

אלגוריתמים

מתרגל: יבגני. aristo@cs.huji.ac.il

אתר הקורס: <http://www.cs.huji.ac.il/~algo>
תרגיל כל שבוע.

בתרגיל היום נדבר על דברים שאתם יודעים, ועל דברים שלא תבינו. איזה כיף.

מה זה אלגוריתם?

התזה של Church-Turing (זה לא משפט, זה 'אמונה') – אומר שהבעיות שניתן לפתור בכל מודל הם זהות. פורמלית – בחירת מודל החישוב 'לא חשובה' – הבעיות שניתנות לפתרון במודל אחד ניתנות לפתרון במודל אחר.
אנו מחפשים אלגוריתמים 'יעילים' למעזר שימוש במשאבים שונים (זמן, מקום, תקשורת,...). איך מודדים? זמן הריצה הוא פונקציה של גודל הקלט.

סימונים אסימפטוטיים:

היו $f, g: N \rightarrow R$. $f = O(g)$ אם "מ" קיים $c > 0$ ו N כך שלכל $n > N$,
 $f(n) \leq cg(n)$.

מסמנים $f = o(g)$ אם $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$.

מסמנים $f = \Omega(g)$ אם קיים $c > 0$, N , כך ש $\forall n \geq N$ מתקיים $f(n) \geq cg(n)$.

נאמר $f = \Theta(g)$ אם "מ" $f = O(g)$ וגם $f = \Omega(g)$.

דוגמאות:

$$n = O(n + 5)$$

צורת כתיבה - $f = n^{O(1)}$ - מה זה אומר? קיים c קבוע ו N כך ש $\forall n > N$ מתקיים $f(n) \leq n^c$.
ל f כזו נקרא פונקציה פולינומיאלית.

גרסה של תזה Church-Turing

בעיות שיש להן אלגוריתם פולינומיאלי במודל חישוב אחד, יש להן גם אלג' פולינומיאלי במודל חישוב אחר.

דוגמא לחישוב זמן ריצה**סדרת פיבונאצי**

$$f_0 = f_1 = 1$$

$$f_k = f_{k-1} + f_{k-2}$$

בעיה – לחשב את f_k (בהנתן k).
אלגוריתם 1 – ע"פ הנוסחה:

- $f(k)$
 - if ($k > 1$)
 - return $f(k-1) + f(k-2)$
 - else
 - return 1

נסמן ב $g(k)$ את זמן הריצה על קלט k . נספור את פעולות החיבור בלבד. $g(0) = g(1) = 0$.
 $g(k) = g(k-1) + g(k-2) + 1$

g	0	0	1	2	4	7
f	1	1	2	3	5	8

נשים לב כי $f(k) \leq g(k+2)$.

$$f(k) \geq 2^{\frac{k-1}{2}} \quad \text{טענה:}$$

הוכחה: $k=0,1$ - מהצבה.

שלב האינדוקציה:

$$f(k+1) = f(k) + f(k-1) \geq 2^{\frac{k-1}{2}} + 2^{\frac{k-2}{2}} = \left(2^{\frac{1}{2}+1}\right) 2^{\frac{k-2}{2}} \geq 2^{\frac{k-2}{2}+1} = 2^{\frac{k}{2}}$$

$$f(k) \geq (\sqrt{2})^{k-1}, \text{ כלומר,}$$

$$\frac{\sqrt{5}+1}{2} \cdot \sqrt{2} \text{ אפשר לכתוב}$$

אלגוריתם 2:

חישוב ממקום 0 עד ל- k . זמן ריצה - סד"ג של $O(k)$

אלגוריתם 3:

$$\begin{pmatrix} f_{k+1} \\ f_k \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} f_k \\ f_{k-1} \end{pmatrix}$$

$$\begin{pmatrix} f_{k+1} \\ f_k \end{pmatrix} = A^k \begin{pmatrix} f_1 \\ f_0 \end{pmatrix}$$

נניח ש- k הוא חזקת 2 $k = 2^n$.

$$A^k = \left(A^{\frac{k}{2}}\right)^2 = \left(\left(A^{\frac{k}{4}}\right)^2\right)^2 = \dots$$

מאחר וההכפלה היא $O(1)$, אזי כל הפעולה היא $O(\log k)$.

ל- k כללי, $a_i = 0, 1$, $k = \sum_{i=0}^n a_i 2^i$. לדוגמא - $k = 2^2 + 2^5$ אז $A^k = A^{2^3} \cdot A^{2^5}$. באופן כללי,

נכפיל את A^{2^i} עבור $a_i = 1$. זמן הריצה יהיה עדיין $O(\log k)$.

נחשב את זמן הריצה של האלגוריתמים ביחידות של פעולות על ביטים.
 אלגוריתם 1 - אקספוננציאלי.

אלגוריתם 2 – כמות פעולות החישוב הוא $O(n)$. מהו מספר הביטים ב $f(k)$? $2^{\frac{k-1}{2}} \leq f_k \leq 2^k$ - כלומר, מספר הספרות הוא $O(k)$. ע"כ – זמן הריצה הוא $O(k^2)$. k פעולות חיבור, כל אחת לוקחת $O(k)$ זמן).

אלגוריתם 3 – יש כאן $O(\log k)$ פעולות כפל מטריצות, של 2×2 . כפל לוקח את מספר הספרות בריבוע, שזה $O(\log k)$ הכפלות של מספרים (בלי לשכוח את החיבור!) – ולכן הזמן הכולל הוא $O(k^2 \log k)$ (פעולת כפל על k ביטים היא ב $O(k^2)$). קיצר – שכרנו יצא בהפסדנו.

נסמן זמן ריצה על קלט k ב $g(k)$. $g(k) = g\left(\frac{k}{2}\right) + O(k^2)$. ע"פ משפט האב – מדובר ב $g(k) = O(k^2)$.

30.10.2006

תרגול 2

בעיה: קידוד למען דחיסה.

רוצים לשלוח קובץ טקסט בו 35 סוגי אותיות.

קוד הופמן

http://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%93_%D7%94%D7%95%D7%A4%D7%9E%D7%9F

דרך א': כל אות תקודד בעזרת סדרה של 6 ביטים.

דרך ב': נניח שאנו יודעים מראש את הקובץ. דוגמא: 4 אותיות שונות, a, b, c, d , שמופיעות

100, 10, 20, 20 פעמים בהתאמה.

בקידוד הרגיל, אורך הקובץ יהיה $= (100 + 10 + 20 + 20) \cdot 2 = 300$ ביט, כלומר –

בואו נסתכל על קידוד אחר. את a נקודד עם 0, את b עם 110, את c עם 111, ואת d עם 10. אפשר לקרוא אותו בצורה חמדנית – לנסות לקרוא כמה שיותר. האורך המתקבל הוא $100 \cdot 1 + 30 + 60 + 40 = 230$, חסכנו 70 ביט!

קצת פורמליות – נתון א"ב Σ , ונתונה פונקציית צפיפות $f: \Sigma \rightarrow N$ (סופרת מספר הופעות של כל אות בקובץ). המטרה היא למצוא קוד בינארי לא"ב Σ , $C: \Sigma \rightarrow \{0,1\}^*$ (כלומר – מ Σ לאוסף כל המילים הסופיות ב $\{0,1\}$). כמובן שאנו מעוניינים בקוד ניתן לפענוח.

קוד ניתן לפענוח: הינו קוד C כך שהעתקת הקידוד $\tilde{C}: \Sigma^* \rightarrow \{0,1\}^*$, $\tilde{C}(a_1, \dots, a_k) = c(a_1)c(a_2)\dots c(a_k)$

אם $u, v \in \Sigma^*$ שתי מילים, אז u נקראת רישא של v אם קיימת מילה w (כלשהי) כך ש $v = uw$. **הגדרה:** קוד C נקרא חסר רישות אם לכל שתי אותיות $a, b \in \Sigma$ $C(a)$ אינה רישא של $C(b)$.

דוגמאות:

1. קידוד באורך קבוע (לכל אות מילת קוד באותו אורך) הינה קוד חסר רישא. (לדוגמא ASCII).

2. $0, 10, 110, 111$ - הינו קוד חסר רישות

a. טענה: קוד חסר רישא הוא קוד ניתן לפיענוח.

b. הוכחה: נניח שההעתקה $\tilde{C}$ איננה חח"ע, כלומר קיימות שתי מילים שונות,

$\tilde{C}(x) = \tilde{C}(y)$ כך ש $x \neq y \in \Sigma^*$

$$x = a_1, \dots, a_k$$

$$y = b_1, \dots, b_m$$

יהי i הקטן ביותר עבורו מתקיים $a_i \neq b_i$ (קיים, כי $x \neq y$).

$$c(a_1), \dots, c(a_{i-1}), c(a_i)$$

$$c(b_1), \dots, c(b_{i-1}), c(b_i)$$

בה"כ $|c(b_i)| \geq |c(a_i)|$, אבל $\tilde{C}(x) = \tilde{C}(y)$, ולכן $c(a_i)$ הינו רישא של $c(b_i)$.

הצגה אחרת של קוד חסר רישא, בעזרת עץ: T הוא עץ בינארי, בכל עלה יש אות אחרת מהא"ב Σ . לכל קודקוד הצלע לבן שמאלי מסומנת ב-0 ולבן ימני ב-1.

קידוד של אות $a \in \Sigma$ הוא שרשור הסימנים על הצלעות משורש העץ ל V_a (הקודקוד עליו יושב a) **טענה:** קוד שמתקבל מעץ הוא קוד חסר רישא.

הוכחה: אם $a, b \in \Sigma$ ו $c(a)$ הוא רישא של $c(b)$ אז המסלול מהשורש ל V_b עובר דרך V_a , ואז V_a איננו עלה.

(עוד) קצת הגדרות:

אם C קוד, לכל $a, b \in \Sigma$ נסמן ב $l(a)$ את האורך $c(a)$ בביטים.

הגדרת הבעיה: נתון א"ב Σ ו $f: \Sigma \rightarrow N \cup \{0\}$ פונקציית צפיפות. רוצים למצוא קוד חסר רישא

כך ש $\sum_{a \in \Sigma} f(a) \cdot l_c(a) \equiv W(C)$ יהיה מינימלי.

האלגוריתם המוצע – ע"י Huffman

סימון: שתי אותיות x, y תיקראנו "אחיות" בעץ קידוד T אם יש להן אב משותף.

אנו בעצם רוצים לבנות את העץ המדובר. Huffman עובד על הרעיון של לבנות את העץ **מלמטה למעלה**.

טענה: אם $x, y \in \Sigma$ בעלות צפיפות (שכיחות) **מינימלית** (כמובן שאין דרישה שהם יהיו שווים), אזי קיים עץ אופטימלי בו x, y אחיות.

למה: עץ אופטימלי הוא בהכרח עץ בינארי מלא ($\equiv$ לכל קודקוד מספר הבנים הוא 0 או 2).

הוכחה: נניח שT עץ ובו יש קודקוד v בעל בן אחד בדיוק. נסתכל על העץ T' , המתקבל מ'צמצום' הצלע שיוצאת מv. אורך הקידוד התקצר ולכן העץ T לא היה אופטימלי.

הוכחה: יהי T עץ אופטימלי. נבחר עלים z, w ברמה הנמוכה בעץ שהם אחים (יש כאלה, מאחר ומהלמה הוא מלא). יהו x, y עלים ברמה הגבוהה יותר בעץ. נניח בה"כ שהצפיפות $f(x) \leq f(y)$ ו $f(z) \leq f(w)$ (ברור כי $f(x) \leq f(y) \leq f(z) \leq f(w)$). נראה שהחלפה בין x ו z יכולה רק לשפר. נקרא לעץ אחרי ההחלפה T' .

$$W(T') = \sum_{a \in \Sigma} f(a) \cdot l_{T'}(a) = f(x) \cdot l_{T'}(x) + f(z) \cdot l_{T'}(z) + \sum_{x, z \neq a \in \Sigma} f(a) l_T(a) =$$

$$f(x) l_T(z) + f(z) l_T(x) + \sum_{x, z \neq a \in \Sigma} f(a) l_T(a) =$$

$$f(x) l_T(x) + f(z) l_T(z) + \sum_{x, z \neq a \in \Sigma} f(a) l_T(a) + f(x) l_T(z) + f(z) l_T(x) -$$

$$f(x) l_T(x) - f(z) l_T(z) = W(t) - (f(x) - f(z))(l_T(x) - l_T(z))$$

הטיפול ב w, y זהה.

אלגוריתם:

נסמן $n := |\Sigma|$

1. מתחילים עם n עצים בעלי קודקוד 1 כ"א.
2. נכניס את כל העצים לתור עדיפויות מינימלי Q .
3. כל עוד $|Q| > 1$

- a. מוציאים את שני העצים t_1, t_2 בעלי הצפיפות המינימלית מהתור.
- b. מחברים ביניהם לעץ חדש t (t_1 ו t_2 יהיו בניו), שמגדירים את הצפיפות של העץ החדש כסכום הצפיפויות.
- c. מכניסים את העץ t המחובר לתור העדיפויות.

הוכחת נכונות האלגוריתם:

באינדוקציה על מספר האותיות. עבור $n \leq 2$ - בסדר.

שלב האינדוקציה: מוצאים ב Σ אותיות x, y בעלות צפיפות מינימלית. מגדירים א"ב חדש Σ' שבו $\Sigma' := \Sigma \setminus \{x, y\} \cup \{w\}$, $f(w) := f(x) + f(y)$. (מקביל לאיחוד האותיות באלגוריתם). לפי הנחת האינדוקציה, האלגוריתם פותר נכון את הבעיה עבור Σ' , ומייצר את T' . נראה ש T אופטימלי. נניח שיש עץ T^* טוב יותר, (עבור Σ) – נגיד שהוא אופטימלי, אפשר להניח ש x, y בו הם אחים.

6.11.2006

תכנון דינמי

בעיית תכנון משימות ממושקלת

יש לנו משימות (קטעים), עם משקל. המטרה שלנו לבצע מקסימום משקל של משקלות. נגדיר פורמלית:

יהו n משימות $[s_i, f_i]$ עם משקל w_i .

