<u>מסלולים קצרים בין כל הזוגות</u>

בהינתן גרף, נרצה לחשב את המסלולים הקצרים בין כל הזוגות. הפלט יהיה שתי מטריצות $n \times n$: $\delta[i][j]$ זה המרחק. $\pi[i][j]$ זה $\pi(j)$ במסלול $j \mapsto i$. (כלומר, כדי לדעת איך להגיע מ-i ל-j: הולכים אחורה לפי ה-π)

פתרון 1: נריץ אלגוריתם SSSP מתאים, n פעמים:

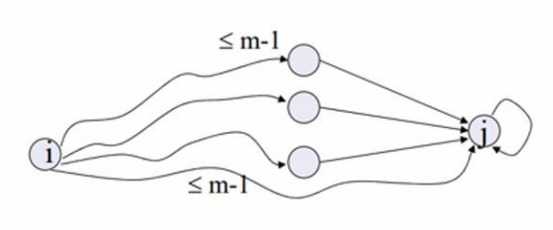
$BFS - O(n^2 + nm) \longrightarrow O(n^3)$
$dijkstra - O(nm + n^2 logn) \longrightarrow O(n^3)$
$BF - O(n^2 m) \longrightarrow O(n^4)$

המקרה הגרוע הוא אם הגרף הוא קליקה, כלומר $m = n^2$

פתרון 2: נשתמש במטריצת שכנויות, תכנות דינאמי:

מטריצת שכנויות: $W[i][j] = \begin{cases} 0, & i = j \\ w(i,j), & (i,j) \in E \\ \infty, & else \end{cases}$

D^m היא מטריצה של משקלי מסלולים מינימליים, עבור מסלולים עם עד m צלעות. $D^m[i][j]$ זה $\delta(i,j)$ עם עד m צלעות.



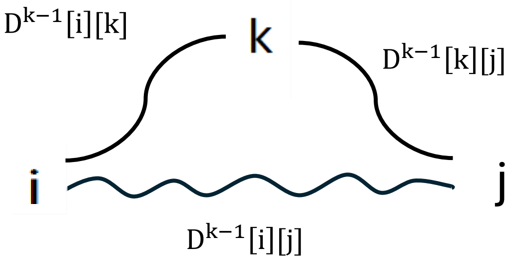
$D^1 = W$, $D^0[i][j] = \begin{cases} 0, & i = j \\ \infty, & else \end{cases}$

אם אנחנו יודעים את $D^{m-1}[i][j]$, אפשר לחשב את $D^m[i][j]$: או שהוספת הצלע לא עזרה (ואז ניקח את הקודם), או שהוספת הצלע עוזרת ואז נשווה בין כל האופציות: מסלול לקודקוד k עם m-1 צלעות, והצלע (k,j). נשים לב שאם $k = m - 1$, זה פשוט האופציה הראשונה. אז בעצם:
$D^m[i][j] = \min_k\{D^{m-1}[i][k] + w(i,k)\}$.

לכל מסלול פשוט יש עד $n - 1$ צלעות, אז מספיק למצוא את D^{n-1}. מתקיים $D^m = D^{n-1}$ לכל $m \geq n - 1$.
אם נעשה חישוב רגיל: יש $n - 1$ מטריצות, בכל אחת צריך למצוא מינימום מתוך $O(n)$ אופציות עבור n^2 תאים. סה"כ $O(n^4)$.

אפשר לשפר את זה, אם נשתמש בכפל מטריצות מיוחד (גיאומטריה טרופית): $AB[i][j] = \min_k\{A[i][k] + b[k][j]\}$.
זה אסוציאטיבי (כי חיבור ומינימום הם אסוציאטיביים), ומתקיים: $D^1 = W, \quad D^m = D^{m-1}W$. אז אפשר פשוט לחשב מטריצות "בריבוע", עד שנעבור את D^{n-1}: $D^{16} = D^8D^8 \longrightarrow D^8 = D^4D^4 \longrightarrow D^4 = D^2D^2 \longrightarrow D^2 = DD$
זה נותן זמן ריצה של $O(n^3 logn)$, כי נעשה $logn$ העלאות בריבוע וכל אחת היא n^3.

