מבנה נתונים ואלגוריתמים - הרצאה 1

30 באוקטובר 2011

פרטים

הקורס יכיל שני נושאים גדולים - הראשון הוא מבנה נתונים והשני הוא אלגוריתמים.
במבנה נתונים, נלמד מבנה נתונים מופשטים (abstract data structures - ADS).
נלמד על מבני הנתונים:

- תור, מחסנית
- ערימות
- עצים
- עצי חיפוש
- עצים מאוזנים
- טבלאות גיבוב (Hash Tables)

בנושא האלגוריתמים נלמד:

- מיון
- גרפים:
  - מסלולים קצרים ביותר
  - עצים פורשים
  - חלוקה של גרפים
  - Flow - זרימה
- דחיסה
- השוואת מחרוזות
- עצי סיפא
- ישומים של FFT
- תורת אינפורמציה
- תכנון דינמי
- בעיות אופטימיזציה

עלות חישוב

אלגוריתם הוא מעין קופסה שמקבלת קלט, באורך מסוים (n) ומוציא פלט, output, באורך m.
האלגוריתם משתמש בשני אמצעים בסיסיים - CPU וRAM.
אנו שואלים - מה הקשר בין אורך הקלט לבין העלות.
נאמר ש$f\left(n\right)=O\left(g\left(n\right)\right)$ אם קיים $c>0$ ו$N\in\mathbb{N}$ כך שלכל $n>N$ מתקיים $f\left(n\right)\leq c\cdot g\left(n\right)$.
נאמר ש$f\left(n\right)=\Omega\left(g\left(n\right)\right)$ אם קיים $c>0$ ו$N\in\mathbb{N}$ כך שלכל $n>N$ מתקיים $f\left(n\right)\geq c\cdot g\left(n\right)$.

נאמר ש$f\left(n\right)=\Theta\left(g\left(n\right)\right)$ אם $f\left(n\right)=O\left(g\left(n\right)\right)$ וגם $f\left(n\right)=\Omega\left(g\left(n\right)\right)$.
למשל אם ניקח:

$$f\left(n\right)=7n^{3}+1207n-\sin\left(n^{2}\right)$$

ברור שקיים N כך שעבור $n>N$ מתקיים:

$$\begin{aligned} f\left(n\right) & \leq 8n^{3}=O\left(n^{3}\right)\\ f\left(n\right) & \geq n^{3}=\Omega\left(n^{3}\right)\end{aligned}$$

לכן:

$$f\left(n\right)=\Theta\left(n^{3}\right)$$

אם ניקח

$$f\left(n\right)=1.2^{n}+712n^{3}+\log_{10}\left(n\right)^{523}$$

עבור $n>N$ מסוים מתקיים:

$$\begin{aligned} f\left(n\right) & > 1.2^{n}=\Omega\left(1.2^{n}\right)\\ f\left(n\right) & < 2\cdot1.2^{n}=O\left(1.2^{n}\right)\end{aligned}$$

לכן

$$f\left(n\right)=\Theta\left(1.2^{n}\right)$$

ניקח

$$f\left(n\right)=n!+2^{n}+n^{17}$$

אזי עבור $n>N$ מסוים:

$$f\left(n\right)\leq n^{n}=O\left(n^{n}\right)$$

לפי נוסחת סטרלינג:

$$n!\approx\left(\frac{n}{e}\right)^{n}\sqrt{2\pi n}$$

אזי

$$f\left(n\right)\geq\frac{1}{2}\left(\frac{n}{e}\right)^{n}\sqrt{2\pi n}$$

לכן

$$f\left(n\right)=\Omega\left(\left(\frac{n}{e}\right)^{n}\right)$$

אם לדוגמה האלגוריתם לוקח זמן חישוב של

$$f\left(n\right)=c\cdot10^{n}$$

וידוע שעבור $n=1$ הזמן הוא $\frac{1}{100}$ שניה. אז עבור $n=12$ נקבל:

$$c\cdot10^{12}=10^{11}\cdot\left(c\cdot10^{1}\right)$$

וזה יוצא 10^{9} שניות - וזה המון.
לכן אנו מתעניינים בקבוצת כל האלגוריתמים שזמן החישוב שלהם הוא פולינומי. נסמן את הקבוצה הזו באות P.
אם זמן החישוב פולינומי אז אורך הפלט חסום על ידי פולינום. (אם אורך הפלט אקספוננציאלי, אז ברור שגם זמן החישוב אקספוננציאלי).