המטרה: למצוא קבוצה $A \subset [1...n]$ כך ש $i \neq j \in A$ אזי $[s_i, f_i] \cap [s_j, f_j] = \emptyset$ בעלת

משקל מקסימלי, כאשר $w(A) = \sum_{i \in A} w_i$.

נתבונן על משפחה של בעיות – לכל $1 \leq j \leq n$, יהי A_j פתרון לבעיה המצומצמת למשימות $1...j$.

כמובן, שקיימות שתי אופציות – שהוא מכיל את j , או שהוא לא מכיל את j . לכל j יש 2 אופציות:

• $j \in A_j$ – ואז, לכל $1 \leq i < j$ כך ש $f_i > s_j$, מתקיים $i \notin A_j$

• $j \notin A_j$ – ואז, $A_j = A_{j-1}$

נניח שמיינו את המשימות לפי זמן סיום.

טענה:

$w(A_j) = \max \{w(A_{j-1}), w(A_i) + w_j\}$ כאשר i הוא האינדקס המקסימלי עבורו $f_i < s_j$.

איך נבצע זאת? נמלא את המערך:

$w(A_1)$	$w(A_2)$	...	$w(A_n)$
----------	----------	-----	----------

משמאל לימין (בתהליך החישוב הרי משתמשים רק באינדקסים נמוכים יותר).

מיון יעלה לנו $n \log n$ ומילוי תא במערך $O(1)$, ובסה"כ $n \log n$.

(גם את חישוב האינדקס i לכל j ניתן לבצע בזמן $n \log n$).

קירוב ע"י ישר שבור

נניח ויש לנו כמה נקודות (לא הרבה) (כלומר, זוגות (x_i, y_i)). נרצה למצוא קו ישר $ax + b$ שממזער

את $\sum (ax_i + b - y_i)^2$.

ללא הוכחה – המינימום מתקבל באמצעות הגדרת:

$$a := \frac{n \sum x_i y_i - \sum x_i y_i}{n \sum x_i^2 - \left(\sum x_i\right)^2}, b = \frac{\sum y_i - a \sum x_i}{n}$$

הגדרת הבעיה:

נתונות נקודות (x_i, y_i) (ממוין לפי x ים), ונתון קבוע C . חלוקה של $1...n$ היא אוסף קטעים זרים

$[s_i, f_i]$ של המספרים $1...n$ כך ש $[1, n] = \bigcup [s_i, f_i]$.

סימון: e_{ij} – הטעות המינימלית של קירוב ע"י ישר אחד $(x_i, y_i), \dots, (x_j, y_j)$.

המטרה: למצוא חלוקה $[s_1, f_1], \dots, [s_k, f_k]$ הממזערת את $\sum_{i=1}^k e_{s_i f_i} + kc$.

מה עושים? נניח שאני יודע את הפתרון האידיאלי ל $j-1$, אני יכול לפתור את זה עבור j .

נסמן w_j כערך הפתרון האופטימלי עבור הנקודות $(x_1, y_1), \dots, (x_j, y_j)$.

רקורסיה:

$$w_0 := 0$$

$$w_j = \min_{1 \leq i < j} \{w_{i-1} + e_{ij} + c\}$$

גם כאן, כמובן, נשתמש בטבלה כדי לחסוך חישובים מיותרים.

פתרון תרגיל 5d, 1:

נחשוב על המערך כגרף. יש לנו n קודקודים, ובכל אחד מהם מצביע לאחד הבא. (כלומר, יש צלע בין i ל $A[i]$).

נלך לקודקוד האחרון, וברור שאף אחד לא מצביע אליו. אני אעשה n צעדים, ואחריהם – אני אהיה איפשהו בתוך המעגל. נשמור את המיקום שהגענו אליו, ונחשב את אורך המעגל (נלך עד שנחזור למקום).

מהרגע שאנו יודעים את אורך המעגל l , אפשר ללכת שוב עד ל $n-l$ ולמצוא זאת.

13.11.2006

טענה: אם כל הקיבולות ברשת זרימה שלמות, אז קיימת זרימה מקסימלית בשלמים

הוכחה: בשיעור.

מציאת זיווג מקסימלי בגרף דו צדדי

נתון גרף דו צדדי $G = (L \cup R, E)$, ואנו רוצים למצוא זיווג M מקסימלי כאשר $M \subset E$ נקרא זיווג אם $(v, u) \in M$ ו $(v', u') \in M$ אז $u \neq u'$ ו $v \neq v'$.

פתרון באמצעות רשת זרימה:

מגדירים רשת זרימה $G' = (L \cup R \cup \{s, t\}, E')$ כאשר $E' = E \cup L \times \{s\} \cup R \times \{t\}$.
נגדיר קיבולות

$$\forall v \in L, c(s, v) = 1$$

$$\forall u \in R, c(u, t) = 1$$

$$\forall (v, u) \in E, c(v, u) = \infty$$

טענה: אם יש זיווג M כך ש $|M| = m$ אז יש זרימה שגודלה m .

הוכחה: אם M זיווג, נגדיר זרימה f - לכל $(u, v) \in M$ -

$$f(s, u) = 1, f(u, v) = 1, f(v, t) = 1$$

קל לראות ש f זרימה, ו $|f| = m$.

טענה: אם f זרימה בשלמים אז קיים זיווג M כך ש $|M| = |f|$.

הוכחה: נתונה זרימה f . נגדיר M -

$$(u, v) \in M \Leftrightarrow f(u, v) = 1$$

M זיווג כי לכל $u \in L$, יש לכל היותר v אחד עבורו $f(u, v) = 1$.

אם נסתכל בחתך $(\{s\} \cup L, \{t\} \cup R)$ אז הזרימה בחתך (ששווה ל $|f|$) שווה ל $|M|$.

בעיה: יש ליגת כדורסל. אנו באמצע העונה. נתאים את זה למספר הנצחונות עד כה:

- חיפה – 10
- ירוחם – 9
- ירושלים – 9
- דימונה – 8

כל זוג קבוצות פרט לחיפה + דימונה צריך לשחק משחק אחד.

האם יתכן שדימונה תנצח?

שאלה: האם ייתכן כי בסוף העונה לדימונה יהיה מס' נצחונות $\leq$ מכלל קבוצה?

בתוך (ח, י) יש כבר 28 נקודות. יש עוד 3 משחקים בתוכה. בסוף יהיו לפחות 31 נקודות. ואז לפחות לאחת הקבוצות $10 <$ נקודות.

פתרון: נסמן את הקבוצות המשתתפות הן $S \cup Z$, כאשר Z היא הקבוצה שאנו רוצים לברר בנוגע אליה אם היא יכולה לנצח.

נניח של Z m נקודות, ואין ל Z יותר משחקים (אם יש – נוסיף את כל המשחקים שנשארו לה, ונניח שהיא ניצחה בהן).

לכל קבוצה $x \in S$ יש a_x נצחונות. ולכל זוג $x, y \in S$ נשאר g_{xy} משחקים ביניהם. האם יתכן שבסוף הליגה לכל $x \in S$ יש פחות מ m נקודות?

נניח כי המצב הוא:

ירוחם 10

חיפה 9

תל אביב 9

ועבור דימונה $m=10$ (כי אנו מניחים שהיא נצחה בשני המשחקים שנותרו לה).

. נגדיר רשת זרימה כך:

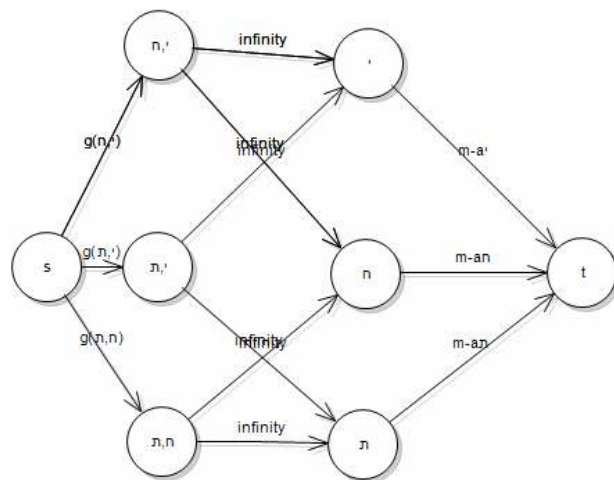
$$G = (V, E)$$

$$V = \{s\} \cup \binom{S}{z} \cup S \cup \{t\}$$

$$c(s, Vxy) = g_{xy}$$

$$c(Vxy, x) = c(Vxy, y) = \infty$$

$$c(x, t) = m - a_x$$



טענה: אם יתכן שבסוף לכל הקבוצות $m \geq$ נקודות, אז קיימת זרימה f ברשת כך ש $|f| = \sum_{x,y \in S} g_{xy}$

(מספר המשחקים שנותרו לי לשחק).

הערה: אם קיימת אפשרות כזו, כל ניצחון יזרום על גבי הצלעות בעלות נפח ∞ למנצח (i, n, t) (אחרי שהוא יעבור את הצלע שמסמנת את מספר המשחקים שנותרו). משם, יש לכל היותר $m - a_x$ נקודות אפשריות, ולכן אם הוא מתאים לדרישה, הוא יגיע ל t).

חזרה על מספרים מרוכבים: יהי $z \in \mathbb{C}$, אזי $z = x + iy$.

לכל $z \in \mathbb{C}$ יש **הצגה פולרית:** אם $z = x + iy$ אז $z = r \cdot e^{i\theta}$

כאשר $r = \sqrt{x^2 + y^2}$, $x = r \cos \theta$, $y = r \sin \theta$.

$$e^z = \sum_{k=0}^{\infty} \frac{z^k}{k!}$$

$$e^{i\theta} = \sum_{k=0}^{\infty} \frac{(i\theta)^k}{k!} = \sum_{n=0}^{\infty} \frac{(i\theta)^{2n}}{(2n)!} + \sum_{n=0}^{\infty} \frac{(i\theta)^{2n+1}}{(2n+1)!} = \sum_{n=0}^{\infty} \frac{(-1)^n \theta^{2n}}{(2n)!} + i \sum_{n=0}^{\infty} \frac{(-1)^n \theta^{2n+1}}{(2n+1)!} =$$

$$\cos(\theta) + i \sin(\theta)$$

$$re^{i\theta} \cdot r'e^{i\theta'} = (rr')e^{i(\theta+\theta')}$$

יהי n שלם. נסמן $\omega_n = e^{\frac{2\pi i}{n}}$. מדובר במספר על מעגל היחידה, בזווית $\frac{2\pi}{n}$. על כן,

$$\omega_n^n = \left(e^{\frac{2\pi i}{n}}\right)^n = e^{2\pi i} = 1$$

זה נקרא שורש יחידה פרימיטיבי מסדר n.

אם $z \in \mathbb{C}$, $z^n = 1$, $z \neq 1$, אזי הוא שורש יחידה פרימיטיבי מסדר n .

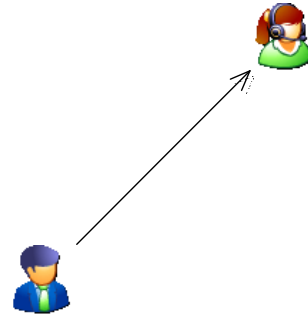
$$\sum_{k=0}^{n-1} z^k = 0 \quad \text{טענה:}$$

$$\left(\sum_{k=0}^{n-1} z^k\right)(z-1) = \sum_{k=0}^{n-1} z^{k+1} - z^k = 0$$

20.10.2006

מכיוון שלא הספקנו ללמוד טרנספורמציות פורייה, אז נמשיך ללמוד זרימה גם היום.

בעיה מהחיים: נניח ואני מעוצבן על מישוהו. אני רוצה לרצוח אותו. סביר אבל שאני רוצה לרצוח גם את אימו, מאחר וגם היא מהווה סכנה.



משפט: Min Cut, Max Flow – תהי $G = (V, E)$ רשת זרימה. אם f זרימה, ו (A, B) חתך, אז $f(A, B) = |f| \leq c(A, B)$. **קיים חתך** A, B וזרימה f כך ש $|f| = f(A, B)$. **הוכחת הטענה:** נפעיל את EK למציאת זרימה מקסימלית. כאשר הוא עצר, אין דרך להגיע מ s ל t בגרף השיווי G' . נגדיר את A ככל הקודקודים v שיש מסלול בגרף השאארי מ s ל v , ו $B = V \setminus A$. ברור כי $c(A, B) = |f|$. מ $f(A, B) = |f| \leq c(A, B)$ נובע שמינימליות החתך גורמת למיקסום הזרימה.

הגדרה פורמלית: נתון גרף $G = (V, E)$, וקודקוד $v_0 \in V$. נתונה $f: V \rightarrow \mathbb{R}^+$, ו $g: E \rightarrow \mathbb{R}^+$.

המטרה: למצוא $A \subset V$ כך ש $v_0 \in A$, ו $\sum_{v \in A} f(v) + \sum_{v \in A, w \notin A} g(v, w)$ מינימלי.

פתרון: נגדיר רשת זרימה. את v_0 כ s . נוסיף קודקוד t . נוסיף צלע מכל קודקוד ל t . כלומר - $G' = \{V \cup \{t\}, E' = E \cup (V \times \{t\})\}$. נגדיר קיבולות $c(u, v) = g(u, v)$ ו $c(v, t) = f(v)$. נגדיר את v_0 להיות המקור.

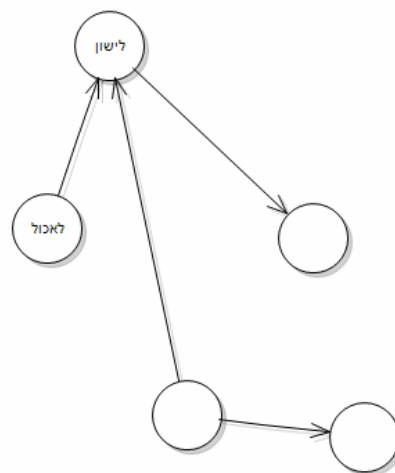
נחפש זרימה מקסימלית. בהנתן חתך (A, B) ברשת, אזי

$$c(A, B) = \sum_{\substack{v \in A \\ u \in B}} c(v, u) = \sum_{\substack{v \in A \\ u \in B - \{t\}}} c(v, u) + \sum_{v \in A} c(v, t) = \sum_{\substack{v \in A \\ u \in V - A}} f(v, u) + \sum_{v \in A} g(v)$$

לפי משפט - **Min Cut, Max Flow**, אם נמצא זרימה מקסימלית ברשת, אזי גם נקבל חתך מינימלי. ראינו ש $c(A, B)$ היא בדיוק מה שרצינו למזער. אם נמצא חתך מינימלי בגרף, הרי נפתור את הבעיה. אבל ראינו זרימה מקסימלית נותנת חתך מינימלי. אז סיימנו. YEY.

