מרחקים אינם מוגדרים)
האלגוריתם שנלמד מזהה מעגל שלילי אם קיים.

תזכורת: Bellman-Ford

חזור $n-1$ פעמים:
- בצע רלקסציה לכל הקשתות

נכונות:
באיטרציה ה-k מחושבים מרחקים במסלולים שבהם לכל היותר k קשתות.

זמן ריצה: $O(n\cdot m)$, כלומר $O(n^2)$ בגרף מישורי

היום נראה אלגוריתם של בזמן $O(n \log^2 n)$

(הטוב ביותר שידוע היום: $O(n \log^2 n / \log\log n)$)

תזכורת מהשיעור הקודם ומהתרגיל: עודפים (slack costs)

$p(v)$ - פונקציה מחיר $p: V \to \mathbb{R}$

אורך עודף $c_p(uv) = c(uv) + p(u) - p(v)$

״בכמה המסלול דרך uv יקר מהמסלול עפ״י p?״

מהתרגיל: - שימוש באורכים עודפים אינו משנה מסלולים קצרים.
- האורך של מסלול מ-s ל-t משתנה ב- $p(s)-p(t)$
- אם p מוגדרת מרחקים בגרף אז c_p אי-שליליות.

$\Leftarrow$ אם ידועים מרחקים מקודקוד כלשהו v, ניתן לחשב מרחקים מכל קודקוד s באמצעות Dijkstra בזמן $O(n)$.

הרעיון: לשלב האבחנה הזו עם פתרון רקורסיבי באמצעות ספרטורים.

האלגוריתם: $SP(G, s)$

1 - מצא מפריד מעגלי $G_{out}, G_{in} \Longleftarrow C$

2 - בחר קדקד כלשהו $r \in C$

3 - חשב δ_i: מרחקים קצרים מ-r ב-G_i (רקורסיבית $SP(G_i, r)$)

4 - חשב A_i: מרחקים בין כל זוג קדקדים של C ב-G_i
(באמצעות MSSP, δ_i הוא הפוט' ההתחלתי)

- יהי G_i' הגרף המלא על הקדקדים של C, כאשר אורך הקשת uv נתון ע"י המרחק מ-u ל-v ב-G_i (נתון ב-A_i)

5 - חשב B: מרחקים קצרים ב-G מ-r לכל הקדקדים של C

נקרא G' Dense Distance Graph

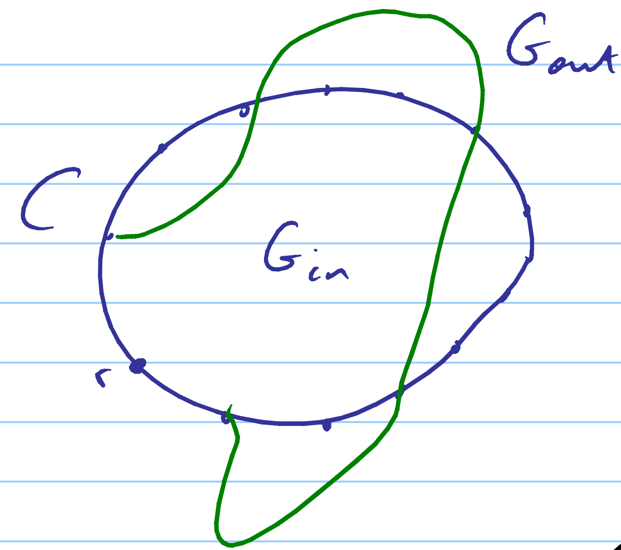
ע"י Bellman-Ford בגרף $G' = G'_{in} \cup G'_{out}$

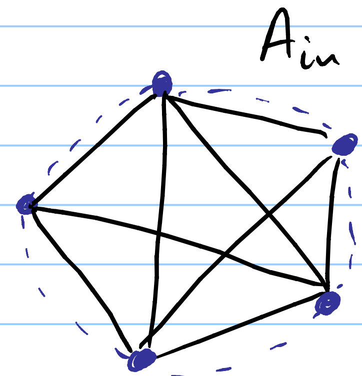
6 - חשב δ_i': מרחקים קצרים ב-G מ-r לכל הקדקדים של G_i