כעת נסתכל על משתנים בוליאניים (0 או 1), כאשר יש 3 פעולות ביניהם - and $(\wedge)$, or $(\vee)$ ו-not $(\sim)$.
נסתכל על $x_1, ..., x_n$ משתנים בוליאנים. האם קיים צירוף $x_1, .., x_n$ עבורו הביטוי הבא יצא 1:

$$(x_1 \wedge \sim x_2 \wedge x_3) \vee (x_4 ...)$$

יש 2^n אפשרויות לביטוי הזה (2 אפשרויות לכל משתנה). לכן זמן פתרון הבעיה הוא:

$$f(SAT) = O(2^n)$$

אך אם נשתמש בNon-Deterministic Turing Machine, מחשב שמפצל את הבעיה ועוסק בכל בעיה במקביל. עבור כל אפשרות לx_1, הוא עובר על כל אפשרות של x_2 וכך הלאה, ואז זמן החישוב הוא $O(P(n))$ כי לכל x_i זמן החישוב הוא 1 כי הוא עובד על כל האפשרויות שלו במקביל.
לכן נאמר שהבעיה הזו היא Non-Deterministic Polynomial כלומר אם היה לי מכונה לא דטרמיניסטית זמן החישוב יהיה פולינומיאלי. נסמן את קבוצת כל האלגוריתמים שזמן החישוב שלהם הוא פולינומי במכונה לא דטרמיניסטית בNP.
כעת יש שני שלבים לבעיה - מציאת הפתרון ובדיקת הפתרון. אם בדיקת הפתרון היא פולינומיאלית אז מציאת הפתרון היא אלגוריתם NP.
האם $P \neq NP$? האם קיים אלגוריתם שזמן הבדיקה שלו פולינומיאלי אך לא קיים לו פתרון בזמן פולינומיאלי? זו בעיה פתוחה, אין לה פתרון כרגע.

בעיית הסוכן הנוסע

- יש לסוכן n ערים לעבור בהן.
- הוא צריך להתחיל ולסיים באותה העיר.
- בין כל שתי ערים יש מרחק מסוים.
- מותר לעבור בכל עיר פעם אחת בלבד.

מה המרחק הקצר ביותר?
יש $n!$ אפשרויות למסלולים.
נתבונן באלגוריתם הבא:

1. נבחר עיר.
2. פתור את האלגוריתם עבור שאר הערים שעוד לא עברתי בהן.

באלגוריתם זה, אנו מפצלים תחילה לn אפשרויות, ואז כל אפשרות ל$n-1$ אפשרויות וכך הלאה. במכונה לא דטרמיניסטית, נקבל שהאלגוריתם הוא פולינומיאלי - $O(n)$.
זו בעיה שעדיין אין לה פתרון פולינומיאלי רגיל (יש קירובים, אך לא פתרון אופטימלי).
יש עוד 2 שאלות בבעיית הסוכן הנוסע:
שאלה ב' - האם קיים פתרון לסוכן הנוסע שעולה פחות מx (במרחק)?
שאלה ג' - מה העלות במרחק של מסלול מסוים בסוכן הנוסע?
שאלה ג' היא שאלת בדיקה, ושאלה ב' היא שאלת פתרון.

שקילות

אם אני יודע את הפתרון של ב', אני יכול בעזרת שיטת החציה לדעת מה אורך המסלול הקצר ביותר.
כעת נסתכל על האלגוריתם:

1. עבור על כל המרחקים (n^2)
2. הפוך את המרחק ל∞ (מספר גדול).
3. פתור את אורך המסלול הקצר ביותר.
4. אם הפתרון לא השתנה, לא עברתי במסלול הזה. אם הפתרון השתנה, עברתי במסלול הזה.

בעזרת האלגוריתם הזה נמצא את כל המרחקים שעברנו בהם, וכך נמצא את המסלול הקצר ביותר - ונפתור את השאלה הראשונה.
ברור שאם פותרים את א' קל לפתור את ב'. לכן למעשה קיבלנו שא' וב' שקולות.
במילים אחרות - ניתן לעשות רדוקציה פולינומיאלית מא' לב' אם העלות של א' היא פולינומיאלית בהינתן זה שהעלות של ב' פולינומיאלית.

משפט Cook

כל בעיה בNP ניתן לעשות לה רדוקציה בזמן פולינומיאלי לבעיית SAT.

סיכום

למדנו מה זה עלות.
O - חסם עליון.
Ω - חסם תחתון.
Θ - החסם העליון והתחתון שווים.
סדר עלויות:

$$\log < \text{Polynomial} < \text{Exponent} < \text{Factorial}$$

מכונות דטרמיניסטיות ומכונות לא דטרמיניסטיות (מקבילות).
קבוצות אלגוריתמים - P (ניתן לפתור בזמן פולינומיאלי במכונה דטרמיניסטית) וNP (ניתן לפתור בזמן פולינומיאלי במכונה לא דטרמיניסטית).
שקילות בין אלגוריתמים - אם א' שייך לP אז גם ב' שייך לP.
בעיית $P \neq NP$.
משפט $Cook$ - כל בעיה NP שקולה לSAT.
דוגמאות לNP - הסוכן הנוסע וSAT.
הגדרת רדוקציה פולינומיאלית - הפיכת בעיה מסובכת לבעיה יותר פשוטה תוך הישארות באותה קבוצה.
הבדל בין שתי סוגי בעיות - בדיקה ופתרון.