בעיה נוספת:

נתון אוסף של פרוייקטים X . לכל פרוייקט x יש $p(x)$. אם $p(x) > 0$ אז אנו רוצים לעשות את x . לכל $x \in X$ נתונה $A_x \subset X$, קבוצת הפרוייקטים שצריכים להתבצע לפני שעושים את x . 'נשים לב שיכולים להיות מעגלים בגרף – יתכן שאני צריך לאכול לפני שאני ישן, ולישון לפני שאני אוכל. אז אני לא אוכל לעשות כלום לעולם'.



נתון גרף $G = (X, E)$. לכל $x \in X$, $A_x = \{y \in X \mid (y, x) \in E\}$. נתון שהגרף אציקלי (לא יתכן מצב כמו קודם).

המטרה היא למצוא קבוצה $S \subset X$ כך שהסכום $\sum_{x \in S} p(x)$ מקסימלי. נדרוש שלכל $x \in S$ מתקיים $A_x \subset S$.

פתרון הבעיה: נגדיר רשת זרימה $G' = (V, E')$. $V = X \cup \{s, t\}$.

$$c(s, x) = p(x)$$

$$c(x, t) = -p(x) \quad E' = E \cup \bigcup_{p(x) > 0} \{s, x\} \cup \bigcup_{p(x) < 0} \{x, t\}$$

$$(x, y) \in E, c(x, y) = \infty$$

(את s נחבר לכל הקודקודים שערכם חיובי, ואת t לכל הקודקודים שערכם שלילי ל t . הקיבולת בין (v, t) יהי $-p(v)$. את הקיבולת (s, v) נגדיר כ $p(v)$)

נסמן - $C = \sum_{p(x) > 0} p(x)$. אם A אוסף פרוייקטים חוקי, אז $c(\{s\} \cup A, X \setminus A \cup \{t\}) \leq C$.

יש לנו טעות כאן. טוב, בואו נעבור לדברים אחרים.

פולינומים מרוכבים

קיימת הצגה במקדמים: $p(x) = \sum_{i=0}^n a_i z^i$

בנוסף, נניח ויש לנו $x_1, \dots, x_{n+2}$ והפולינום p הוא פולינום מדרגה $n \geq$ כך ש $p(x_i) = y_i$.

מספיקות $n + 1$ נקודות בשביל לייצג פולינום מדרגה n . איך עוברים בין 'הצגת ערכים' להצגת מקדמים?

מהצגה למקדמים להצגה בערכים – פשוט להציב x_i שונים ב p . זמן ריצה – אפשר לכל x_i אפשר

לחשב את $p(x_i)$ בזמן $O(n)$. בסה"כ - $O(n^2)$.

מה עושים בכיוון השני?

מהצגה לערכים להצגה במקדמים – אינטרפולציית לגרנז'.

$$- \sum_i y_i \frac{\prod_{j \neq i} (z - x_j)}{\prod_{j \neq i} (x_i - x_j)} \text{ - פולינום } C. \text{ מעביר מהערכים אל המקדמים. זמן ריצה - } O(n^3).$$

27.11.2006

תרגול 6

ציטוט השיעור: 'לא הבנתם? טוב, בואו ננסה לרשום את האינדקסים של הפולינום בכתיב בינארי'

נתון פולינום $p(z) = \sum_{i=0}^{n-1} a_i z^i$. רוצים לחשב את הערכים של הפולינום הנ"ל בנקודות

$$\omega_n = e^{\frac{2\pi i}{n}} \text{ כאשר } \omega_n^0, \omega_n^1, \dots, \omega_n^{n-1}$$

Fast Fourier Transform – FFT מבוסס על:

נניח $n = 2^k$ נגדיר שני פולינומים $P_{\text{even}}, P_{\text{odd}}$

$$P_{\text{even}}(z) = \sum_{j=0}^{\frac{n}{2}-1} a_{2j} z^j$$

$$P_{\text{odd}}(z) = \sum_{j=0}^{\frac{n}{2}-1} a_{2j+1} z^j$$

נשים לב להבחנה - $p(z) = p_{\text{even}}(z^2) + z \cdot p_{\text{odd}}(z^2)$ בפרט:

$$p(\omega_n^t) = p_e(\omega_n^{2t}) + \omega_n^t p_o(\omega_n^{2t}) =$$

$$p_e(\omega_{n/2}^t) + \omega_n^t p_o(\omega_{n/2}^t) = P_e(\omega_{n/2}^t) + \omega_n^t P_o(\omega_{n/2}^t)$$

$$p(\omega_4^1) = P_e(\omega_2^1) + \omega_4^1 p_o(\omega_2^1) \text{ לדוגמא:}$$

דוגמא: $n = 8 = 2^3$. הפולינום הוא $p(z) = a_0 + \dots + a_7 z^7$

$$(a_0 + a_2 z + a_4 z^2 + a_6 z^3) \quad (a_1 + a_3 z + a_5 z^2 + a_7 z^3)$$

$$((a_0 + a_4 z) \quad (a_2 + a_6 z)) \quad ((a_1 + a_5 z) \quad (a_3 + a_7 z))$$

$$(a_0 \pm a_4) \quad (a_2 \pm a_6), \dots,$$

קצת התבלבלנו. ננסה לחשב את הערך של הפולינום $(a_0 + a_2 z + a_4 z^2 + a_6 z^3)$ ב 4 שורשי יחידה.

זה יהיה שווה ל:

$$a_0 + a_4 + a_2 + a_6$$

$$a_0 - a_4 + a_2 + a_6$$

$$a_0 + a_4 + a_2 - a_6$$

$$a_0 - a_4 + a_2 - a_6$$

טוב. לא הבנו. בואו ננסה לכתוב את זה בכתיב בינארי.

000	000
001	100
010	010
011	110
100	001
101	101
110	011
111	111

ננסה לחשב את ערכי הפולינום $a_{000} + a_{100}z$ ב $1, -1$.

טוב, בואו ננסה להציב ב $a_0 + a_2z + a_4z^2 + a_6z^3$.
עבור $z = \omega_4^0$, אזי $p(\omega_4^0) = p_e(\omega_2^0) + \omega_4^2 p_o(\omega_2^0)$.

איך מכפילים פולינומים?

נתונים שני פולינומים, p, q . $p = \sum_{i=0}^{n-1} a_i z^i, q = \sum_{i=0}^{n-1} b_i z^i$. אנו רוצים לחשב את המקדמים $p \cdot q$.
נבצע FFT בגודל $2n$ על p ($FT_{2n}(p)$) ועל q ($FT_{2n}(q)$). נכפיל את הוקטורים שקיבלנו.
ונקבל $FT_{2n}(q) \times FT_{2n}(p)$. על זה נפעיל FT_{2n}^{-1} ומתקיים:

$$pq = \frac{1}{n} FT_{2n}^{-1} (FT_{2n}(q) \times FT_{2n}(p))$$

ניזכר במה זה בעצם FFT:

$$\begin{pmatrix} p(\omega_n^0) \\ \dots \\ p(\omega_n^{n-1}) \end{pmatrix} = \underbrace{(\omega_n^{ij})}_{Vandermonda} \begin{pmatrix} a_0 \\ \dots \\ a_{n-1} \end{pmatrix}$$

נסתכל על המטריצה $U = (w_n^{-ij})$. $U^{-ij} = \overline{\omega_n^{ij}}$.

מתקיים $UV = nI$. ולכן יש לחלק ב n (שזה שקול ללהכפיל ב $\frac{1}{n}$).

למה זה עובד? pq הוא פולינום מדרגה $> 2n$, ו $2n$ נקודות מספיקות כדי לייצג אותו.

מה יקרה אם מחשבים את כפל הפולינומים עם FT מסדר n ? כמובן שלא בהכרח נקבל תשובה, כי לא מספיקות n קוארידנטות כדי לייצג פולינום מדרגה גדולה מ n .

$$f = \frac{1}{n} FT^{-1} (FT_n(p) \times FT_n(q))$$

טענה: $pq = h(x^n - 1) + f$ עבור h פולינום.

הוכחה: כמו ממקודם, $f(\omega_n^k) = p(\omega_n^k)p(\omega_n^k)$, לכל $0 \leq k \leq n$.
 $pq - f$ מתאפס ב ω_n^k לכל $0 \leq k \leq n$. לכן $h \cdot (x - \omega_n^0) \dots (x - \omega_n^{n-1})$.
 $pq - f = (x - \omega_n^0) \dots (x - \omega_n^{n-1}) \cdot h$.

4.12.2006

תרגול 7

הערה: כל המספרים בתרגול הם כחלק מ $\mathbb{Z}$, אלא אם הוזכר אחרת.

תורת המספרים המאוד אלמנטרית

אומרים ש k מחלק את n (ומסמנים $k | n$) עבור $k, n \in \mathbb{Z}$ אם קיים $m \in \mathbb{Z}$ כך ש $km = n$.

לדוגמא - $7 | 42$, $8 \nmid 15$.

טענות קלות - $d | a$ ו $d | b$ אז $d | xa + yb$ לכל $x, y \in \mathbb{Z}$.

הוכחה: $d | a$ קיים k כך ש $dk = a$

$d | b$ קיים k' כך ש $d \cdot k' = b$

ואז $xa + yb = xdk + ydk' = d(xk + yk')$

הגדרה: a, b , לא שניהם אפס, אזי $\gcd(a, b) = \max\{d, d | b \wedge d | a\}$ (greatest common

divider) – מחלק משותף מקסימלי.

הערה: לכל $d \in \mathbb{Z}$ מתקיים $d | 0$.

דוגמאות ל $\gcd$:

$$\gcd(14, 21) = 7$$

$$\gcd(0, 17) = 17$$

למען שלמות ההגדרה: $\gcd(0, 0) = 0$.

תכונות GCD

$$1. \gcd(a, b) = \gcd(b, a)$$

$$2. \gcd(-a, b) = \gcd(a, b)$$

$$3. \gcd(a, b) = \gcd(|a|, |b|)$$

$$4. \gcd(k, 0) = |k|$$

$$5. \gcd(a, ak) = |a|$$

חלוקה עם שארית

אם $a > 0, b > 0$ אז ניתן לרשום את $a = qb + r$ כאשר $0 \leq r < b$.

טענה: יהיו a, b לא שניהם אפס. נגדיר $D = \{xa + yb \mid x, y \in \mathbb{Z}\}$ אז

$$\gcd(a, b) = \min\{0 < d \in D\}$$

דוגמא: $a = 6, b = 28$. $\gcd(a, b) = 2$.

$$D = \{0, 6, 28, 22, 2, \dots\}$$

נשים לב שאת 2 אנו מקבלים עבור $5 \cdot 6 - 1 \cdot 28$.

הוכחה: נסמן $s := \min\{d \in D, d > 0\}$.

קיימים q, r כך ש $a = qs + r$ (חילוק עם שארית).

$$r = a - qs = a - q(xa + yb) = (1 - qx)a - q(yb)$$

כלומר $r \in D$. ידוע $0 \leq r < s$, ולכן $r = 0$, כלומר $s | a$.

באותה דרך נקבל $s | b$, ולכן $s \leq \gcd(a, b)$.

כיוון שני

$$s = xa + yb$$

$$\gcd(a, b) \mid a$$

$$\gcd(a, b) \mid b$$

$$s > 0 \text{ ופרט } \gcd(a, b) \leq s \text{ ועל כן } \gcd(a, b) \mid s$$

$$s = \gcd(a, b) \text{ בסה"כ, משני הכיוונים}$$

מסקנה: אפשר להציג את המחלק המשותף המקסימלי של a, b ע"י צירוף שלם של a, b .

הגדרה: $a, b \in \mathbb{Z}$ נקראים זרים אם $\gcd(a, b) = 1$.

טענה: אם $\gcd(a, n) = 1, \gcd(b, n) = 1$ אז $\gcd(ab, n) = 1$.

הוכחה: לפי הטענה הקודמת – קיימים x, y, x', y' כך ש $xa + yn = 1$ וגם $x'b + y'n = 1$.

$$xx'ab + (yx'b + xy'a + nyy')n = 1 \text{ נקבל, ונשווה את המשוואות,}$$

אלגוריתם לחישוב gcd

טענה: אם $b > 0$ אז $\gcd(a, b) = \gcd(b, a \bmod b)$ (זו השארית בחלוקה)

הוכחה: תהי D כמו מקודם. $a \bmod b = a - \lfloor a/b \rfloor b$, ולכן $b, a \bmod b \in D$. נגדיר

$$D' = \{xb + y(a \bmod b) \mid x, y \in \mathbb{Z}\}.$$

קבלנו $D' \subset D$. באופן דומה, $a = \lfloor a/b \rfloor b + a \bmod b$, ובאותו אופן $D \subset D'$, ועל כן $D = D'$.

אלגוריתם אוקלידס ($a, b \geq 0$)

(מחשב את GCD – נובע מהטענה הקודמת)

$$Euc(a, b)$$

if (b=0)

return a

else return Euc(b, a mod b)

תזכורת: מספר פיבונאצ'י: $f_0 = 0, f_1 = 1$. עבור $k \geq 2$ $f_k = f_{k-1} + f_{k-2}$

הסדרה נראית $0, 1, 1, 2, 3, 5, 7, 12, \dots$

טענה: אם $a > b \geq 1$ אז $Euc(a, b)$ מבצע k קריאות רקורסיביות אז $a \geq f_{k+2}$ ו $b \geq f_{k+1}$.

הוכחה: באינדוקציה על k .

