Data Structures ~ Homework 1 ~ *2025B*

Shahar Perets

April 21, 2025

הערות לבודק:

תרגיל בית זה לא נעשה על גבי ה־PDF שניתן בקובץ התרגיל בהתאם לאישור מהפורום במודל.
כל שאלה מופיעה בעמוד נפרד.

פרטים אישיים

שם: שחר פרץ
תעודת זהות: 334558962

המשך בעמוד הבא

. . . . (1) . . . .

בשאלה זו נבנה פעולה `Shift(L, i)` בתנאים שונים.

(א) באמצעות ה־ADT של List:

input : List L, Integer i
output: nothing (in-place)

```
P = List()
n = Length(L)
for j = 0 to n do
    Insert(P, Length(P − 1), Retrieve(L, (i + j) mod n))
end
for j = 0 to n do
    Insert(L, j, Retrieve(P, j)) // replace L[j] with P[j]
    Delete(L, j + 1)
end
```

לא ניתן לנתח את סיבוכיות זמן הריצה שכן לא ידועה סיבוכיות זמן הריצה של פעולות תלויות ADT בהן האלגו' משתמש, כמו `Retrieve`.

(ב) במערך מעגלי L:

```
j ← 0
n ← L.length
M ← L.maxlength
start ← L.start
do
    val ← L.array[(start + n − 1 − j) mod M]
    L.array[(start − 1 − j) mod n]
    j ← (j + 1) mod M
while j ≠ i mod M
L.start ← start − i
```

הפעולות המבועות (לבדן) הן כתיבה ל־array ולקיחת מידע ממנו יארכו $O(1)$, בעוד הלולאה תיגמר רק כאשר נעביר i מספרים מסוף ה־arrya לתחילתו (המונה j גדל ב־1 בכל ריצה), עד לכדי מודולו בגודל של המערך הפנימי M. סה"כ $\Theta(i \bmod M) = O(i)$.

(ג) ברשימה מקושרת חד־כיוונית:

input : Linked List L, Integer i
output: nothing (in-place)

```
n ← Length(L)
cutNode ← Retrieve(L, n − i)
if cutNode.next ≠ null then
    L.last ← cutNode
    cutNode.next ← L.first
    L.first ← cutNode.next
end
```

אין לולאות וכל הפעולות פרט ל־Retrieve אורכות $O(1)$, בעוד Retrieve אורך $O(n-i)$, וסה"כ $\Theta(N-i+k)$ כאשר k קבוע כלומר $\Theta(n-i)$ או $O(n)$.

המשך בעמוד הבא

א. נתאר מימוש ל"מחסנית המינימום" כאשר סיבוכיות הזמן הנדרשת לכל פעולה היא $O(1)$.

לשם בניית מבנה נתונים כזה, ניעזר בשתי רשימות מקושרות דו־כיווניות. לאחת נקרא Content ולשנייה Minimums. הסרת איבר (ב־ADT קרוי RemoveLast) תתבצע באמצעות הסרת ה־node/איבר האחרון בשתי הרשימות (הסרת איבר מרשימה מקושרת אורך $O(1)$, ופעמיים $2O(1) = O(2) = O(1)$). איתחול הרשימה (Init) ידרוש אתחול את שתי הרשימות מקושרות, שזוהי גם פעולה שאורכת $O(1)$ פעמיים. לקיחת המינימום (Min) תתבצע ע"י פעולת RetrieveLast של הרשימה המקושרת, כלומר באמצעות גישה לאיבר האחרון בה. זוהי גם פעולה האורכת $O(1)$. הוספת איבר תתבצע בצורה הבאה:

input : A compareable number val
output: nothing (in-place)

```
min ← Retrieve-Last(Minimums).item
if min < val then
 | Insert-Last(Minimums, val)
else
 | Insert-Last(Minimums, min)
end
Insert-Last(Content, val)
```

משום שפעולות Insert-Last ו־ Retrieve-Last ברשימה מקושרת דו־כיוונית אורכות $O(1)$, ובגלל שכן גם בעבור השוואות בין מספרים גם אורכות $O(1)$, סה"כ כל הפעולות יקחו $O(1)$ גם כאן.

