מבני נתונים ואלגוריתמים - תרגול 3

13 בנובמבר 2011

תיקון טעות משיעור קודם

המקרה הטוב ביותר של אלגוריתם הפירוק מהשיעור האחרון הוא עבור מספרים מהצורה $n = p^p$ (p ראשוני) ואז מקבלים סיבוכיות $\Theta\left(\frac{\lg n}{\lg\lg n}\right)$ (לא נוכיח).

דוגמאות לאלגוריתמים

כפל פולינומים

קלט:

$$\begin{aligned} f(x) &= a_0 + a_1 x + ... + a_n x^n \\ g(x) &= b_0 + b_1 x + ... + b_n x^n \end{aligned}$$

(f, g ייוצגו כמערכים עם $n+1$ תאים).

פלט:

$$h(x) = f(x) \cdot g(x)$$

אלגוריתם נאיבי:

A, B - מערכי הקלט באורך $n+1$.

C - מערך הפלט באורך $2n+1$.

האלגוריתם:

אלגוריתם 1 אלגוריתם נאיבי לכפל פולינומים

```
for i=0 to 2n:
 C[i]=0
for i=0 to n:
 for j=0 to n:
  C[i+j] += A[i]·B[j]
return C;
```

סיבוכיות הזמן היא $\Theta\left(n^2\right)$.

סיבוכיות הזיכרון היא $\Theta(n)$.

אלגוריתם קאראצובה:

"הפרד ומשול".

f, g פולינומים ממעלה n.

$$\begin{aligned} f(x) &= f_0(x) + x^{\left\lfloor \frac{n}{2} \right\rfloor} f_1(x) \\ g(x) &= g_0(x) + x^{\left\lfloor \frac{n}{2} \right\rfloor} g_1(x) \\ \deg f_1, \deg g_1 &\leq \left\lceil \frac{n}{2} \right\rceil \\ \deg f_0, \deg g_0 &\leq \left\lfloor \frac{n}{2} \right\rfloor \end{aligned}$$

מעכשיו נניח n זוגי.

$$\begin{aligned} f\cdot g &= \left(f_0+x^{\frac{n}{2}}f_1\right)\left(g_0+x^{\frac{n}{2}}g_1\right)\\ &= f_0g_0+\left(f_0g_1+f_1g_0\right)x^{\frac{n}{2}}+f_1g_1x^n \end{aligned}$$

מספיק לחשב את המכפלות f_0g_0, f_0g_1, f_1g_0, f_1g_1 שהן כפל פולינומים ממעלה $\frac{n}{2}$ לכן

$$T\left(n\right)=4T\left(\frac{n}{2}\right)+\Theta\left(n\right)$$

(זה $\Theta\left(n\right)$ כי זה חיבור מערכים).
לפי משפט המאסטר $T\left(n\right)=\Theta\left(n^2\right)$ (כך שלמעשה לא עשינו כלום).
אך מה קאראצובה עשה? קסמים!
נחשב ברקורסיה את

$$f_1g_1,\, f_0g_0,\, h=\left(f_0+f_1\right)\left(g_0+g_1\right)$$

(אלה פולינומים ממעלה $\frac{n}{2}$).
3 המכפלות שאנו צריכים הן

$$f_1g_1,\, f_0g_0,\, h-f_0g_0-f_1g_1$$

האלגוריתם הוא:

אלגוריתם 2 אלגוריתם קאראצובה

אם $n\le 1$ כפול נאיבית.
אם n אי זוגי, נכתוב $n=n+1$ ונרפד את f וg בעוד 0 בסוף.
מחלקים את f ל$f_0,\, f_1$ כמו מקודם.
מחלקים את g ל$g_0,\, g_1$ כמו מקודם.
מחשבים $h_0=f_0+f_1,\, h_1=g_0+g_1$.
מחשבים ברקורסיה את:

$$\begin{aligned} a &= f_0g_0\\ b &= f_1g_1\\ c &= h_0h_1 \end{aligned}$$

מחשבים את $c^{'}=c-a-b$.
לבסוף מחזירים את $a\left(x\right)+x^{\frac{n}{2}}c^{'}\left(x\right)+x^nb\left(x\right)$.

הסיבוכיות:

$$T\left(n\right)=3T\left(\frac{n}{2}\right)+\Theta\left(n\right)$$

לפי משפט המאסטר:

$$T\left(n\right)=\Theta\left(n^{\log_2 3}\right)=\Theta\left(n^{1.58}\right)$$

וזה יותר טוב מ$\Theta\left(n^2\right)$.