בסיס: $k = 0, a, b \geq 1$ ולכן $a \geq f_{0+2} = 1, b \geq f_{0+1} = 1$

שלב האינדוקציה: נניח עבור $k-1$, ונוכיח עבור k . מחשבים את $Euc(a, b)$. בחישוב של

$$Euc(b, a \bmod b) \text{ נעשות } k-1 \text{ קריאות רקורסיביות. לפי הנחת האינדוקציה,}$$

$$a \bmod b \geq f_k$$

נתון $a > b$, ולכן

$$b \geq f_{k+1}$$

$$b + a \bmod b \geq f_k + f_{k+1} = f_{k+2} \text{ ומכאן הטענה.}$$

טענה: אם $a > b$, $b < f_{k+1}$, אזי בהפעלת $\text{Euc}(a,b)$ מתבצעות פחות מ- k קריאות רקורסיביות. נזכור

כי, $f_k = \Theta\left(\left(\frac{\sqrt{5}+1}{2}\right)^k\right)$, ועל כן חישוב $\text{gcd}(a,b)$ לוקח $O(\log(\max(a,b)))$.

נשים לב שמדובר בפעולות על מספרים (לפעמים לא תמיד נרצה להגיד שהן $O(1)$).

אלגוריתם אוקלידס מורחב

$$E - \text{Euc}(a,b) \rightarrow (d,x,y)$$

(כך ש $d = \text{gcd}(a,b) = xa + yb$)

$$\text{Euc}(a,b)$$

if (b=0)

return (a,1,0)

else

$$(d,x,y) \leftarrow \text{Euc}(b, a \bmod b)$$

$$\text{return } (d, y, x - \lfloor a/b \rfloor y)$$

שכן אם $d = xb + y(a \bmod b)$ אז

$$ya + \left(x + \lfloor a/b \rfloor y\right)b = xb - \underbrace{\left(a - \lfloor a/b \rfloor b\right)}_{=a \bmod b} y = d$$

הוכחת נכונות האלגוריתם

מניחים שהאלגוריתם עובד נכון על $b, a \bmod b$, ומהשיוון בשורה למעלה מקבלים נכונות עבור

$$\text{Euc}(a,b)$$

חבורה

קבוצה S עם פעולה $\cdot : S \times S \rightarrow S$ המקיימת:

- $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- קיים $e \in S$ כך ש $a \cdot e = e \cdot a = a$ $\forall a$.
- לכל $a \in S$ קיים $b \in S$ כך ש $a \cdot b = e$

דוגמאות:

- $\{\mathbb{Z}, +\}$ (פעולת חיבור בלבד)
 - $\{\mathbb{Z}_n, +_{\bmod n}\}$
 - יהי $n > 1$, נגדיר $\mathbb{Z}_n^* = \{1 \leq k \leq n \mid \text{gcd}(k,n) = 1\}$. החבורה כאן היא $\{\mathbb{Z}_n^*, \cdot_{\bmod n}\}$.
- לדוגמא: עבור $n = 9$, $\mathbb{Z}_9^* = \{1, 2, 4, 5, 7, 8\}$, כאן, $4 \cdot_{\bmod 9} 7 = 1$.

נוכיח ש $\mathbb{Z}_n^*$ חבורה:

- נראה שהקבוצה סגורה לכפל. נתון $\gcd(a, n) = 1, \gcd(b, n) = 1$ ועל כן $\gcd(a \cdot b, n) = 1$. $a \cdot b = a \cdot b \bmod n = ab - kn, k \in \mathbb{Z}$, ועל כן $\gcd(ab \bmod n, n) = 1$
- קיום הפכי - $a \in \mathbb{Z}_n^*$ - יודעים ש $\gcd(a, n) = 1$. קיימים x, y כך ש $xa + yn = 1 \Rightarrow xa = 1 \bmod n$ ולכן x הוא ההופכי של a .

עוד דוגמא לחבורה: אם F שדה, אז $F \setminus \{0\}$ חבורה ביחס לכפל.

אם G חבורה, ו $x \in G$, אז $\langle x \rangle := \left\{ \underbrace{x \cdot x \cdot \dots \cdot x}_{k \text{ times}} \right\}$, עבור k שלם, כאשר מוגדר $x^0 = e$. קוראים לזה 'תת חבורה (הציקלית) שנוצרת ע"י x '.
 חבורה G נקראת ציקלית אם קיים $x \in G$ כך ש $\langle x \rangle = G$.
 לדוגמא: $\mathbb{Z}_9^*$, עבור 2 , נקבל $\langle 2 \rangle = \langle 1, 2, 4, 8, 7, 5 \rangle$

11.12.2006

המשך תורת המספרים האלמנטרית

$$x \bmod 7 = 1$$

שאלה: האם קיים מספר $0 \leq x < 140$ כך ש $x \bmod 4 = 1$ ו $x \bmod 5 = 2$.

$$x \bmod 5 = 2$$

משפט השאריות הסיני

אם $n_1, \dots, n_k$ מספרים זרים בזוגות (כלומר, $\gcd(n_i, n_j) = 1$ לכל $i \neq j$). נסמן $n = \prod_{i=1}^k n_i$ אז

לכל $\{a_i\}_{i=1}^k$ $0 \leq a_i \leq n_i$ קיים יחיד x $0 \leq x \leq n$ כך ש $\forall i, x \bmod n_i = a_i$.

הוכחה:

$$m_i = \prod_{j \neq i} n_j$$

נגדיר $c_i = m_i \cdot (m_i^{-1} \bmod n_i) \bmod n$. למה קיים $m_i^{-1} \bmod n_i$ בהכרח? מאחר

$$\gcd(m_i, n_i) = 1$$

תזכורת: הופכי ב $\mathbb{Z}_n^*$ - אם $\gcd(k, n) = 1$ אז $xk + yn = 1 \Rightarrow x = k^{-1} \bmod n$

$$c_i \bmod n_i = m_i \cdot (m_i^{-1} \bmod n_i) \bmod n_i =$$

$$(m_i \bmod n_i)(m_i^{-1} \bmod n_i) \bmod n_i = 1$$

למה זה נכון: באופן כללי - $(a \cdot b) \bmod k = (a \bmod k)(b \bmod k) \bmod k$. נשים לב שהשתמשנו גם בכך ש n_i מחלק של n .

עבור $j \neq i$, $c_i \bmod n_j = 0$ - שכן $n_j \mid m_i$ ולכן גם את c_i .

$$c_i \bmod n_j = \delta_{ij} \text{ - קיבלנו כי }$$

$$x = \sum a_i c_i \bmod n$$

$$x \bmod n_i = \sum_{j \neq i} a_j c_j \bmod n_i + (a_i c_i) \bmod n_i = a_i$$

הראנו קיום. נותר להוכיח יחידות: מספר האפשרויות ל $\{a_i\}_{i=1}^k$ שונים הוא $n_1 \cdot n_2 \cdot \dots \cdot n_k = n$.

מספר ה'תשובות' לבעיה הוא גם n (כי אני צריך לתת מספר בין 0 ל $n-1$ (כולל)). בהנחה שהפונקציה בעיה $\leftarrow$ תשובה היא חח"ע, אזי הוכחנו יחידות, וברור כי היא חח"ע.

בואו נכתוב את זה קצת פורמלית:

לכל בחירה של $\{a_i\}_{i=1}^k$ כאשר $0 \leq a_i < n_i$ קיים x כך ש $x \bmod n_i = a_i$ לכל $1 \leq i \leq k$.

$0 \leq x < n$. מספר האוספים השונים של $\{a_i\}$ האלה הוא n . מספר ה x_i ים הוא n . לכל x יש

$\{a_i\}_{i=1}^k$ יחידים $x \bmod n_i = a_i$ ולכן לא יתכנו שני x ים שונים עם שאריות זהות מודולו כל n_i .

נשים לב כי הוכחת המשפט נותנת לנו את האלגוריתם לפתרון הבעיה.

חבורות

G חבורה, $x \in G$. החבורה הנותרת ע"י $\langle x \rangle = \{e, x^1, x^2, \dots, x^{-1}, x^{-2}, \dots\}$. נניח ש G חבורה סופית, $x \in G$. נסתכל על $x, x^2, \dots$. ולכן קיימים k, n כך ש $x^k = x^n$, ועל כן, $x^{n-k} = e$.

כלומר, לכל $x \in G$ קיים m מינימלי כך ש $x^m = e$. ל m מינימלי כך ש $x^m = e$ קוראים **הסדר** של x , ומסמנים $\text{ord}(x)$.

טענה: $\text{ord}(x) = |\langle x \rangle|$

הוכחה: נסמן $m = \text{ord}(x)$. נגדיר $H = \{e, x^1, x^2, \dots, x^{m-1}\}$. H היא **תת חבורה** שכן $x^k \cdot x^n = x^{k+n}$ אם $k+n < m$, אזי $k+n = m+l$ כאשר $0 \leq l < m$, ואז $x^{k+n} = x^m \cdot x^l = ex^l = x^l$.

אם $x^k \in H$ אז $x^{m-k} \in H$ ו $x^{m-k} x^k = x^m = e$ ולכן יש לנו גם הופכי. אם $|H| < m$ אז קיימים $0 \leq k < n < m$ כך ש $x^k = x^n$, ואז $x^{n-k} = e$ כאשר $n-k < m$, בסתירה להגדרת m .

ע"פ הגדרת H , $H \subset \langle x \rangle$, מצד שני, $x \in H$ ו H תת חבורה, ולכן $\langle x \rangle \subset H$. לכן - $H = \langle x \rangle$, ובפרט $|H| = |\langle x \rangle| = m$.

משפט לגרנז': אם G חבורה סופית, H תת חבורה (של G), אזי $|H| \mid |G|$. **הוכחה:** נגדיר יחס $\sim$ על G . $g \sim g'$ אם קיים $h \in H$ כך ש $g = hg'$. זהו יחס שקילות: $\forall g, g' \sim g$ 1.

2. אם $g \sim g'$ אז קיים h , $g = hg'$ ואז $g' = h^{-1}g$ (נשים לב - אנו בחבורה - לכל איבר יש הפכי), ועל כן $g' = h^{-1}g$, כלומר $g' \sim g$.

3. $g_1 \sim g_2, g_2 \sim g_3$. על כן $g_1 = h_1 g_2$, $g_2 = h_2 g_3$ $\Rightarrow g_1 = h_1 h_2 g_3$.

קיבלנו כי $\sim$ הוא יחס שקילות. על כן, G מתחלקת למחלקות שקילות. H היא מחלקת שקילות, כי $h, g \in H$, אזי $h = hg^{-1}g$ ומתקיים $hg^{-1} \in H$ אם $h \in H$ ו $g \notin H$ אז $\neg(g \sim h)$, כי אם $g \sim h$, אזי $g = h'h \in H$, בסתירה לכך ש $g \notin H$.

תהי S מחלקת שקילות של $\sim$. נראה כי $|S| = |H|$. יהי $g \in S$. נגדיר העתקה $\varphi: H \rightarrow S$ כך ש $\varphi(h) = g \cdot h$.

φ היא חח"ע $h \neq h' \Rightarrow hg \neq h'g$. היא על - כי אם $g' \in S$, כלומר $g' \sim g$, $g' = hg$ $g' = \varphi(h)$

כעת, מדוע $\varphi(h) \in S$? כי לכל h , $h \cdot g \sim g$ ועל כן S היא מחלקת שקילות. סה"כ φ היא העתקה חח"ע ועל $|H| = |S|$. מכאן - לכל מחלקות השקילות אותו גודל, ועל כן $|H| \mid |G|$ - מ.ש.ל.

טענה: אם G חבורה סופית, אז $x \in G$ אז $|G| \mid \text{ord}(x)$ (הסדר של x מחלק את $|G|$).

הוכחה: $\text{ord}(x) = |\langle x \rangle|$ ולפי משפט לגרנז' $|\langle x \rangle| \mid |G|$.

מסקנה: אם p ראשוני, אז $0 < k < p$, אז $k^{p-1} \equiv 1 \pmod p$.

הוכחה:

(הערה: כל השוויונות כאן הם $\pmod p$ או $\pmod n$, בהתאמה).

נתבונן ב $\mathbb{Z}_p^* = \{1, \dots, p-1\}$. ניקח $k \in \mathbb{Z}_p^*$. $k^{p-1} = k^{|\mathbb{Z}_p^*|} = k^{\text{ord}(k) \cdot m} = (k^{\text{ord}(k)})^m = 1^m = 1$.

באופן כללי, עבור $n > 0$, נסתכל ב $\mathbb{Z}_n^*$. סימון $0 \in \mathbb{Z}_n^*$ $\varphi(n)$ (פונקציית φ של אוילר).

לכל $x \in \mathbb{Z}_n^*$ נקבל $x^{|\mathbb{Z}_n^*|} = x^{\text{ord}(x) \cdot m} = 1$.

נחשב את $\varphi(n)$. פירוק לראשוניים של n :

$$n = p_1^{m_1} \cdot p_2^{m_2} \cdot \dots \cdot p_k^{m_k}$$

כאשר $\{p_i\}$ ראשוניים שונים.

נשים לב ש $\{p_i^{m_i}\}_{i=1}^k$ זרים בזוגות. $0 \leq x \leq n$, אז x זר ל n אם ו x זר ל p_i לכל i . $\Leftrightarrow$ לכל i

$x \pmod{p_i^{m_i}}$ זר ל p_i .

לכל i , מספר ה a_i כך ש $0 \leq a_i < p_i^{m_i}$ ו a_i זר ל p_i הוא $\left(1 - \frac{1}{p_i}\right) p_i^{m_i}$. לכל i .

$a_1, \dots, a_k$ קיים x כך ש $x \pmod{p_i^{m_i}} = a_i$.

$$\mathbb{Z}_n^* = \prod \left(1 - \frac{1}{p_i}\right) p_i^{m_i} = \prod \left(1 - \frac{1}{p_i}\right) n$$

18.12.2006

תרגול 9

n אנשים רוצים לגשת לבסיס נתונים. בכל יחידת זמן רק 1 יכול לגשת (אם שניים מנסים יחדיו אז מקבלים שגיאה).

אלג' הסתברותי – כל אחד בכל יחידת זמן בהסתפרות p מנסה לגשת לבסיס הנתונים.

$A_{i,t}$ - האיש i מנסה בזמן t .