ב. נרצה להוסיף פעולה Add'(d) המוסיפה קבוע d לכל המספרים הנוכחיים. נסמן את הפעולות במבנה החדש שביקשנו לבנות ב־' בסוף שם הפעולה. בעת אתחול המבנה, נאתחל משתנה בשם Addition שבתאחול יהיה שווה ל־0. נסמן ב־: הגדרת פונקציה שלא מחזירה כלום, וב־= פונקציית למבדא קצרה. כך, נגדיר מחדש את ה־ADT באופן הבא: (פעולות שלא צוינו יותרו ללא שינוי)

```
Add(d): Addition ← Addition + d
Min'(L)=Min(L) + Addition
Insert'(L, n)=Insert(O, n − Addition)
Retreive'(L, i)=Retrieve(L, i) + Addition
```

כל השינויים הם ברמת פעולות אריתמטיות בלבד שאורכות $O(1)$, ועל כן לא יגדילו את הסיבוכיות האסימפטוטית.

ג. נוסיף את הפעולה DeleteMin שתמחק את המספר הקטן ביותר מהמבנה.

```
currentMin ← Retrieve-Last(Minimums)
currentVal ← Retrieve-Last(Content)
while currentVal.item ≠ currentMin.item do
 | currentVal ← currentVal.prev
 | currentMin ← currentMin.prev
end
currentVal.prev.next ← currentVal.next      // change the .next pointer of the previous node
currentMin.prev.next ← currentMin.next      // same here
```

נטען שהסיבוכיות היא $O(t)$ כאשר t מספר האיברים שהוכנסו אחרי המינימום. הפעולה הכבדה היחידה, הלולאה, מתחילה מסוף מבנה הנתונים ובפנים חוזרת איבר אחורה, ומסתיימת כאשר האיבר האחרון ברשימת המינימומים הוא האיבר ברשימה הכוללת את איברי המבנה עצמו. זה יקרה כאשר המינימום האחרון (והיחיד, משונות איברי הרשימה). נוכיח את הטענה: נסמן ב־m את המיקום האחרון, נפצל למקרים, אם אנחנו ב־$i > m$, אז מיחידותו אנחנו באיבר אחר ובפרט גדול ממנו (כי אינו המינימום האחרון בעצמו) ולכן Content[i] לא הנוכחי Minimums[i], אם $i = m$ אז המינימום האחרון הוא Content[i] = Minimums[i], אחרת $i > m$ אך לא נגיע לנקודה זו באלגוריתם כי הלולאה הייתה נעצרת במקרה הקודם יותר בו $i = m$. סה"כ הלולאה נעצרת כאשר היא מגיעה למינימום האחרון שהוכנס, ולוקח זמן ליניארי להגיע לשם (בכל איטרציה חוזרת אחד אחורה) ולכן $O(t)$ סיבוכיות (כל שאר הפעולות מתבצעות ב־$O(1)$).

המשך בעמוד הבא

(3)

א. נראה כי בעבור הפונקציות הבאות, $\boldsymbol{f(n) = o(g(n))}$

$$f(n) = n^{15}\log^{12} n,\ g(n) = \frac{n^{17}}{\log^{12} n}$$

$$\limsup_{n\to\infty}\frac{f(n)}{g(n)} = \limsup_{n\to\infty}\frac{n^{15}\log^{12} n}{\frac{n^{17}}{\log^{12} n}} = \limsup_{n\to\infty}\frac{n^{15}\log^{24} n}{n^{17}} = \limsup_{n\to\infty}\frac{\log^{24} n}{n^2} \overset{!}{=} 0$$

נבחין כי הטענה שנדרשה מאיתנו בסוף נגררת מכך ש־$\log^a n = o(n^2)$, שנובע מהירכיית חסמים.