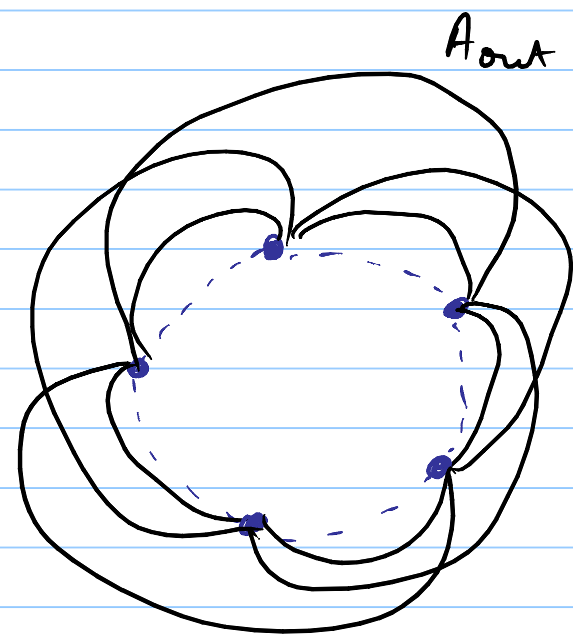
ע"י Dijkstra בגרף G_i, עם פונקצית מחיר δ_i ואתחול המרחקים לקדקדים של C לערכם ע"פ B.

- יהי $\delta' = \delta'_{in} \cup \delta'_{out}$ מרחקים מ-r ב-G

7 - חשב מרחקים מ-s ב-G ע"י Dijkstra ופונקצית מחיר δ'.







זמן ריצה:

1.	ספרטור -	G	$O(n)$
4.	MSSP -	G_i	$\sum_i O(n_i \log n_i) = O(n \log n)$
5.	Bellman-Ford	G'	קודקודים $O(\sqrt{n})$, קשתות $O(n)$ $\Rightarrow O(n^{3/2})$
6.	Dijkstra	G_i	$\sum_i O(n_i) = O(n)$
7.	Dijkstra	G	$O(n)$

$$O(n^{3/2})$$

$$T(n) = T(n_1) + T(n_2) + O(n^{3/2}) = O(n^{3/2})$$

כבר יותר טוב מ $O(n^2)$, אבל הבטחנו $O(n \log^2 n)$ ....

צוואר הבקבוק הוא חישוב Bellman-Ford ב-G'.

בתחילת האיטרציה ה-k של BF, $B(u)$ הוא המרחק מ-r ל-u שמשתמש בלכל היותר k קשתות.

לעשות רלקסציה לכל הקשתות של G' פירושו לחשב:

$$\forall v \in C \qquad B(v) = \min_{u \in C} \begin{Bmatrix} B(u) + A_{in}(u,v), \\ B(u) + A_{out}(u,v) \end{Bmatrix}$$

אם נגדיר $A'_i = B(u) + A_i(u,v)$

A'_i היא מטריצה עם $O(\sqrt{n})$ שורות ועמודות.

אז צעד של BF מוצא את האיבר המינימלי בכל עמודה של A'_i

באופן נאיבי זה לוקח $O(n)$ זמן (כמספר האיברים ב-A'_i)

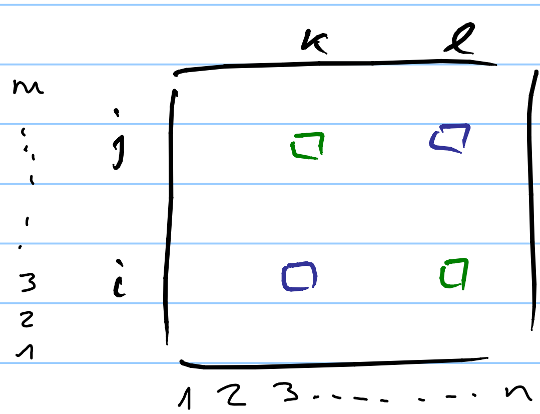
נראה שניתן לעשות זאת ב $O(\sqrt{n} \log n)$ זמן

(למעשה אפשר ממש ב-$O(\sqrt{n})$ SMAWK).