$S_{i,t}$ - האיש i מצליח בזמן t . $S_{i,t} = A_{i,t} \cap \bigcap_{j \neq i} \overline{A_{j,t}}$ (כלומר, הוא מנסה לגשת, וכל השאר לא מנסים לגשת).

$$\Pr(S_{i,t}) = p \cdot (1-p)^{n-1}$$

נרצה למקסום את ההסתברות:

$$f(p) = p(1-p)^{n-1}$$

$$f'(p) = (1-p)^{n-1} - p(n-1)(1-p)^{n-2} = (1-p)^{n-2}(1-p-p(n-1)) =$$

$$(1-p)^{n-2}(1-np)$$

מאחר $0 \leq p \leq 1$. הנגזרת מתאפסת ב $p = \frac{1}{n}$, ולכן הסתברות 'הצלחה' מקסימלית עבור $p = \frac{1}{n}$.

תזכורת: $\left(1 - \frac{1}{n}\right)^n \leq \frac{1}{e} \leq \left(1 - \frac{1}{n}\right)^{n-1}$ (צד ימין יורד מונוטוני ל $\frac{1}{e}$, צד שמאל עולה מונוטוני ל $\frac{1}{e}$, עבור $n \rightarrow \infty$).

קיבלנו $\Pr(S_{i,t}) = \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{ne}$. מה הסיכוי שמישהו יצליח בזמן t ?

$$\Pr(S_{i,t}) = \sum_{i=1}^n \Pr(S_{i,t}) \geq \frac{1}{e}$$

$F_{i,T}$ - שחקן i נכשל בכל התורות מ 1 עד t .

$$\Pr(F_{i,T}) = \prod_{t=1}^T (1 - \Pr(S_{i,t})) \leq \left(1 - \frac{1}{ne}\right)^T$$

ניקח $T = \lceil ne \rceil$. $\Pr(F_{i, \lceil ne \rceil}) \leq \frac{1}{e}$.

אם $T = \lceil ne \rceil c \log n$ עבור c כלשהו, ואז $n^{-c} = \left(\frac{1}{e}\right)^{c \log n} = \left(1 - \frac{1}{en}\right)^{en \cdot c \log n} \leq \Pr(F_{i,T})$.

כמה זמן צריך לחכות עד שבהסתברות גדולה כולם הצליחו?

ההסתברות שמישהו נכשל היא: $\Pr\left(\bigcup_{i=1}^n F_{i,T}\right) \leq n \cdot \Pr(F_{i,T}) \leq n^{-c+1}$.

לסיכום: אם ניקח $T = 2en \log n$, אזי נקבל כי $\Pr(\text{everyone succeed till } T) \geq 1 - \frac{1}{n}$.

X משתנה מקרי. נסמן תוחלת של X היא $E(X) = \sum_x x \cdot \Pr(X = x)$

שוונות: $V(X) = E(X^2) - (E(X))^2$

סטיית תקן: $\sigma(X) = \sqrt{V(X)}$

אי שיוויון מרקוב: X משתנה מקרי חיובי, $\varepsilon > 0$, אז הסיכוי ש $\Pr(X \geq \varepsilon) \leq \frac{E(X)}{\varepsilon}$

אי שיוויון צ'בישב: X משתנה מקרי $\Pr(|X - E(X)| > \varepsilon) \leq \frac{V(X)}{\varepsilon^2}$

חוק המספרים הגדולים

$x_1, \dots, x_n$ משתנים מקריים בלתי תלויים.

הגדרה: x, y משתנים מקריים בלתי תלויים אם

$\forall x, \forall y (P(X = x) \wedge Y = y) = P(X = x)P(Y = y)$. נרחיב את ההגדרה להרבה משתנים.

טענה: $Var\left(\sum_{i=1}^n x_i\right) = \sum_{i=1}^n V(x_i)$

הוכחה: $E\left(\left(\sum_{i=1}^n x_i\right)^2\right) - \left[E\left(\sum_{i=1}^n x_i\right)\right]^2$

הערה: $E(X + Y) = E(X) + E(Y)$ אם X, Y בלתי תלויים אז

$E(X \cdot Y) = E(X) \cdot E(Y)$

נחזור להוכחה:

$\sum_i E(X_i^2) + \sum_{i \neq j} E(X_i X_j) - \left(\sum_i E(X_i^2) + \sum_{i \neq j} E(X_i)E(X_j)\right)E(X_j) =$

$\sum_i (x_i^2) - \left((E(x_i))^2\right) = \sum_{i=1}^n V(X_i)$

מ.ש.ל.

חוק המספרים הגדולים (הגרסה החלשה):

$x_1, \dots, x_n$ משתנים מקריים בלתי תלויים שווים התפלגות כך ש $\forall i, V(x_i) = V$ אזי

$$\Pr\left(\left|\frac{\sum_{i=1}^n x_i}{n} - E(X)\right| \geq \varepsilon\right) \leq \frac{V\left(\frac{\sum x_i}{n}\right)}{\varepsilon^2} = \frac{V\left(\sum_{i=1}^n x_i\right)}{\varepsilon^2 n^2} = \frac{nV}{\varepsilon^2 n^2} = \frac{V}{\varepsilon^2 n}$$

משמעות – ניקח ε קבוע. אם n הולך וגדל, אז הסיכוי שאני חורג מהתוחלת ביותר מ ε קטן ככל ש n גדל.

אי שיוויון צ'רנוב

אם $x_1, \dots, x_n$ משתנים מקריים בלתי תלויים $E(x_i) = 0$, $|X_i| \leq 1$, $\sigma = \sigma\left(\sum_{i=1}^n x_i\right)$, אזי

$$\Pr\left(\left|\sum x_i\right| > k\sigma\right) \leq 2e^{-k^2/2}$$

עבור $0 \leq k \leq 2\sigma$.

משמעות: נגיד שיש לנו משהו שאנו שואלים אותו שאלות, ואנו יודעים שהוא עונה נכון בהסתברות $p = \frac{2}{3}$. נשאל אותו n פעמים ונבחר את התשובה שמופיעה יותר פעמים. נחשב n 'טוב'.

נגדיר x_i -התשובה על השאלה בפעם ה i . נניח שהתשובה הנכונה היא 0. $\Pr\left(\sum_{i=1}^n X_i \geq \frac{n}{2}\right)$ הוא הסיכוי שבחר 1 יותר פעמים, כלומר תשובה לא נכונה.

$$Y_i = X_i - E(X_i) = X_i - (1-p)$$

$$V\left(\sum_{i=1}^n Y_i\right) = V\left(\sum x_i\right) = \sum V(x_i) = n \cdot p \cdot (1-p) \quad \text{נגדיר}$$

$$P\left(\sum x_i \geq \frac{n}{2}\right) = P\left(\sum y_i \geq n\left(\frac{1}{2} - (1-p)\right)\right) \leq P\left(\left|\sum Y_i\right| \geq \left(\frac{1}{2} - (1-p)\right)n\right) \leq 2e^{\frac{-\frac{1}{2}(1-p)^2 n^2}{2np(1-p)}} = 2e^{\frac{-\left(p-\frac{1}{2}\right)^2}{p(1-p)}n}$$

25.12.2006

אלגוריתמים הסתברותיים, המשך

טענה: (ללא הוכחה) – אי שיוויון צרנוב: יהו $X_1, \dots, X_n$ משתנים מקריים בלתי תלויים המקבלים

ערכים 0 ו-1. יהי $\mu = E\left(\sum_{i=1}^n x_i\right)$, אזי לכל $0 \leq \delta \leq 1$, $\Pr\left(\left(\sum_{i=1}^n x_i - \mu\right) > \delta \mu\right) \leq 2e^{-\frac{\delta^2 \mu}{3}}$.

וגם עבור $R \geq 8$, $\Pr\left(\sum x_i > R\mu\right) \leq 2^{-R}$.

בעיית חלוקת משימות

יש m משימות ו- m מעבדים, רוצים לחלק את המשימות בין המעבדים, באופן 'שיוויוני'. אם עושים את זה

באופן מרוכז, מקבלים לכל מעבד $\left\lceil \frac{m}{n} \right\rceil$ משימות. רוצים לעשות זאת באופן מבוזר.

אלגוריתם הסתברותי: כל משימה מוקצית בהסתברות שווה לכל מעבד. נראה שבסיכוי כבוע הפיזור שיוויוני.

נגדיר $Y_{ij}, 1 \leq i \leq n, 1 \leq j \leq m$

$Y_{ij} = \begin{cases} 1, & \text{if mission } j \text{ goes to processor } i \\ 0, & \text{otherwise} \end{cases}$ כאשר

$X_i = \sum_{j=1}^m Y_{ij}$ - מספר המשימות שמגיעות למעבד i .

$$E(x_i) = \sum_{j=1}^m E(Y_{ij}) = \frac{m}{n}$$

מאי שיוויון צרנוב נובע, עבור $\delta = \frac{1}{2}$:

$$\Pr\left(\left|\sum_{j=1}^m Y_{ij} - \frac{m}{n}\right| > \frac{1}{2} \frac{m}{n}\right) \leq 2e^{-\frac{1}{12} \frac{m}{n}}$$

ניתן לרשום זאת גם כך:

$$\Pr\left(\left(X_i \leq \frac{1}{2} \frac{m}{n}\right) \vee \left(X_i \geq \frac{3}{2} \frac{m}{n}\right)\right) \leq 2e^{-\frac{1}{12} \frac{m}{n}}$$

אם ניקח $m = cn \log n$, אזי נקבל

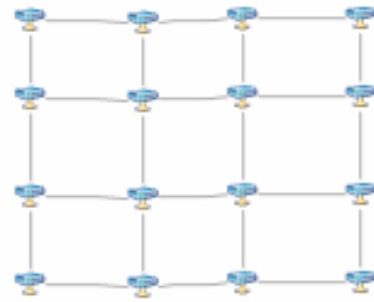
$$\Pr\left(\left(X_i \leq \frac{1}{2} \frac{m}{n}\right) \vee \left(X_i \geq \frac{3}{2} \frac{m}{n}\right)\right) \leq 2e^{-\frac{c}{12} \log n} = 2n^{-c/12}$$

$$\Pr\left(\exists i, \left(X_i \leq \frac{1}{2} \frac{m}{n}\right) \vee \left(X_i \geq \frac{3}{2} \frac{m}{n}\right)\right) \leq n \cdot 2n^{-c/12} = 2n^{-c/12+1}$$

תוצאה: עבור $m \geq 24 \cdot n \log n$, $\Pr\left(\exists i, x \geq \frac{3}{2} \frac{m}{n} \vee x \leq \frac{1}{2} \frac{m}{n}\right) \leq \frac{2}{n}$

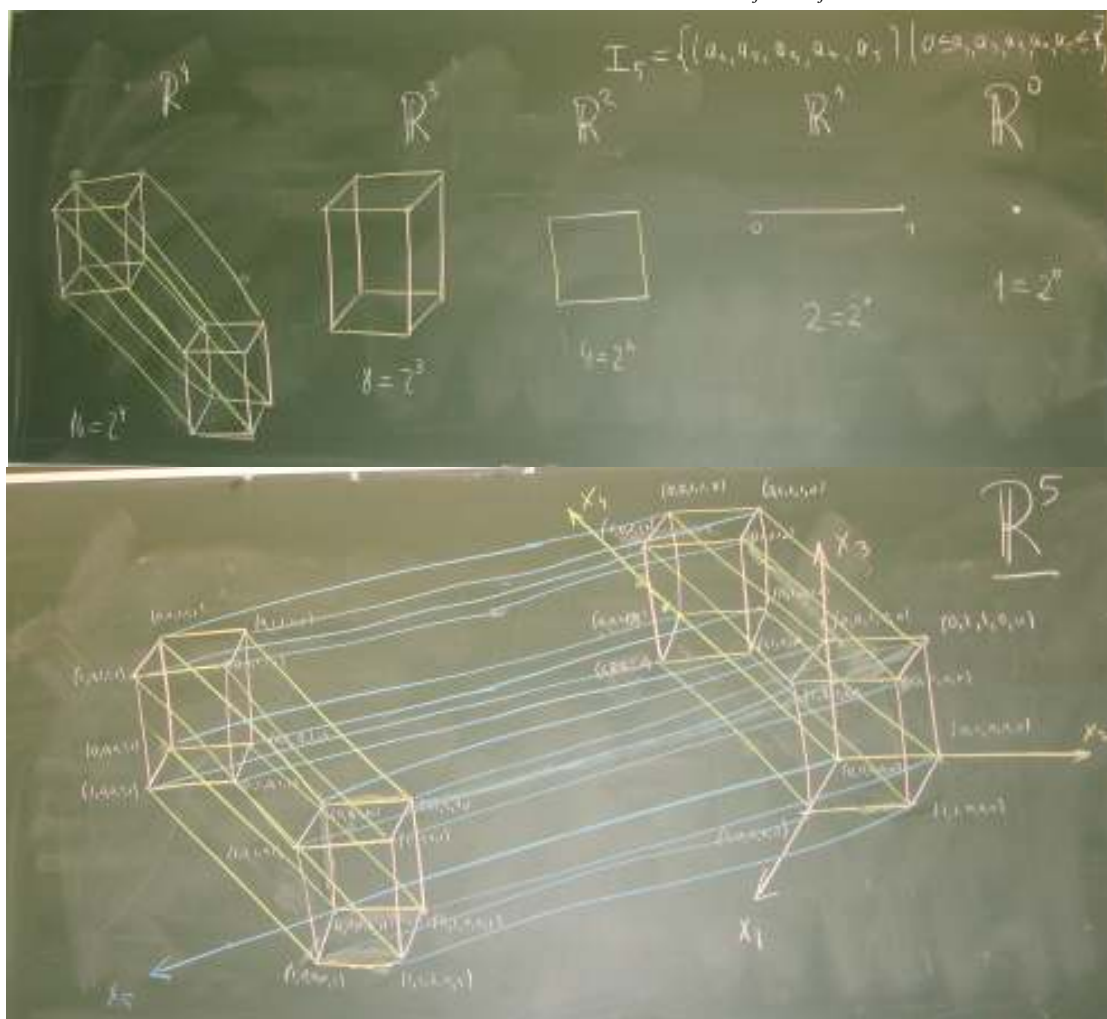
בעיה – ניתוב ברשתות

נתון גרף $G = (V, E)$



פונקציית ניתוב: פונקציה f כך ש $f(u, v)$ זאת מסילה מ u ל v . דוגמא לניתוב בגרף הנ"ל מ u ל v - הולכים אופקית ואז אנכית. נגדיר מראש יחס סדר $<$ על קודקודים אם שתי הודעות שבאות מקודקודים u, v מגיעות לקודקוד w ו'רוצות' להמשיך באותו כיוון, אז אם $u < v$ ההודעה מ u מחכה עד שההודעה מ v תישלח.