ב.

$$f(n) = \sum_{i=1}^{n}\log i^3,\ g(n) = \sum_{i=1}^{n}\log i^2$$

נראה $\boldsymbol{f(n) = \Theta(g(n))}$:

$$f(n) = \sum_{i=1}^{n}\log(i^3) = 3\sum_{i=1}^{n}\log i = \Theta\left(\sum_{i=1}^{n}\log i\right) = \Theta\left(2\sum_{i=1}^{n}\log i\right) = \Theta\left(\sum_{i=1}^{\log n}\log(i^2)\right) = \Theta(g(n)) \quad \top$$

ג. נוכיח כי $\boldsymbol{f(n) = \omega(g(n))}$ באמצעות כך שנראה ש־$g(n) = o(f(n))$.

$$f(n) = (\log n)^n,\ g(n) = (\sqrt{n})^{\log n}$$

$$\limsup_{n\to\infty}\frac{g(n)}{f(n)} = \limsup_{n\to\infty}\frac{(\sqrt{n})^{\log n}}{(\log n)^n} = \limsup_{n\to\infty}\left(\frac{(\sqrt{n})^{\log(\sqrt[n]{n})}}{\log n}\right)^n \overset{(1)}{=} \limsup_{n\to\infty}\left(\frac{(\sqrt{n})^0}{\log n}\right)^n = \left(\limsup_{n\to\infty}\frac{1}{\log n}\right)^n = 0^n = 0$$

השוויון המוסמן ב־(1) נכון כי $\lim_{x\to\infty}\sqrt[x]{x} = 1$, וכן $\log 1 = 0$.

ד.

$$f(n) = n^n,\ g(n) = n!$$

נוכיח באמצעות גבולות ש־$\boldsymbol{f(n) = \omega(g(n))}$

$$\liminf_{n\to\infty}\frac{f(n)}{g(n)} = \liminf_{n\to\infty}\frac{n^n}{n!} = \liminf_{n\to\infty}\frac{\prod_{i=1}^{n} n}{\prod_{i=1}^{n} i} = \liminf_{n\to\infty}\prod_{i=1}^{n}\frac{n}{i} = \liminf_{n\to\infty}\frac{n}{1}\cdot\prod_{i=2}^{n}\frac{n}{i} = \infty\cdot\left(\liminf_{n\to\infty}\underbrace{\prod_{i=2}^{n}\frac{n}{i}}_{\geq 1}\right) = \infty$$

(הערה: החלק המסומן כ־≥ 1 גדול ממש מאחד כי הוא מכפלה של מספרים גדולים מאחד, שכן $\frac{n}{i} \geq 1$ לכל $n \leq i$ ובפרט $i \in [n]\setminus\{2\}$).

ה.

$$f(n) = 1.6^{\log\log\log n},\ g(n) = \log\log n$$

נראה ש־$\boldsymbol{f(n) = \omega(g(n))}$. נעשה זאת באמצעות תחשיב גבולות:

$$\lim_{x\to\infty}\frac{f(n)}{g(n)} = \lim_{x\to\infty}\frac{1.6^{\log\log\log n}}{\log\log n} = \lim_{x\to\infty}\left(\frac{1.6}{(\log\log n)^{\frac{1}{\log\log\log n}}}\right)^{\log\log\log n} = \lim_{x\to\infty}\left(\frac{1.6}{(\log\log n)^0}\right)^{\log\log\log n} = \lim_{x\to\infty}\left(\frac{1.6}{1}\right)^{\log\log\log n} = 1.6^{\infty} = \infty$$

כדרוש.

המשך בעמוד הבא

יהיו $f,g\colon \mathbb{N}^{\mathbb{N}}$. נוכיח/נפריך את הטענות הבאות.

א. **נפריך** את הטענה $f=o(g) \implies f\neq\Omega(g)$. נתבונן בשתי הפונקציות הבאות:

$$D_1=\begin{cases}0 & n\in\mathbb{N}_{\text{even}}\\ 1 & n\in\mathbb{N}_{\text{odd}}\end{cases},\ D_2=\begin{cases}0 & n\in\mathbb{N}_{\text{odd}}\\ 1 & n\in\mathbb{N}_{\text{even}}\end{cases}$$

ידוע $\limsup_{n\to\infty}\frac{D_1}{D_2}<\infty$ אמ"מ $D_1=\Omega(D_2)$. לכן $D_1\neq\Omega(D_2)$ אמ"מ השוויון של הגבול לא מתקיים. מכיוון שהגבול לא מוגדר הוא בפרט אינו קטן מאינסוף ולכן $D_1\neq\Omega(D_2)$. אך גם הגבול $L:=\limsup_{n\to\infty}\frac{D_2}{D_1}$ אינו מוגדר, ובגלל ש־$D_1=o(D_2)\iff$ $L=0$ אז $D_1\neq o(D_2)$ כי L אינו מוגדר ובפרט לא שווה ל־0. סה"כ מצאנו שתי פונקציות כך ש־$D_1\neq\Omega(D_2)\wedge D_1=o(D_2)$ וזו דוגמה נגדית לטענה.