עיקוף: תכונת Monge

מטריצה $A_{m\times n}$ נקראת Monge אם עבור כל $i<j$ ו $k<l$ קיים

$$A_{ik}+A_{jl} \geq A_{il}+A_{jk}$$



הגד׳: $\mathcal{E}(j)=\arg\min_i A_{ij}$ סדרת האינדקסים של השורות בהן מתקבלים האיברים המינימליים בעמודה

<u>למה</u> - אם A מטריצת Monge אז הסדרה $\mathcal{E}$ אינה עולה.

הוכחה: נניח $\mathcal{E}(k)=i$. כלומר, עבור $j>i$,

$$A_{ik}\leq A_{jk}$$

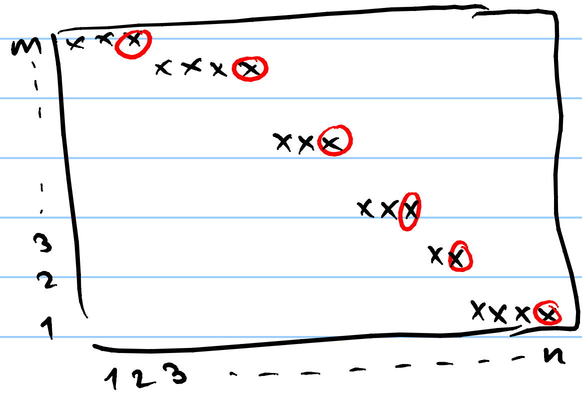
מכיוון שעבור $l>k$

$$A_{ik}+A_{jl}\geq A_{il}+A_{jk}$$

חייב להתקיים

$$A_{jl}\geq A_{il}$$

כלומר $\mathcal{E}(l)\leq i$ $\square$



למשל, כך יכולה להראות סדרת האינדקסים של האיברים המינימליים בעמודות.

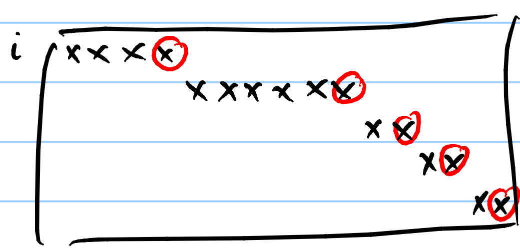
נגדיר breakpoint של A להיות זוג (i,k) כך ש $\mathcal{E}(k)=i$
אבל $\mathcal{E}(k+1)\neq i$

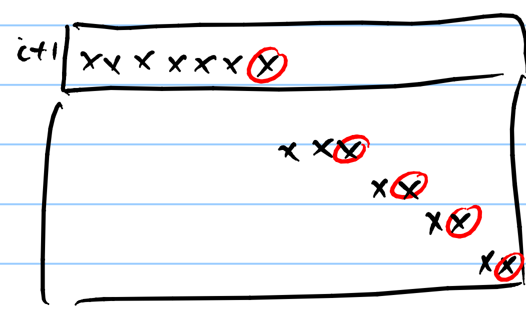
נשים לב שאם ידועה סדרת ה-breakpoints, אז ניתן למצוא את כל האיברים המינימליים של כל העמודות בזמן לינארי במספר העמודות.

כיצד מוצאים את ה breakpoints של A?

נניח שמצאנו את ה-breakpoints של המטריצה המתקבלת מ-i השורות הראשונות של A.