עוד שיפור: **אלגוריתם פלויד-וורשאל – Floyd-Warshall:**
הגדרה: עבור מסלול $P = (i, v_1 \cdots v_k, j)$, **קודקוד ביניים** הוא כל אחד מ: $\{v_1 \cdots v_k\}$.
עבור $V = \{1 \cdots n\}$, $D^k[i][j]$ זה אורך המסלול הקצר $j \mapsto i$ שבו קודקודי ביניים הם רק מתוך $\{1 \cdots k\}$. $D^n[i][j] = \delta(i,j)$.
אז מתקיים: $D^k[i][j] = \min(D^{k-1}[i][j], D^{k-1}[i][k] + D^{k-1}[k][j]) \qquad D^0[i][j] = W[i][j]$




```
FW(G)          Θ(n^3)
  D ← W
  for(k = 1 → n)
    for(i = 1 → n)
      for(j = 1 → n)
        if(D[i][j] > D[i][k] + D[k][j])
          D[i][j] ← D[i][k] + D[k][j]
          π[i][j] ← π[k][j]
  return D
```

אם הגרף ריק יחסית – m הרבה יותר קטן מ n^2 – אז מהיר יותר לעשות דיקסטרה n פעמים.
אם יש צלעות שליליות, אפשר לעשות **משקול מחדש – reweighting.** נחלק משקלים כך שכל המשקלים אי-שליליים, בצורה שבה שמסלול ב G' הוא קצר אמ"מ אותו מסלול קצר בG. אז לדוגמה, להוסיף את אותו משקל לכל הצלעות **לא** יעבוד.

משפט reweighting: לכל פונקציה $h: V \to \mathbb{R}$, נגדיר: $w'(u,v) \coloneqq w(u,v) + h(u) - h(v)$. זה משמר את המסלולים, כי כל מסלול $v \mapsto u$ משתנה באותו ערך: $+h(u) - h(v)$. כל מה שבאמצע מצטמצם.

האלגוריתם של ג'ונסון – Johnson's algorithm קובע h ככה שכל המשקלים יהיו אי-שליליים:

נוסיף קודקוד s עם צלע לכל קודקוד במשקל 0 – $O(n)$

נעשה $BF(G,s)$ - $O(nm)$

ונגדיר $h(v) \coloneqq \delta(s,v)$. – $O(n)$

בגלל אי"ש המשולש, מתקיים:

$$\delta(s,v) \le \delta(s,u) + w(u,v) \Longrightarrow h(v) \le h(u) + w(u,v) \Longrightarrow w'(u,v) = w(u,v) + h(u) - h(v) \ge 0$$

לכל $u \in V$, נעשה דיקסטרה כדי למצוא $\delta'(u,v)$ לכל v – $O(nm + n^2 logn)$,

ולכל $v \in V$ נגדיר: $D(u,v) \leftarrow \delta'(u,v) + h(v) - h(u)$ כדי להחזיר את המשקלים למצב המקורי.

סיכום APSP:

בלי משקלים	BFS n פעמים	$O(n^2 + nm)$
משקלים אי-שליליים	דיקסטרה n פעמים	$O(nm + n^2 logn)$
משקלים שליליים	כפל מטריצות – גיאומטריה טרופית	$O(n^3 logn)$
	קודקודי ביניים – Floyd-Warshall	$O(n^3)$
	ג'ונסון – reweighting	$O(nm + n^2 logn)$

מציאת מסלול קל ביותר עם בדיוק k צלעות:

פתרון רקורסיבי: מקרי בסיס:

מסלול מקודקוד לעצמו באורך 0 שווה 0.

מסלול באורך 1 זה הצלע.