ב. **נוכיח.** נניח שגבול במנה $\frac{f}{g}$ קיים, ונראה שאם $f\neq\Omega(g)$ אז $f=o(g)$.

הוכחה. מההנחה $f\neq\Omega(g)$ ידוע ש־$\neg\liminf_{n\to\infty}\frac{f(x)}{g(x)}>0$. משום ששתי הפונקציות חיוביות אז $\liminf_{n\to\infty}\frac{f(x)}{g(x)}\geq 0$, וסה"כ $\liminf_{n\to\infty}\frac{f(x)}{g(x)}=0$ ובפרט $\lim_{x\to\infty}\frac{f(x)}{g(x)}=0$. באופן שקול $f=o(g)$ כדרוש. ■

ג. **נפריך** כי $\forall f,g\in\mathbb{N}^{\mathbb{N}}\colon f=O(g)\vee g=O(f)$.

הפרכה. נראה דוגמה נגדית. נבחר $f:=D_1,g:=D_2$ מסעיף א'. נניח בשלילה את הטענה. נפריד למקרים:

- אם $f=O(g)$, אז קיימים $0<c\in\mathbb{R}$,$n_0\in\mathbb{N}$ כך ש־$\forall n\geq n_0\colon f(n)\leq cg(n)$. נבחין כי או n_0 או n_0+1 אי־זוגיים, ונסמן ב־n_0' את אי־הזוגי מביניהם. נבחין כי $c=1\cdot c=cg(n_0')\leq f(n_0')=0$ וסה"כ $0=c>0$ ובפרט $0\neq 0$ וזו סתירה.
- אם $g=O(f)$ אז קיימים $0<c\in\mathbb{R}$,$n_0\in\mathbb{N}$ כך ש־$\forall n\geq n_0\colon g(n)\leq cf(n)$. גם במקרה הזה או n_0 או n_0+1 זוגיים ואת הזוגי מביניהם נסמן n_0'. אז $0<c=cf(n_0')\leq g(n_0')=0$ וכאן $0\neq 0$ וסתירה גם כאן.

סה"כ סתירה בכל המקרים. ■

ד. **נפריך.** נתבונן בטענה אם f,g מונוטוניות עולות אז $f=O(g)\vee g=O(f)$.

הפרכה. נבנה שתי פונקציות הסותרות את הטענה.

$$f(n)=\begin{cases}n^{n-1} & n\in\mathbb{N}_{\text{even}}\\ n^{n} & n\in\mathbb{N}_{\text{odd}}\end{cases},\ g(n)=\begin{cases}n^{n} & n\in\mathbb{N}_{\text{even}}\\ n^{n-1} & n\in\mathbb{N}_{\text{odd}}\end{cases}$$

- אם $f(n)=O(g(n))$ אז קיים c כך ש־$f(n)\leq cg(n)$ לכל $n\geq n_0$. בפרט בעבור איזשהו $n\geq n_0+1$ אי־זוגי, יתקיים $n^n\leq cn^{n-1}$. נחלק ונקבל $n\leq c$. נסמן ב־c' את $c+1$ אם $c+1$ אי־זוגי אחרת $c':=c+2$, בפרט $n\leq c\leq c'$ ולכן $f(c')\leq cg(c')$ כלומר $c'^{c'}\leq cc'^{c'-1}$, גם כאן נחלק ונקבל $c'\leq c$, אך $c+1\leq c'\leq c$ כלומר $1\leq 0$ וסתירה.
- אם $g(n)=O(f(n))$ אז קיים c כך ש־$g(n)\leq cf(n)$ לכל $n\geq n_0$. בפרט בעבור איזשהו $n\geq n_0+1$ זוגי, יתקיים $n^n\leq cn^{n-1}$. נחלק ונקבל $n\leq c$. נסמן ב־c' את $c+1$ אם $c+1$ זוגי אחרת $c':=c+2$, בפרט $n\leq c\leq c'$ ולכן $g(c')\leq cf(c')$ כלומר $c'^{c'}\leq cc'^{c'-1}$, גם כאן נחלק ונקבל $c'\leq c$, אך $c+1\leq c'\leq c$ כלומר $1\leq 0$ וסתירה.