לכל BP (j,k) בסדר עולה של העמודות, נשווה את $A_{i+1,k} \; ! \; A_{j,k}$





מהלמה:

אם $A_{i+1,k} \leq A_{j,k}$ אז לכל עמודה $k > \ell$ המינימום נמצא בשורה $i+1$
$\Longleftarrow$ ה-BP (j,k) מתבטל

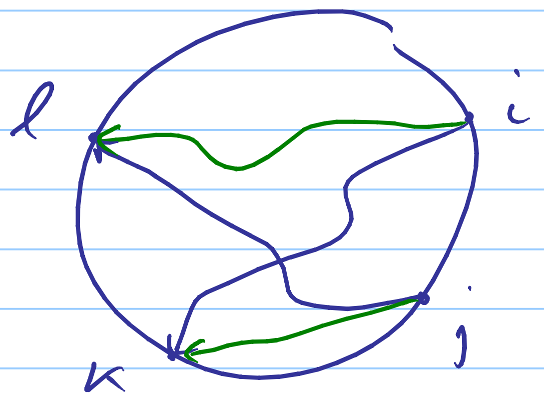
אם $A_{i+1,k} > A_{j,k}$ אז לכל עמודה $k < \ell$ המינימום אינו בשורה $i+1$.
$\Longleftarrow$ אפשר להפסיק להשוות

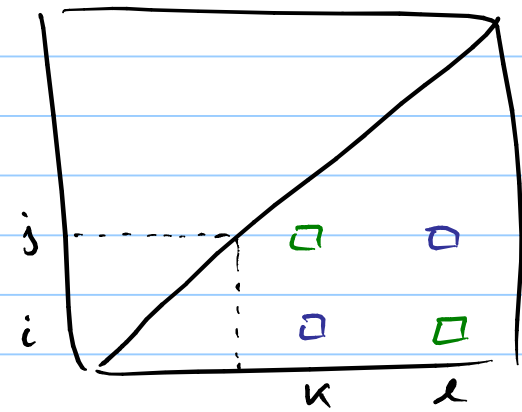
את (j,k) ה- BP הראשון ששרד, צריך למצוא את העמודה בה שורה i מפסיקה להיות המינימום ושורה j מתחילה להיות המינימום.
ניתן לעשות זאת ע״י חיפוש בינארי.

<u>אנליזת זמן:</u>

בכל פעם שמוסיפים שורה מוסיפים לכל היותר BP אחד.
לכן, מספר ה- BP הכולל שנוספו לכל היותר m
כל השוואה שנעשית, פרט לאחרונה בכל איטרציה, מורידה BP אחד.
לכן מספר ההשוואות הכולל הוא לכל היותר $2m$.
בנוסף, בכל איטרציה עושים חיפוש בינארי על לכל היותר n איברים,
כלומר בזמן $O(\log n)$.
סה״כ $O(m\log n)$ למציאת ה- BP של מטריצה עם m שורות ו- n עמודות

מה הקשר לבעיה שלנו ?

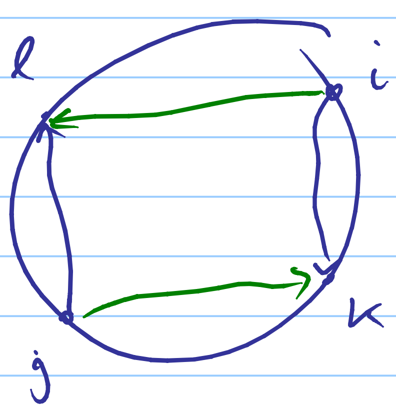


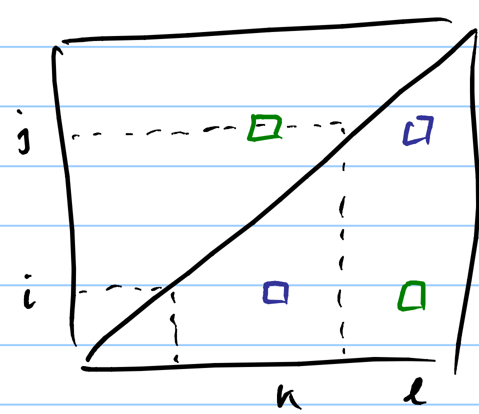


$i<j<k<l$

$$\underset{+B(i)}{A(i,k)}+\underset{+B(j)}{A(j,l)} \geq \underset{+B(i)}{A(i,l)}+\underset{+B(j)}{A(j,k)}$$