גרף הקובייה: Hyper Cube, $V = \{0, 1\}^n$, $v = (a_1, \dots, a_n)$ מחובר ל $u = (b_1, \dots, b_n)$ אם קיים i כך ש $a_i \neq b_i$ ו $a_j = b_j, i \neq j$. לדוגמא:



מתקיים (כמו שרשום על הלוח) $|V| = 2^n$ (n הוא המימד), והדרגה של כל קודקוד היא n .

ניתוב בקוביה:

מקודקוד $u = (a_1, \dots, a_n)$ ל $v = (b_1, \dots, b_n)$ כל שינוי של קוארדינטה אחת הוא 'הליכה בצלע קיימת' – ועל כן נשנה את כל הקוארדינטות השונות של u לקוארדינטות של v אחת-אחת. נראה דוגמא שבה ניתוב זה לוקח 'הרבה' זמן. כל קודקוד $v = (a_1, \dots, a_n)$ רוצה לשלוח הודעה לקודקוד $\sigma(v) = (a_n, \dots, a_1)$, בניתוב לעיל. יהי $w = (0, 0, 0, \dots, 0)$. כל קודקוד מהצורה $(a_1, \dots, a_{n/2}, 0, 0, 0, \dots, 0)$ עובר דרך w . מספר הקודקודים האלה הוא $2^{n/2}$.

קודקוד w בכל יחידת זמן שולח לכל היותר n הודעות ולכן זמן הניתוב $\leq \frac{2^{n/2}}{4}$.

טענה (ללא הוכחה): לכל ניתוב קיימת תמורה σ (שתגדיר ממי למי שולחים) כך שזמן הניתוב

$$\leq \frac{2^{n/2}}{\sqrt{n}}.$$

אלגוריתם הסתברותי של L. Valiant

כל קודקוד v מגריל באקראי קודקוד $g(v)$, ואז הניתוב הוא $v \rightarrow g(v) \rightarrow \sigma(v)$ (כאשר $\rightarrow$ משמעו הליכה בניתוב הרגיל שהגדרנו קודם).

ניתוח

בחירת $g: V \rightarrow V$ קובעת בעצם את הניתוב. רוצים להראות שלרוב פונקציות g ההודעות לא מחכות הרבה. נתבונן ב $X =$ קבוצת כל הפונקציות g כך שקיים קודקוד v שההודעה שהוא שולח מחכה cn פעמים.

כמה ביטים צריך כדי לתאר פונקציה $g \in X$? אם הודעה מחכה להודעה אחרת, אז למסלולים שלהם צלע משותפת. (נשים לב שצריך גם באותו זמן, אבל לא טענו לאדם). רוצים לתאר את הפונ' g :

בשביל לתאר את $v, g(v)$ צריך $n-1$ ביטים (כי יש קודקוד שבוודאות עוברים דרכו). לכן צריך $n + n + cn \cdot (n-1)$ ביטים.

בסה"כ, אנחנו מתארים cn מסלולים שיש להם צלע משותפת עם המסלול מ v ל $g(v)$. לכל i יש k_i מסלולים שהצלע המשותפת היא הצלע ה i ית.

1.1.2007

שיטת ההצפנה של רבין (מיכאל אוזר רבין)

שיטת ההצפנה א-סימטרית

לAlice יש מפתח פרטי, לכולם, ובפרט לBob, יש מפתח ציבורי.

Bob שולח הודעה C , שהיא הגרסה המוצפנת של M לAlice. Alice יכולה לשחזר את M באמצעות המפתח הפרטי שלה.

מפתח פרטי - p, q ראשוניים, כך ש $p \equiv q \equiv 3 \pmod{4}$. מפתח ציבורי - $n = p \cdot q$.

הצפנה: הודעה היא M כאשר $0 < M < n$. $C = M^2 \pmod{n}$.

פענוח: נתונים C, p, q . נסמן

$$m_p = C^{\frac{p+1}{4}} \pmod{p}$$

$$m_q = C^{\frac{q+1}{4}} \pmod{q}$$

$$m_p^2 = C^{\frac{p+1}{2}} \pmod{p} = M^{p+1} \pmod{p} =_{a^p \equiv a \pmod{n}} M^2 \pmod{p}$$

$$m_q^2 = M^2 \pmod{q}$$

הערה: אם F שדה למשוואה $x^2 = y$ יש לכל היותר שני פתרונות.

אם $x^2 \equiv c \pmod{n}$ אז $x^2 \equiv C \pmod{p}, x^2 \equiv C \pmod{q}$.

$$x^2 \equiv m_p^2 \pmod{p}, x^2 \equiv m_q^2 \pmod{q}$$

ועל כן $x \equiv m_q \pmod{q}$ או $x \equiv -m_q \pmod{q}$. כנ"ל בנוגע ל p - $x \equiv m_p \pmod{p}$ או $x \equiv -m_p \pmod{p}$.

נחשב ע"פ משפט השאריות הסיני את האופציות, בהנתן p, q, m_p, m_q את x_1 :

$$x_1 \equiv m_p \pmod{p}, x_1 \equiv m_q \pmod{q}$$

$$x_2 \equiv -m_p \pmod{p}, x_2 \equiv m_q \pmod{q}$$

$$x_3 \equiv m_p \pmod{p}, x_3 \equiv -m_q \pmod{q}$$

$$x_4 \equiv -m_p \pmod{p}, x_4 \equiv -m_q \pmod{q}$$

אלו כל המספרים שמקיימים $x_i^2 \equiv C \pmod{n}$.

דוגמא: יהי $p = 7, q = 11 \Rightarrow n = 77$. ההודעה $M = 10$. ההצפנה -

$$C = M^2 \pmod{77} = 100 \pmod{77} = 23$$

פענוח:

$$m_p = C^{\frac{7+1}{4}} \pmod{7} = 4$$

$$m_q = C^{\frac{11+1}{4}} \pmod{11} = 1$$

נשתמש במשפט השאריות הסיני, ונקבל את ארבעת האופציות הבאות:

$$a_p = 22, a_q = 56$$

$$x_1 = (4 \cdot 22 + 1 \cdot 56) \bmod 77 = 67$$

$$x_2 = (56 - 88) \bmod 77 = 45$$

$$x_3 = 32$$

$$x_4 = 10$$

כפי ששמנו לב, הרביעי הוא אכן המספר שהתחלנו איתו.

הערה: אם בוחרים הודעה $0 < M < \frac{n}{2}$ ובפענוח בוחרים את שתי התשובות $x_i < \frac{n}{2}$ שקטנות מ $\frac{n}{2}$.

מערכת 'קשה' לשבירה

טענה: בהנתן אלגוריתם A שמקיים $\left[A \left(\underset{=M^2}{C} \right) \right]^2 \equiv C \bmod n$ אזי אפשר לחשב את p, q .

(מכך, בהנחה שפירוק לגורמים היא בעיה קשה, נובע שיטת הצפנה זו קשה).

הוכחה: ניקח מספר מ $\{0, \dots, n-1\}$ ונעלה אותו בריבוע. יש 4 מספרים במקור שהולכים לכל מספר בטווח (שכן עובדים מודולו n). אלגוריתם ששובר את ההצפנה מחזיר אותי למקור.

למה: אם M, M' שתי הודעות כך ש $M \neq M', M \neq -M'$, אבל $M^2 \equiv (M')^2 \bmod n$ אזי $\gcd(M + M', n) > 1$.

הוכחה:

$$M = m_p \bmod p, M = m_q \bmod q$$

$$M' = \begin{cases} -m_p \bmod p, m_q \bmod q \\ OR \\ m_p \bmod p, -m_q \bmod q \end{cases}$$

במקרה הראשון, $M + M' \equiv 0 \bmod p$, $M + M' \not\equiv 0 \bmod q$, ולכן

$$\gcd(M + M', n) = 0. \text{ במקרה ב' באופן דומה נקבל } \gcd(M + M', n) = q.$$

המשך הוכחת הטענה

אלגוריתם לחישוב p, q - בחר M מקרי. מחשבים $M' = A(M^2)$. אם $M' \neq M, M' \neq -M$ נחשב את $\gcd(M, M')$ והצלחנו. אם לא, ננסה שוב.

אלגוריתם מצליח בסיכוי $\frac{1}{2}$, ולכן זה מספיק טוב כדי לפרק את n .

פרוטוקול לחישוב סכום בלי דליפת מידע

$$\text{יהי } N \text{ שבטוח } \sum_{i=1}^n x_i < N$$

שחקן 1 מגריל מספר r אקראי $0 \leq r < N$. מוסיף את המשכורת שלו למספר הזה מודולו N , ומעביר לשחקן 2. הוא מוסיף למספר זה את המשכורת שלו (גם מודולו N), ומעביר הלאה. כך עד שזה חוזר לשחקן 1. הוא מחסיר את r , ויש לו את הסכום.

למה אין דליפת מידע?

יהי $i \neq 1$. i מקבל את $y_i := r + \sum_{j=1}^{i-1} x_j \bmod N$. מתפלג אחיד על $0, \dots, N-1$ אז גם y_i . מתפלג אחיד על $0, \dots, N-1$. עבור $i = 1$ - הוא מקבל את הסכום של כולם, אבל הוא לא יודע שום דבר חוץ מזה.

למה צריך את המודולו N ? אחרת, אם חוזרים על הפעולה הרבה מאוד פעמים, השחקן השני יכול לדוגמא למצוא חסם מלמטה על x_1 . אם עושים מודולו, הוא לא יודע את x_1 - הוא לא יכול לדעת אם לא התבצע 'Overflow'.

כעת, נחפש פרוטוקול שבו כל קבוצה של $n-1 \geq$ שחקנים לא תגלה כלום חוץ מהסכום של כולם.

פרוטוקול

נמצא N כמו קודם. כל שחקן מגריל $y_1^i, \dots, y_n^i$ כך ש $\sum_{j=1}^n y_j^i = x_i \bmod N$. כל שחקן i שולח

לשחקן j את y_j^i . כל שחקן j מחשב את הסכום ששלחו לו, $z_i = \sum_{i=1}^n y_j^i$.

בסוף מחשבים את $\sum_{j=1}^n z_j \bmod N = \sum_{i=1}^n x_i$.

8.1.2007

נזכיר שאלגוריתם מקסימיזציה נקרא C ($0 < C$) מקרב אם לכל מופע של הבעיה הוא מחזיר פתרון O שמקיים $q(o) \leq q(opt) \leq cq(0)$.

הגדרה:

1. סכמת אפרוקסימציה (Approximation scheme) עבור בעיית אופטימיזציה היא אלגוריתם קירוב שמקבל בנוסף למופע של הבעיה גם $\varepsilon > 0$ ומחזיר שהוא $1 + \varepsilon$ מקרב.
2. נאמר שסכמה אפרוקסימציה היא Ptas (Polynomial time approximation scheme) אם לכל $\varepsilon > 0$, הסכמה רצה בזמן פולינומיאלי ב n .
3. נאמר שסכמה אפרוקסימציה היא FPTAS (Fully ptas) אם זמן הריצה פולינומיאלי ב $n, \frac{1}{\varepsilon}$.

דוגמא - $O(n^{1/\varepsilon})$ - היא Ptas, אבל לא Fptas.

Subset-Sum

הבעיה: נתונים n מספרים $0 < x_1, \dots, x_n \in \mathbb{N}$ ומספר $t > 0$. אנו מחפשים את תת הקבוצה

$$S' \subseteq \{x_1, \dots, x_n\} \text{ כך ש } \sum_{x \in S'} x \leq t \text{ מקסימלי תחת האילוץ ש } \sum_{x \in S'} x \leq t.$$

פתרון מדויק ואקספוננציאלי

אם נסמן ב L_i את הסכומים החלקיים האפשריים מ $\{x_1, \dots, x_i\}$ אז $L_{i+1} = L_i \cup (L_i + x_{i+1})$. נשם לב שאם נתקלנו באחד ה L_i ים במספר גדול מ t אפשר לזרוק אותו

Exact-Subset-Sum(S,t)

order S from the bigger to the smaller

$$n \leftarrow |S|$$

$$L_0 \leftarrow \{0\}$$

for $i = 1 \dots n$

$$L_i \leftarrow \text{Merge}(L_i, L_{i-1} + x_i)$$

remove from L_i all elements that exceed t

return $\max \{L_n\}$

נשים לב שהאלגוריתם הנ"ל פולינומיאלי בגודל הרשימות L_i ($1 \leq i \leq n$), ולכן אם אורך הרשימות יהיו תמיד חסומות ע"י ביטוי פולינומיאלי בגודל הקלט, אז האלגוריתם ירוץ בזמן פולינומיאלי. אבל עבור קבוצה של n מספרים עשויים להיות 2^n סכומים חלקיים.

נציג PTAS. הרעיון הוא במקום להחזיר את L_i נחזיק 'קירוב' שלה, ו'גסות' הקירוב תקבע ע"י רמות הדיוק הנדרשת - ε . הקירוב שנשתמש בו הוא 'Trimming' ('קיצוץ', כמו לקזז שכר). נרצה להחזיק רשימה שהיא תת רשימה של L_i שמקיימת:

$$1. \text{ לכל איבר } z \in L_i \text{ קיים } y \in L'_i \text{ המקיים } \frac{z}{1+\delta} \leq y \leq z \text{ עבור } \delta > 0 \text{ קבוע}$$

$$2. \text{ אורך הרשימה יהיה פולינומיאלי באורך הקלט ו } \frac{1}{\varepsilon}.$$

נשתמש באלגוריתם הבא:

בהינתן רשימה $L = (y_1, \dots, y_m)$ ממוינת וללא חזרות, נגדיר:

```

Trim( $L, \delta$ )
 $L' \leftarrow \{y_1\}$ 
last  $\leftarrow y_1$ 
for  $i \leftarrow 2$  to  $m$ 
    if  $(y_i > last \cdot (1 + \delta)) \{$ 
        append  $y_i$  to the end of  $L'$ 
        last  $\leftarrow y_i$ 
    }
return  $L'$ 

```

דוגמא: עבור $\delta = 0.1$, $L = \langle 10, 11, 12, 15, 20, 21, 22, 23, 24, 29 \rangle$, האלגוריתם יחזיר -

$$L' = \langle 10, \cancel{11}, 12, 15, 20, \cancel{21}, \cancel{22}, 23, \cancel{24}, 29 \rangle = \langle 10, 12, 15, 20, 23, 29 \rangle$$

נשים לב שזמן הריצה של Trim הוא $\Theta(|L|)$.

