

מבנה נתונים ואלגוריתמים - הרצאה 3

6 בנובמבר 2011

עלות רקורסיה

נניח שיש לנו פונק' רקורסיבית, $REC(x)$. הפונק' מורכבת משני חלקים - גוף הפונק', שנסמן $g(n)$ ואיברי קריאות לרקורסיה $REC(\frac{x}{a})$, אזי מסמנים

$$f(n) = g(n) + bf\left(\frac{n}{a}\right)$$

אך יש דוגמה שונה: פונק' עצרת

$$f(x) = \begin{cases} 1 & x = 1 \\ x \cdot f(x-1) & x > 1 \end{cases}$$

נסמן את העלות של הפונק' ב $k(x)$, אזי:

$$k(x) = 1 + k(x-1)$$

אך זה מקרה יחסית חריג, לרוב נרא פונק' מהצורה הראשונה. למדנו 3 שיטות למצוא את הנוסחה של פונק' כזו - ניחוש והוכחה, הצבה, ומשפט מאסטר.

דוגמה

למשל, מכפלת מטריצות בצורה מסויימת תתן עלות כזו:

$$f(n) = n + 7f\left(\frac{n}{8}\right)$$

תחילה נעשה בהצבה:

$$\begin{aligned} f(n) &= n + 7f\left(\frac{n}{8}\right) \\ &= n + 7\left(\frac{n}{8} + 7f\left(\frac{n}{8^2}\right)\right) \\ &= n\left(1 + \frac{7}{8}\right) + 7^2 f\left(\frac{n}{8^2}\right) \\ &= n\left(1 + \frac{7}{8}\right) + 7^2\left(\frac{n}{8^2} + 7f\left(\frac{n}{8^3}\right)\right) \\ &= n\left(1 + \frac{7}{8} + \frac{7^2}{8^2}\right) + 7^3 f\left(\frac{n}{8^3}\right) \\ &= n\left(1 + \frac{7}{8} + \left(\frac{7}{8}\right)^2 + \left(\frac{7}{8}\right)^{k-1}\right) + 7^k f\left(\frac{n}{8^k}\right) \end{aligned}$$

נעצור כאשר $8^k = n$ כלומר $k = \log_8 n$.

$$\begin{aligned} f(n) &\leq n \frac{(1 - (\frac{7}{8})^\infty)}{1 - \frac{7}{8}} + 7^{\log_8 n} \cdot f(1) \\ &= n \cdot c + n^{\log_8 7} \cdot f(1) = O(n) \end{aligned}$$

כיוון שבפונק' עצמה יש לנו n אז זה גם $\Omega(n)$ ולכן $f(n) = \Theta(n)$. כעת נשתמש במשפט מאסטר:

$$f(n) = g(n) + bf\left(\frac{n}{a}\right)$$

משווים את $g(n)$ ל $n^{\log_a b}$, ויש 3 אפשרויות (ראינו אותן בשיעור הקודם).
 במקרה הזה נקבל ש $f(n) = \Theta(g(n))$.
 כעת בשיטת הניחוש: ננחש שקיים c כך שהחל מ n מסוים מתקיים

$$f(n) \leq cn$$

הוכחה:

ניקח $n = 1$ אז קיים c עבורו $f(1) \leq c$.
 נניח נכונות עבור $k \leq n$. נחשב עבור n :

$$\begin{aligned} f(n) &= n + 7f\left(\frac{n}{8}\right) \\ &\leq n + 7 \cdot c \frac{n}{8} \\ &\leq n \left(1 + \frac{7}{8}c\right) \leq cn \end{aligned}$$

עבור $c > 8$.

ADS - Abstract Data Structures

מחסנית - Stack

עובדת על עקרון LIFO - Last In First Out.
 יש לה מספר פעולות:

- Create - יצירת מחסנית ריקה
- Push - להכניס למחסנית
- Pop - להוציא מהמחסנית
- isEmpty - לבדוק האם המחסנית ריקה
- Empty - לרוקן את המחסנית

למעט Empty, אנו רוצים שהעלות של כל הפעולות תהיה $O(1)$.

ייצוג מחסנית ע"י מערך

- Create:

1. הגדר מערך A בגודל n .

2. הגדרת משתנה $k = 0$ כמונה (כמה מספרים יש לי במחסנית).

- Push(x):

1. $A(k) = x$

2. $k++$

3. if $k \geq n-1$

output(Stack full)

- $x = A.Pop$:

1. אם $k = 0$:

"המחסנית ריקה"

2. אחרת, $x = A(k)$.

• A.IsEmpty :

1. אם $k = 0$ החזר כן.
אחרת לא.

• A.Empty :

1. $k = 0$

ייצוג מחסנית באמצעות רשימה מקושרת

• A.Create :

1. $P = \text{null}$

• A.Push(x) :

1. $P_1 = \text{malloc}(\text{stack})$

2. $P_1 \rightarrow \text{value} = x$

3. $P_1 \rightarrow \text{next} = P$

4. $P = P_1$

• $x = \text{A.Pop}$:

1. אם $P = \text{null}$ החזר "מחסנית ריקה". אחרת:

2. $y = P \rightarrow \text{value}$

3. $P_1 = P$

4. $P = P \rightarrow \text{next}$

5. $\text{free}(P_1)$

• A.IsEmpty :

1. אם $P = \text{null}$ כן, אחרת לא.

• A.Empty :

1. $\text{while } (\sim \text{A.IsEmpty})$

$y = \text{Pop}$

(זה $O(n)$).

תור - Queue

עובד על עקרון FIFO - First In First Out.
הפעולות:

• New - תור חדש

• Enqueue - הכנסת איבר

• Dequeue - הוצאת איבר

• Front - מי בראש התור

• IsEmpty - האם התור ריק

• EmptyQueue - לרוקן את התור

ייצוג תור באמצעות מערך

נשים לב לבעיה מסוימת - אם אני מכניס לפי הסדר, תחילה במקום 0 וכך הלאה, ואז מוציא איבר ומכניס איבר אחר וכו', התור במערך זה ימינה כל הזמן, ובסוף זה יגיע לקצה המערך וייגמר המקום, כשתחילת המערך ריקה. לכן, נכניס איברים למערך בצורה מעגלית, ולכן נצטרך את המשתנים הבאים:

• inp - נקודת הכנסה

• out - נק' הוצאה

• s - גודל התור (כי אחרת לא נוכל לדעת אם התור ריק או מלא)

מימוש הפעולות:

• $Q.New$:

1. הגדר מערך A בגודל N .

2. $inp=out=s=0$

• $Q.IsEmpty$:

1. אם $s = 0$, אחרת לא

• $Q.EmptyQueue$:

1. $inp=out=s=0$

• $Q.Enqueue(x)$:

1. אם $s == N$ החזר "התור מלא". אחרת:

2. $A(inp) = x$

3. $inp++$

4. $s++$

5. אם $inp == N$ או $inp = 0$

• $y = Q.Dequeue$:

1. אם $s == 0$ החזר "התור ריק". אחרת:

2. $y = A(out)$

3. $out++$

4. $s--$

5. אם $out == N$ או $out = 0$

• $y = Q.Front$:

1. אם $s == 0$ החזר "התור ריק". אחרת:

2. $y = A(out)$