עתה נראה שהפונקציות מונוטוניות עולות:

$$\begin{cases}\text{if } n\in\mathbb{N}_{\text{even}}: & f(n)=n^{n-1}\leq n^{n+1}\leq(n+1)^{(n+1)}=f(n+1) \quad g(n)=n^n\leq(n+1)^n=g(n+1)\\ \\ \text{if } n\in\mathbb{N}_{\text{odd}}: & f(n)=n^n\leq(n+1)^n=f(n+1) \quad g(n)=n^{n-1}\leq n^{n+1}\leq(n+1)^{n+1}=g(n+1)\end{cases}$$

סה"כ בכל המקרים אכן $\forall n\in\mathbb{N}\colon f(n)\leq f(n+1)\wedge g(n)\leq g(n+1)$, כלומר f,g מונוטוניות עולות כדרוש. זו אכן סתירה למשפט, וסיימנו. ■

המשך בעמוד הבא

(5)

5.1 סעיף א'

נוכיח באינדוקציה ש־$T(n) = T(\lfloor \alpha n \rfloor) + T(\lfloor (1-\alpha)n \rfloor) + 1 = O(n)$ עבור $\alpha < 1$. נניח $T(c) = 1$ עבור ערכי c קטנים.

הוכחה. בסיס. נקבל $1 = T(n) = T(\lfloor \alpha n \rfloor) + T(\lfloor 1-\alpha \rfloor n) + 1 = 0 + 0 + 1 = cn$ עבור $n = 1$.

צעד. נניח באינדוקציה מלאה את הטענה בעבור $k < n$, ונוכיח בעבור n. מה.א.:

$$\begin{aligned} T(n) &= T(\lfloor \alpha n \rfloor) + T(\lfloor 1-\alpha \rfloor n) + 1 \\ &= c\lfloor \alpha n \rfloor + c\lfloor (1-\alpha)n \rfloor + 1 \\ &\le c(\alpha n + n - \alpha n) + 1 \quad \text{ה.א.} \\ &\le cn + 1 \le cn \end{aligned}$$

נבחר $c = 1$ ו־$n_0 = 2$ ואכן כל השוויונות לעיל יתקיים כך שקיבלנו $\forall n \ge n_0 \colon T(n) \le cn$ כדרוש.

הערה: השתמשנו בנתון $\alpha < 1$ כאשר הפעלנו את הנחת האינדוקציה, עבורה צריך להתקיים $\alpha n \le n \wedge (1-\alpha)n \le n$. ■

5.2 סעיף ב'

א. $T(n) = 8T\left(\frac{n}{2}\right) + n^3 + n^2 \cdot \log^3 n$. נוכיח $T(n) = \Theta\left(n^3 \log n\right)$ באינדוקציה מלאה על n. ה.א. $c^- n^3 \log n \le T(n)$ לכל $n \ge n_0^-$ כלשהו. נבחר $c^- = 1$. צעד:

$$c^- n^3 \log n \overset{(1)}{\le} c^- n^3(\log n - \underbrace{\log 2}_{=1}) + n^3 \le \underbrace{8c^- \left(\frac{n}{2}\right)^3}_{c^- n^3} \cdot \log\left(\frac{n}{2}\right) + n^3 + n^2 \log^3 n \le \underbrace{8T\left(\frac{n}{2}\right) + n^3 + n^2 \log^3 n}_{T(n)}$$

הערה: (1) יתקיים לכל $c^- \ge 1$, ובפרט $c^- = 1$.

קיבלנו $c^- n^3 \log n \le T(n)$ כלומר $T(n) = \Omega\left(n^3 \log n\right)$.