Subset-Sum קירוב אלגוריתם

Approx-Subset-Sum(S, t, ε)

```

 $n \leftarrow |S|$ 
 $L_0 \leftarrow \langle 0 \rangle$ 
for  $i = 1 \dots n$ 
     $L_i \leftarrow Merge(L_{i-1}, L_{i-1} + x_i)$ 
    Trim  $\left( L_i, \frac{\varepsilon}{2n} \right)$ 
    remove from  $L_i$  elements greater than  $t$ 
return  $\max \{L_n\}$ 

```

נשים לב, שבExact-subset-sum, L_i ביטא את אוסף כל הסכומים החלקיים של $x_1, \dots, x_n$ שלא חורגים מ- t – וזה עשוי להיות גדול (2^i) - L_n היא קבוצת כל הסכומים החלקיים של $x_1, \dots, x_n$ שלא חורגים מ- t . מההגדרה – אנחנו מחפשים $\max \{L_n\}$.

טענה: Approx-Subset-Sum הוא FPTAS.

הוכחה: 1. נראה שלכל $0 < \varepsilon < 1$ האלגוריתם מחזיר $1 + \varepsilon$ קירוב. באינדוקציה על i ניתן להראות שלכל איבר $y \in P_i$ יהיה L_i מאלגוריתם Exact-Subset-Sum, כלומר אוסף הסכומים החלקיים של $\{x_1, \dots, x_i\}$ שלא חורגים מ- t קיים $z \in L_i$ (כאן הוא מהאלגוריתם הנוכחי – עם Trim) כך

$$y \cdot \frac{\varepsilon}{1 + \frac{\varepsilon}{2n}} \leq z \leq y$$

: נוכח באינדוקציה

שלב: נניח עבור $i-1$, נוכיח עבור i . יהי $y \in P_i$. עבור i , מהנחת האינדוקציה קיים

אזי אם $y \in P_{i-1}$, אז סיימנו. אחרת, $y \in P_{i-1} + x_i$. כעת, קיים $z \in L_i$ כך ש $y' \leq z$ (כאשר y' הוא הנציג של y ב L_i).¹

באינדוקציה על i - לכל $y \in P_i$ קיים $z \in L_n$ כך ש $z \leq y$. $\frac{y}{\left(1+\varepsilon/2n\right)^i} \leq z$. בפרט עבור

$$\max(L_n) \text{ אזי וודאי כי } \frac{OPT}{1 + \left(\frac{\varepsilon}{2n}\right)^n} \leq Z^x \text{ (כי } z^x \geq z \text{ לכל } z \in L_n)$$

אפשר לראות זאת - $e^{\varepsilon/2} \leq \left(1 + \varepsilon/2n\right)^n \leq \left(1 + \frac{1}{x_n}\right)^{x_n} \xrightarrow{\infty} e$ נשתמש בכך ש $f(x) = e^x$

עבור $0 < \xi < \varepsilon/2$ נכפיל ב $\varepsilon/2$ את שני

$$\frac{e^{\varepsilon/2} - e^0}{\varepsilon/2} = f'(\xi) = e^\xi \leq_{0 < \xi < 1} e^{1/2}$$

3. זמן ריצה: נרצה להראות ש L_i לכל i הוא באורך פולינומיאלי ב $\frac{1}{\epsilon}, n, Ln(t)$. אורך

נשים לב שהיחס בין איברים עוקבים ב L_i הוא לכל הפחות $1 + \frac{\varepsilon}{2n}$ (חוץ מל 0) – כי אחרת,

¹ כאן קצת הסתבכנו. נשלים בהזדמנות (במבחן לדוגמא) [א.ש.].

ולכן

$$t \geq a_r > a_1 (1 + \delta)^{r-i}$$

ולכן

$$r-1+\log_{(1+\varepsilon/2n)}a_1<\log_{(1+\varepsilon/2n)}t\Rightarrow r<C+\log_{1+\varepsilon/2n}t$$

$$r < c + \frac{\ln t}{\ln\left(1+\varepsilon/2n\right)} \stackrel{\substack{\text{will be proved that} \\ \ln(1+x) \geq \frac{x}{1+x}, \text{ for } x > -1}}{\leq} C + \frac{\ln t}{1+\varepsilon/2n} \left(2+\varepsilon/2n\right) \leq C + \ln(t) \left(2+\varepsilon/2n\right) =$$

$$\frac{\ln t}{\frac{1}{1+2n/\varepsilon}} = \ln t \left(1 + \frac{2n}{\varepsilon}\right) = \ln t + \frac{2n}{\varepsilon} \ln(t)$$

ההוכחה שהובטחה באמצע:

$$f(x)=\ln(1+x)-\frac{x}{1+x}\geq 0 \text{ טענה:}$$

$$\text{הוכחה: נגזור, ונקבל } f'(x)=\frac{1}{1+x}-\frac{1}{1+x}-x\frac{-1}{(1+x)^2}=\frac{x}{(1+x)^2} \text{ זה מקבל}$$

$$\text{מינימום גלובלי ב} 0. \text{ ולכן } -\ln(1+x) \geq \frac{x}{1+x}$$

15.1.2007

עיבוד מקבילי

המודל: קיימים מספר כלשהו של מעבדים זהים, מסונכרנים (כלומר הזמן מחולק ליחידות דיסקרטיות והמעבדים מבצעים / לא מבצעים פעולה בכל יחידת זמן), בעלי זכרון משותף – Parallel (PRAM Random access memory)

נניח Concurrent Read (כל מספר של מעבדים יכול לקרוא מאותו תא של זכרון).

נניח בנוסף אחד (ונציין מה משתמשים)

1. Concurrent Write – מותר למספר כלשהו של מעבדים לכתוב לתא מסוים בזכרון באותו

תור. רק זה בעל ה-id הנמוך מצליח לכתוב. נקרא (CRCW – concurrent read, concurrent write)

2. Exclusive write – רק מעבד אחד יכול לכתוב לכל תא בכל תור. אם שניים מנסים לכתוב,

המחשב מתפוצץ. (CREW)

תאור אלגוריתם מקבילי: נתאר באופן הבא – בכל תור, ניתן לרשום מספר כלשהו של פעולות בו-זמניות (Concurrent) שעשויות להיות פעולות חישוב, קריאה וכתובה (תחת המגבלות EW או CW) ל-PRAM. יהיו חשובים לנו שני הממדים:

1. $T(n)$ מספר התורות שלוקח לחשב קלט בגודל n .

2. $W(n)$ מספר הפעולות הכולל בכל התורות שבצעו כל המעבדים.

עקרון WT: אם אלגוריתם מקבילי מבצע סה"כ w פעולות בז תורות, אז עבור p מעבדים הוא יצליח

לבצע את החישוב ב $t + \frac{w}{p}$ תורות לכל היותר.

הוכחה: נניח בתור ה- i מבוצעות w_i פעולות. $(1 \leq i \leq t)$, אז ניתן לסמלץ את פעולת האלגוריתם עם

p מעבדים ב $\frac{\lceil w_i \rceil}{p}$ תורות. סה"כ נקבל שאפשר להריץ את האלגוריתם

$$\text{ב } \sum_{i=1}^t \frac{\lceil w_i \rceil}{p} \leq \sum_{i=1}^t \left(\frac{w_i}{p} + 1 \right) = \sum_p \frac{w_i}{p} + t \text{ תורות, כנדרש.}$$

דוגמא: במודל CRCW ניתן לבצע n -bitwise OR, n -bitwise AND ב $O(1)$.

לדוגמא OR:

תור 1: מעבד אחד כותב 0 לתא הפלט.

תור 2: לכל אחד מ- n הביטים אם יש בו 1 כתוב בתא הפלט 1.

AND:

תור 1: מעבד אחד כותב 1 לתא הפלט.

תור 2: לכל ביט, אם יש בו 0 כתוב לתא הפלט 0.

Prefix Sums

הבעיה: נתונים n איברים $\{x_1, \dots, x_n\}$ מתוך קבוצה S , ומוגדרת על S פעולה אסוציאטיבית שנסמנה

*. אנו מחפשים את $\{S_1, \dots, S_n\}$ כך שלכל $1 \leq i \leq n$ מתקיים $S_i = x_1 * x_2 * \dots * x_i$. נניח

$n = 2^w$ עבור $w \in \mathbb{N}$.

רעיון ראשון (גרוע):

ניתן לפתור באופן רקורסיבי (Divide and Conquer) ע"י חישוב הרישיות של $x_1, \dots, x_{n/2}$ ונקבל

$$s_1, \dots, s_{n/2}, s_{n/2}' * s_1', \dots, s_{n/2}' * s_{n/2}' \text{ ואז פולט את } (s_1', \dots, s_{n/2}') \text{ ו } (s_1, \dots, s_{n/2})$$

ניתוח: $T(n)$ - ניתן לבצע את חישוב הרישיות של $\{x_1, \dots, x_{n/2}\}$ ו $x_{n/2+1}, \dots, x_n$ במקביל, ואז נקבל

$$T(n) = T\left(\frac{n}{2}\right) + O(1) = O(\log n) \text{ (תורות, לא מעבדים - מניחים } \frac{n}{2} \text{ מעבדים).}$$

$$W(n) = 2W\left(\frac{n}{2}\right) + O(n) = O(n \log n)$$

אמנם קיצרנו משמעותית את הזמן, אבל הגדלנו (ע"י הכפלה ב $\log n$) את מספר הפעולות הכולל שנבצע.

רעיון שני (טוב יותר)

יש לנו $x_1, x_2, \dots, x_n$. ניקח כל זוג $(x_1, x_2), (x_3, x_4), \dots, (x_{n-1}, x_n)$ ונבצע עליו את הפעולה -

$$y_1 = x_1 * x_2$$

....

$$y_{n/2} = x_{n/2}$$

כעת נגדיר:

$$s_2 = y_1 = x_1 * x_2$$

$$s_4 = y_1 * y_2 = x_1 * x_2 * x_3 * x_4$$

...

$$s_n = y_1 * \dots * y_{n/2} = x_1 * \dots * x_n$$

קיבלנו את כל הרישיות הזוגיות. כעת נחשב את האי זוגיות:

$$s_3 = s_1 * x_3$$

$$s_5 = s_2 * x_5$$

...

$$s_{n-1} = s_{n/2-1} * x_{n-1}$$

(יכול להיות שהתבלבלנו, בואו נרשום פסאודו קוד): (האינדקס משמאל הוא התור)

1. if $(n = 1)$

a. $s_1 \leftarrow x_1$

b. exit

2. for $1 \leq i \leq n/2$

a. $y_i \leftarrow x_{2i-1} * x_{2i}$

$3, \dots, 2 + T\left(\frac{n}{2}\right)$ - recursively compute prefix sums of $y_1, \dots, y_{n/2}$ and store them in $s'_1, \dots, s'_{n/2}$

$T\left(\frac{n}{2}\right) + 3$ - for $1 \leq i \leq n$

$$\left\{ \begin{array}{l} \text{for } i \text{ even : } s_i = s'_{i/2} \\ \text{for } i = 1 : s_i = x_i \\ \text{else } s_i = s'_{\left(\frac{i-1}{2}\right)} * x_i \end{array} \right\}$$

ניתוח:

$$T(n) = T\left(\frac{n}{2}\right) + O(1) = O(\log n) \text{ - זמן}$$

פעולות -

$$W(n) = W\left(\frac{n}{2}\right) + O(n) = n + \frac{n}{2} + \frac{n}{4} + \dots + 1 \leq 2n \Rightarrow W(n) = O(n)$$

חישוב חיבור 2 מספרים

נתונים 2 מספרים $(a = a_1, \dots, a_n), (b = b_1, \dots, b_n)$ עם n ביטים (n חזקה של 2). נרצה לחשב את החיבור שלהם.

הרעיון יהיה לחשב את הcarry $(c_1, \dots, c_{n+1})$ - הcarry עבור האינדקס i ביעילות בתור אחד נחשב את $m = (a + b) = m_1, \dots, m_{n+1}$.

נחשוב על a_i, b_i כפרמטרים לפונקציה $x_i : \{0, 1\} \rightarrow \{0, 1\}$ שמקבלת $c = carry$ ומחזירה את הcarry של $a_i + b_i + c$.

אם $a_i = b_i = 0$ אז $x_i(c) = 0$ (לא משנה מה השארית מהחיבור הקודם, לא תהיה לי שארית עכשיו).

אם $a_i = b_i = 1$ אז $x_i(c) = 1$ (לא משנה אם היתה שארית - תהיה עכשיו).

אם $a_i \neq b_i$ (אחד מהם 0, השני 1), אזי $x_i(c) = c$.

$$c_1 = 0$$

$$c_2 = x_1(0)$$

$$c_3 = x_2(x_1(0)) \quad \text{אנו מחפשים את}$$

..

$$c_{n+1} = x_n \circ x_{n-1} \circ \dots \circ x_1(0)$$

מכיוון שפעולות ההרכבה (של פונקציות) אסוציאטיביות, אפשר לחשב את $x_1, x_2(x_1), \dots, x_n \circ x_{n-1} \circ \dots \circ x_1$ ע"י prefix-sum ונציב בסוף 0 בכולם. זה ניתן לביצוע ב $O(\log n)$ תורות ו $O(n)$ פעולות.