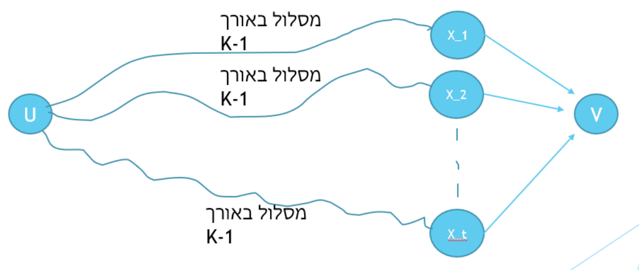
מסלול באורך k זה המסלול הכי קל באורך k-1 ועוד צלע.

ברקורסיה זה $O(n^k)$. בתכנות דינאמי $O(nk)$

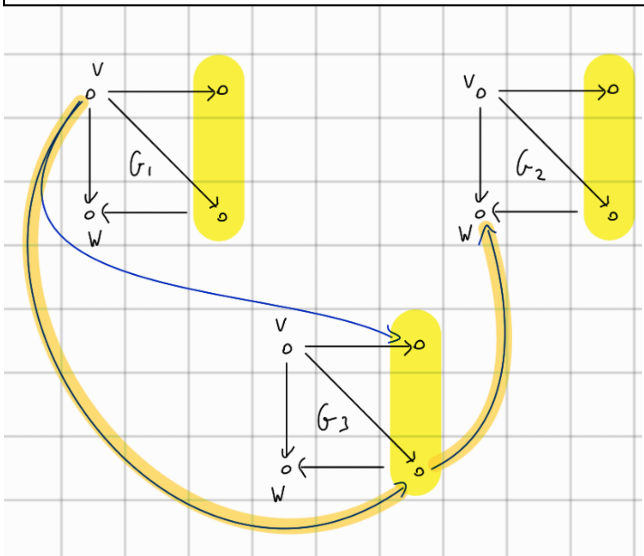
נמלא מטריצה $DP[n][k]$ שבה $DP[i][j]$

זה המסלול הקל $i \mapsto u$ באורך j.

נחזיר את $DP[v][k]$



מציאת מסלול $w \mapsto v$ שעובר דרך לפחות קודקוד אחד מקבוצה $U \subseteq V$. נבנה 3 עותקים של הגרף. בנוסף נחבר בין $G_1 \mapsto G_2$ רק צלעות שמגיעות לקודקוד ב-U, ובין $G_2 \mapsto G_3$ רק צלעות שיוצאות מקודקוד ב-U. ונמצא מסלול $v_1 \mapsto w_3$.



זיהוי ארביטראז': אנחנו מחפשים מעגל כך ש:

$$w_1 \cdot w_2 \cdot w_3 > 1 \Rightarrow \ln(w_1 \cdot w_2 \cdot w_3) > 0 \Rightarrow \ln(w_1) + \ln(w_2) + \ln(w_3) > 0 \Rightarrow -\ln(w_1) - \ln(w_2) - \ln(w_3) < 0$$

נבנה גרף מתאים ונמצא מעגל שלילי. **מציאת מעגל שלילי:** נעשה BF כרגיל. אם בשלב האחרון צלע יכולה עדיין להתכוץ, הצלע הזאת מחוברת למעגל שלילי (לא בהכרח חלק ממנו). כדי למצוא את המעגל, נלך אחורה לפי π ונשמור רשימה של קודקודים שהיינו בהם. כשמגיעים לאותו קודקוד שוב – סגרנו מעגל.

כמה מסלולים $t \mapsto s$ באורך k קיימים: נשתמש במטריצת שכנויות (לא משקלים, רק 0,1). מתקיים: $A^k[i][j] = n$ אמ"מ קיים מסלול $j \mapsto i$ באורך n. הוכחה באינדוקציה על k: מתקיים עבור 0,1. עבור k: מסלול באורך k זה מסלול $u \mapsto s$ באורך k-1 ועוד צלע (u,t). אז מספר המסלולים הוא:

$$\sum\nolimits_{u \in V} |\{paths\ (s \mapsto u) of\ length\ k-1\}| \cdot |u,v| =$$
$$\sum\nolimits_{u \in V} (A^{k-1}[s][u]) \cdot A[u][v] = \sum\nolimits_{i \in |V|} (A^{k-1}[s][i]) \cdot A[i][v] = A^k[s][v]$$