מבני נתונים

רשימה מקושרת

מורכבת מתאים וכל תא מחזיק איבר ומצביע לאיבר הבא.
שומרים מצביע לתחילת הרשימה.

הבדלים ממערך

- אם יש לי איבר אפשר להוסיף איבר אחריו ב$O\left(1\right)$ או להסיר אותו ב$O\left(1\right)$.
- אם יש מצביע לאיבר ברשימה אפשר "לנוע קדימה" ב$O\left(1\right)$.

הערות

- לפעמים שומרים גם מצביע לאיבר האחרון ברשימה כי אז אפשר לשרשר שתי רשימות.

וריאציה נפוצה

רשימה דו מקושרת - שומרים בכל איבר מצביע לאיבר הבא ולאיבר הקודם.

מחסנית

- Top - מחזיר את האיבר בראש המחסנית ($O(1)$)
- Pop - מוציאים את האיבר הראשון במחסנית ומחזירים אותו.
- $\text{Push}(x)$ - דוחפים את x לראש המחסנית.
- IsEmpty - האם המחסנית ריקה.

תור

אותן הפעולות כמו מחסנית עם ההבדל שpush מכניסה איבר לסוף התור.

דרך קלה לזכור

תור - FIFO : First In First Out.
מחסנית - LIFO : Last In First Out.

תרגיל 1

יש שתי מחסניות. אחת מלאה ואחת ריקה.
כתבו אלגוריתם המעביר את תוכל המחסנית הראשונה למחסנית השניה ומרוקן את הראשונה.

פתרון

קלט: S_1 מלאה, S_2 ריקה.

אלגוריתם 3 פתרון תרגיל 1

ניצור מחסנית ריקה T.

```
while S_1.notempty():
  T.Push(S_1.Pop())
while T.notempty():
  S_2.Push(T.Pop())
```

זמן ריצה: $\Theta(n)$ כאשר n הוא מס' האיברים בS_1.

תרגיל 2

יש n אנשים ששמותיהם $1, ..., n$ שעומדים במעגל. מתחילים מאיש מס' 1.
מדלגים k אנשים ומוציאים את האיש ה-k-י מהמעגל.
חוזרים על התהליך (כשמתחילים היכן שעצרנו) עד אשר אין אנשים במעגל.
כתבו אלגוריתם המדפיס את האנשים המוצאים לפי הסדר של ההוצאה ("תמורת יוספוס").

פתרון

נשתמש בתור.

אלגוריתם 4 פתרון תרגיל 2

```
Q = empty queue
for i=1 to n:
 Q.push(i)
while Q.not_empty():
 for j=1 to k-1:
  Q.Push(Q.Pop())
 print(Q.Pop())
```

סיבוכיות זמן: $\Theta(nk)$.

ערימה (תור עדיפויות)

- push - $O(\lg n)$
- pop - $O(\lg n)$
- top - $O(1)$

תרגיל 3

נתונות k רשימות ממוינות באורך n. מזגו את הרשימות לרשימה ממוינת אחת ב-$O(n\lg k)$.

פתרון

רוצים בכל צעד למצוא את האיבר הקטן ביותר מבין כל ההתחלות ולהכניס אותו למערך הפלט.
נחזיק k מצביעים לכל רשימה - $i_1, ..., i_k$.
כל פעם נמצא את r כך ש

$$L_r[i_r] = \min_{s\in\{1,...,k\}}\{L_s[i_s]\}$$

כאשר $L_1, ..., L_k$ הרשימות, ונוסיף את $L_r[i_r]$ לרשימה הממוזגת ונקדם את i_r ב-1.
בבית - ראו שאם מוצאים את i_r ע"י מעבר על כל $i_1, ..., i_k$ אז הסיבוכיות היא $O(nk^2)$.
נמצא את המינימום בעזרת ערימה:

אלגוריתם 5 פתרון תרגיל 3

```
H = empty heap of pairs (מספר והאינדקס של הרשימה ממנה המספר הגיע)
i_1, ..., i_k = array of integers
for i=1 to k:
  i_j=0
for j=1 to k:
  H.push(L_j[i_j], j)
for x=0 to nk-k-1:
  (a, j)=H.pop()
  output[x]=a;
  i_j++
  H.push(L_j[i_j], j)
x=nk-k
while H.not_empty():
  output[x]=H.pop()
  x++;
```

סיבוכיות:
הלולאה הראשית היא $(nk-k)\Theta(\lg k) = \Theta(nk\lg k)$.