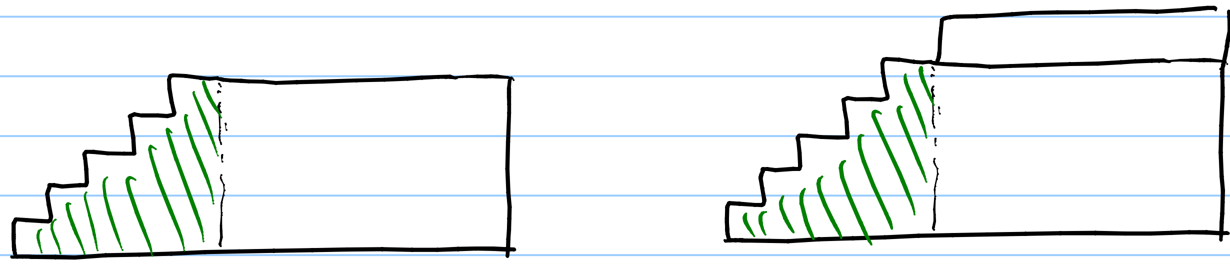
$$A'(i,k)+A'(j,l) \overset{?}{\geq} A'(i,l)+A'(j,k)$$





???

המטריצה שלנו אינה Monge, אבל כל מטריצה משולשית היא Monge.



לשורה ה $(i+1)$ אין כל השפעה על ה- BPs במשולש הריק.
שאר המטריצה ריבועית, ולהוסיף שורה למטריצה ריבועית אנו כבר יודעים.

כלומר, אנחנו יודעים למצוא את ה- BPs של מטריצת Monge משולשת ב $O(n\log n)$ זמן.

$\Leftarrow$ יודעים למצוא את האיבר המינימלי בכל עמודה של מטריצת Monge משולשת ב- $O(n\log n)$ זמן

$\Leftarrow$ יודעים למצוא את המינימום בכל עמודה של A_i' ע"י חלוקה לשתי מטריצות משולשות, והשוואת שני האיברים המינימליים בכל עמודה ב $O(n\log n)$ זמן.

כאן n הוא מספר השורות/עמודות במטריצה. כלומר, גודל הספרטור C כלומר $O(\sqrt{n})$.

מציאת כל המינימום בעמודות שקולה לאיטרציה אחת של BF.
מספר האיטרציות הוא כמספר הקדקודים (שהם $\sqrt{n}$), כלומר
סה"כ זמן הריצה של BF הוא

$$\sqrt{n}\cdot\sqrt{n}\log(\sqrt{n}) = O(n\log n)$$

וזמן הריצה של האלגוריתם כולו:

$$T(n) = T(n_1) + T(n_2) + O(n\log n) = n\log^2 n$$

מהו צוואר הבקבוק האמיתי של האלגוריתם?

- MSSP $\quad O(n\log n)$
- BF $\quad O(n\log n)$ (אפשר $n\alpha(n)$)
- Dijkstra $\quad O(n)$

כלומר, MSSP הוא צוואר הבקבוק.

הזכרנו כבר של - MSSP יש חסם תחתון של $O(n\log n)$.
האם אין תקווה לשיפור?

$1 \Rightarrow$ אם הזמן של MSSP ו- BF אינו מאוזן, יכולים להרשות להריץ יותר BF.

במקום לחלק לשניים, משתמשים ב- r-division עם

$r=\frac{n}{\log n}$ (בספרטור רגיל $r=\frac{3n}{4}$)

כל איטרציה עדיין לוקחת $n\log n$ (עכשיו MSSP ו- BF מאוזנים)
אבל עומק הרקורסיה הוא רק

$$\log_{\log n}(n)=\frac{\log n}{\log\log n}$$

וזמן הריצה יוצא

$$O(n\log^2 n/\log\log n)$$

(2) החסם התחתון של MSSP אינו תופס לחישוב המרחקים
רק בין הקדקודים של הפאה האינסופית

?

חסם תחתון חזק יותר

אלגוריתם לינארי לבעיה