מהרגע שיש לנו את $c_1, \dots, c_{n+1}$ ניתן לחשב ע"י עוד n פעולות בתור אחד m_i :

$$\forall 1 \leq i \leq n, m_i = (a_1 + b_i + c_i) \bmod 2$$

$$m_{n+1} = c_{n+1}$$

נחשב סיבוכיות: $T(n) = O(\log n)$, ו $W(n) = O(n)$. (תחת ההנחה של n מעבדים).

חישוב מקסימום ב $O(1)$ זמן

נתונים $x_1, \dots, x_n$ מספרים שונים. מחפשים את המקסימלי מביניהם.

אלגוריתם:

נגדיר מטריצה $n \times n$, כך שבמקום ה i, j רשום 1 אם $x_i \geq x_j$, 0 אחרת. אם יש לנו n^2 מעבדים, זה מתבצע ב $O(1)$.

כרגע נבצע bitwise and על השורות. ברגע שנמצע שורה שכולה 1ים (כלומר, bitwise and החזיר 1), אזי האינדקס שלה הוא הנכון.

פסאודו קוד:

נתון מערך A עם $x_1, \dots, x_n$

1. for $1 \leq i, j \leq n$
 if $x_i \geq x_j$
 $B(i, j) = 1$
 else $B(i, j) = 0$
2. for $1 \leq i \leq n$
 do $M(i) = B(i, 1) \wedge \dots \wedge B(i, n)$

בסוף $M(i)$ היחיד שבו יש 1 הוא זה עם האינדקס i של המקסימום.

$$T(n) = O(1) \text{ - זמן}$$

$$W(n) = O(n^2)$$

לא משהו.

אלגוריתם עדיף:

$$T(n) = O(\log \log n) \text{ פעולות, } O(n \log \log n)$$

נניח כי $n = 2^{2^k}$, וכי יש לנו $2n$ מעבדים.

נשווה כל זוג $(x_1, x_2), \dots, (x_{n-1}, x_n)$. קיבלנו $x'_1, \dots, x'_{n/2}$ (פעולות השוואה).

את התוצאה נחלק לרביעיות – יש לנו $\frac{n}{16}$ רביעיות – ניתן לכל אחד 16 מעבדים – אפשר לקבל את המקסימום של כל רביעיה ב $O(1)$. קיבלנו $x_1'', \dots, x_{n/16}''$. נחלק לקבוצות של 16 ונקצה לכל 16, 256 מעבדים. יש לנו $x_1''', \dots, x_{n/256}'''$. וכך הלאה...

כלומר – בתור הראשון משווים ומוצאים מקסימום בכל קבוצה $\{x_1, x_2\}, \{x_3, x_4\}, \dots, \{x_{n-1}, x_n\}$. מקבלים מקסימום $x_1', \dots, x_{n/2}'$. מחלקים לרביעיות, אז יש $\frac{n}{8}$ רביעיות. נקצה לכל רביעיה 16 מעבדים, ואז נחלק לקבוצות של 16, וכן הלאה. נקבל עץ שמספר הקודקודים ברמה k הוא $\frac{n}{2^{2^k}}$, שעומקו $\log \log(n)$. סה"כ $T(n) = O(\log \log n)$ תורות. אבל בכל תור השתמשנו ב $O(n)$ מעבדים, ולכן בסה"כ $T(n) = O(\log \log n)$
 $W(n) = O(n \log \log n)$

22.1.2007

FFT

נתונים $p(x), q(x)$, פולינומים ממעלה $n-1 \geq$. נרצה להכפיל אותם.

$$p(x) = \sum_{i=1}^{n-1} a_i x^i, q(x) = \sum_{i=1}^{n-1} b_i x^i$$
$$p(x)q(x) = \left(\sum_{i=0}^{n-1} a_i x^i \right) \left(\sum_{i=1}^{n-1} b_i x^i \right) = \sum_{i=0}^{2n-2} x^i \sum_{j=0}^i a_j b_{i-j}$$

באופן ישיר, זה יעלה לנו $O(n^2)$.

$$x_1, \dots, x_n \in \mathbb{F}$$

משפט האינטרפולציה:

לכל $y_1, \dots, y_n \in F$ קיים פולינום יחיד ממעלה $n-1 \geq$ כך ש $\forall i, p(x_i) = y_i$.
זה מוביל אותנו לייצוג ע"י ערכים בנקודות, כלומר, במקום להחזיר את $a_0, \dots, a_{n-1}$ מחזיקים את $p(x_1), \dots, p(x_n)$.

נשים לב, שאם יש לנו הצגה בנקודות של p ושל q , אז ההצגה של pq תהיה מכפלת הערכים. כמובן שזה לא מייצג אותו באופן יחיד – ויש צורך ב $2n$ נקודות.

נרצה להכפיל פולינומים בהצגת מקדמים – וכאן יבוא ה FFT

אנו יודעים שכפל בהצגת ערכים עולה $O(n)$, וכפל בהצגת נקודות עולה $O(n^2)$ – ואז אם נדע לעבור ביעילות בין ההצגות, אז הרווחנו.

לכן השאלה – איך עוברים בין הצגות בצורה יעילה – ו FFT עושה את זה ב $(n \log n)$

הנקודות ש FFT מעריך בהן הן שורשי היחידה מסדר שהוא חזקה שלמה של 2. אנחנו צריכים לפחות $2n-1$ נקודות, אז לכן נבחר את החזקה המינימלית של 2 שגדולה מ $2n-1$, כלומר $2^k > 2n-1$. יהיו $\omega_1, \dots, \omega_k$ שורשי היחידה.

נשתמש בהבחנה הבאה:

$$p(x) = \sum_{i=0}^{n-1} a_i x^i = \sum_{i=0}^{n/2-1} a_{2i} x^{2i} + \sum_{i=0}^{n/2-1} a_{2i+1} x^{2i+1} = \sum_{i=0}^{n/2-1} a_{2i} x^{2i} + x \sum_{i=0}^{n/2-1} a_{2i+1} x^{2i}$$

$$\text{נסמן } P_{od} = \sum_{i=0}^{n/2-1} a_{2i+1} x^i, P_{even} = \sum_{i=0}^{n/2-1} a_{2i} x^i \text{ ואז } P(x) = P_{even}(x^2) + x \cdot P_{od}(x^2)$$

שורשי היחידה מסדר n בריבוע הם שני עותקים רצופים של שורשי היחידה מסדר $n/2$ ולכן ניתן

לחשב מערכי P_{even}, P_{od} ב $n/2$ שורשי היחידה מסדר $n/2$ את p בשורשי היחידה מסדר n

$$\text{ב } O(n). \text{ קיבלנו } T(n) = 2 \left(\frac{n}{2} \right) + O(n) \Rightarrow T(n) = O(n \log n)$$

כיוון שני

צריך לדעת גם לעבור מהצגת ערכים בנקודות $1, \omega_k, \dots, \omega_k^{k-1}$ חזרה להצגת מקדמים.

$$\underbrace{\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_k & \dots & \dots & \omega_k^{k-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 1 & \omega_k^{(k-1)} & \omega_k^{2(k-1)} & \dots & \omega_k^{(k-1)(k-1)} \end{pmatrix}}_V \begin{pmatrix} a_0 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} p(1) \\ \vdots \\ p(\omega_k^{k-1}) \end{pmatrix}$$

לבצע את הטרנספורם ההפוך זה כמו להכפיל ב $V^{-1} = kI - V \cdot \overline{V}$ (זה הצמוד המרוכב).

$$V^{-1} = \frac{1}{k} \overline{V}$$

$$V^{-1} = \frac{1}{k} \begin{pmatrix} 1 & \dots & 1 \\ 1 & \overline{\omega_k} & \overline{\omega_k^{k-1}} \\ \vdots & \vdots & \vdots \\ 1 & \overline{\omega_k^{k-1}} & \overline{\omega_k^{(k-1)^2}} \end{pmatrix}$$

ולכן ניתן לחשב את הטרנספורם ההפוך ע"י אותו אלגוריתם כאשר ω_k מוחלף ב $\overline{\omega_k}$ ובסוף מחלקים ב k .

הצפנות:

1. הצפנה פומבית
2. RSA
3. רבין

RSA

המפתח הפומבי (n, e) כאשר e מספר שזר ל $\phi(n)$. מפתח פרטי $d = e^{-1} \pmod{\phi(n)}$. בהנתן $\phi(n)$ קל למצוא את d , אבל למצוא את $\phi(n)$ זה קשה כמו למצוא את p, q .

ההצפנה: בהנתן $1 \leq m < n$ ההצפנה $\varphi(m) = m^e \pmod{n}$

פענוח: $\psi(x) = x^d \pmod{n}$

ראינו בכיתה ש φ, ψ פונקציות הופכיות. אין הוכחות לRSA אבל היא נחשבת בטוחה.

שאלות ממבחן (שנה שעברה)

בנה קוד הפמן:
(אותיות למעלה, תדירות למטה)

$$\begin{pmatrix} a & b & c & d & e \\ 8 & 35 & 12 & 40 & 23 \end{pmatrix}$$

תהי f פונקציית התדירות שמוגדרת. אנו נרצה למזער את $L = \sum_{w \in \Sigma} l(c(w)) \cdot f(w)$

הרעיון הוא לבנות עץ מלמטה מעלה. ראה תרגול לכשנלמד הנושא.

שאלה 1ב', מועד א שנה שעברה

אנו רוצים להגיע מי-ם לת"א, כשבדרך תחנות הדלק הן $d_1, \dots, d_k$. מיכל דלק מספיק לו קילומטרים. נרצה למצוא מסלול שעוצר בכמה שפחות תחנות.

גודל החיפוש מעריכי.

אלגוריתם חמדני לא דווקא עובד. ננסה לבנות אלגוריתם דינמי.

טענה: אם O אופטימלי שהתחנה האחרונה שלו היא $0 \leq j < k$ אז $O \setminus \{j\}$ פתרון אופטימלי לבעיה עבור $d_1, \dots, d_{j-1}$.

הוכחה: נניח בשלילה כי $O \setminus \{j\}$ לא אופטימלי עבור $d_1, \dots, d_{j-1}$, אזי ניקח O' פתרון אופטימלי קצר יותר מ $O \setminus \{j\}$ עבור $d_1, \dots, d_{j-1}$, ואז $O' \cup \{j\}$ פתרון קצר יותר לבעיה המקורית מאשר O , בסתירה לאופטמליות של O .

פסאודו קוד:

$A(i)$ מחזיק את אורך הפתרון האופטימלי לבעיה $\{d_1, \dots, d_i\}$, $B(i)$ מחזיק את התחנה האחרונה במסלול האופטימלי

1. אתחל A, B מערכים באורך $k+2$.

2. $B(1) \leftarrow 0, A(1) \leftarrow 0, A(0) \leftarrow 0, B(0) \leftarrow \text{NULL}$

3. עבור $i = 2, \dots, k+1$

a. $A(i) \leftarrow 1 + \min_{j < i, d_i - d_j \leq n} \{A(j)\}$

b. $B(i) \rightarrow j$ ה j עבורו מתקבל המינימלי.

4. החזר את $(B(n+1), B(B(n+1)), \dots, \text{NULL})$

טענה: $A(i)$ = אורך הפתרון האופטימלי עבור $\{d_1, \dots, d_i\}$

הוכחה: עבור $i = 0, 1$ ברור.

עבור $i \rightarrow i-1$ - נניח פתרון אופטימלי ל $\{d_1, \dots, d_i\}$. נניח התחנה האחרונה שלו היא $i > j$. אז

נקבל כי $1 + A(j) = \text{Induction Hypothesis } OPT(i) = 1 + OPT(j)$ (מהטענה שהוכחנו). ולכן

$A(i) \leq 1 + A(j)$ ולכן $A(i) \leq OPT(i)$. אבל $A(i)$ אורך של פתרון מותר ל $\{d_1, \dots, d_i\}$, ומכאן השוויון.

שאלה 4 ב

נתונה מטריצה של $A(0,1)$ בגודל $n \times n$. תן אלגוריתם יעיל המכריע אם קיימת תמורה $\pi \in S_n$ כך

$$\prod_{i=1}^n A_{i, \pi(i)} > 0$$

זה מופיע בשאלה על רשתות זרימה, לכן ננסה למצוא רשת זרימה.

זה שקול למציאת σ כך ש $A_{i, \sigma(i)} = 1$ לכל i .

נוכיח שבעיה זו שקולה להכרעה האם קיים זיווג מושלם.

נגדיר גרף דו צדדי $V = \left\{ \overbrace{u_1, \dots, u_n}^L, \overbrace{v_1, \dots, v_n}^R \right\}, E = \left\{ (u_i, v_j) \mid \forall A_{ij} = 1 \right\}$

טענה: אם קיימת σ כך ש $A_{i, \sigma(i)} = 1$ לכל i , אז קיים זיווג מושלם בגרף G .

הוכחה: נגדיר $M = \left\{ (u_i, v_{\sigma(i)}) \mid 1 \leq i \leq n \right\}$. ברור ש $M \subseteq E, |M| = n$. צריך להראות שבכל

קודקוד חלה לכל היותר צלע אחת מ M . אז אם $(u_i, v_{\sigma(i)}) \in M$ ו $(u_j, v_{\sigma(j)}) \in M$, אזי אם

$i = j \Rightarrow \sigma(i) = \sigma(j)$, או אם $i \neq j \Rightarrow \sigma(i) \neq \sigma(j)$ כי σ תמורה.

טענה: אם קיים זיווג מושלם ב G אז קיימת σ כך ש $A_{i, \sigma(i)} = 1$ לכל i .

הוכחה: בהינתן M זיווג מושלם נגדיר $\sigma(i) = j$ היחיד כך ש $(u_i, v_j) \in M$. זה מגדיר σ

תמורה כי M זיווג, וכמו כן $A_{i, \sigma(i)} = 1$ לכל i כי $M \subseteq E$ ולכן מכיוון ש $(u_i, v_{\sigma(i)}) \in E$ לכל i

(מבניית σ) אזי $A_{i, \sigma(i)} = 1$ (מבניית G).

הראנו שהבעיה שקולה לבעיית מציאת זיווג מושלם. מאחר ויש לנו אלגוריתם למציאת זיווג מושלם, אזי סיימנו. דיינו.