נוכיח את הכיוון השני. ה.א. $T(n) \le c^+ n^3 \log n$ לכל $n \ge n_0^+$. צעד:

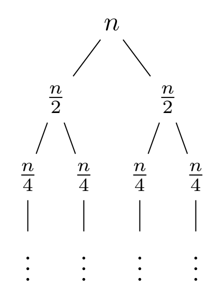
$$T(n) = 8T\left(\frac{n}{2}\right) + n^3 + n^2 + \log^3 n \overset{(1)}{\le} 8c^+ \left(\left(\frac{n}{2}\right)^3 \log\left(\frac{n}{2}\right)\right) + 2n^3 \overset{(2)}{\le} c^+ n^3 \log^3 n - c^+ n^3 + 2n^3 \overset{(3)}{\le} c^+ n^3 \log^3 n$$

(3) מתקיים לכל $c^+ \ge 2$, ונבחר $c^+ = 2$. מהררכיית חסמים, (2) ו־(1) מתקיים החל מ־n_1, n_2 כלשהו, ונבחר להוכיח את הטענה בעבור $n_0^+ = \max\{n_1, n_2\}$.

קיבלנו $T(n) = O(n^3 \log n)$ ומהאינדוקציה הקודמת סה"כ $\boldsymbol{T(n) = \Theta\left(n^3 \log n\right)}$.

לא ניתן לספק בסיסים לאינדוקציות כי לא ניתן בסיס לנוסחת הנסיגה. עם זאת, הבסיס יכול להתחיל מכל נקודה n_0, אז אין זה משנה כל עוד קיים בסיס קבוע.

ב. $T(n) = 2T\left(\frac{n}{2}\right) + \sqrt{n}$. נפעל בשיטת העץ:



בשכבה ה־i בעץ תהיה סיבוכיות $\sqrt{\frac{n}{2^i}} \cdot 2^i$, ויהיו $\log n$ שכבות (מתי שהביטוי $2^i < 1$ מקרה בסיס בו סיבוכיות $T(<1) = 1$), ונקבל:

$$\sum_{i=0}^{\log n - 1} 2^i \cdot \sqrt{\frac{n}{2^i}} = \sqrt{n} \sum_{i=0}^{\log n - 1} \sqrt{\frac{4^i}{2^i}} = \sqrt{n} \sum_{i=0}^{\log n - 1} \sqrt{2}^i = \sqrt{n}(\sqrt{2})^{\log n} = \sqrt{n}\sqrt{2^{\log n}} = (\sqrt{n})^2 = \boldsymbol{\Theta(n)}$$

ג. נתבונן בנוסחה $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log n}$. העץ זהה לעץ של הסעיף הקודם, וגם כאן בכל שכבה יהיו 2^i צמתים, אך הפעם בכל צומת

סיבוכיות של $\frac{\frac{n}{2^i}}{\log\left(\frac{n}{2^i}\right)}$. סה"כ נסכום את השכבות:

$$T(n)=\sum_{i=0}^{\log n-1}2^i\frac{\frac{n}{2^i}}{\log\left(\frac{n}{2^i}\right)}=n\sum_{i=0}^{\log n-1}\frac{1}{\log n-i}\overset{(1)}{=}n\sum_{i=1}^{\log n}\frac{1}{i}=n\log(\log(n-1))=\boldsymbol{\Theta(n\log\log n)}$$

השוויון ה־(1) נכון מהפיכת סדר הסכימה.

המשך בעמוד הבא

. . . . (6) . . . .

נפתור את נוסחאות הנסיגה הבאות באמצעות נוסחאת המאסטר:

1. נתבונן בנוסחא $T(n) = 3T\left(\frac{n}{3}\right) + n\log^3 n$. הקבועים הם $a = b = 3$, ונבחין ש־$f(n) := n\log^3 n = \omega(n^{\log_3 3}) = \omega(n)$, ולכן אין זה בהכרח אחד משני המקרים הראשונים של מספר המאסר. אך לכל $\varepsilon > 0$ כבר מתקיים $f(n) = o(n^{1+\epsilon})$, ולכן אין זה גם המקרה השלישי של המשפט. סה"כ לא **ניתן להראות חסם** לנוסחאת נסיגה זו באמצעות משפט המאסטר בגלל שאין הפונקציה נופלת לאף אחד משלושת המקרים של המשפט.
2. נתבונן בנוסחאת הנסיגה $T(n) = T\left(\frac{4n}{5}\right) + 8$. כאן הקבועים $a = 1, b = \frac{5}{4}, f(n) = 8$. מתקיים $f(n) = 8 = \Theta(n^0) = \Theta(n^{\log_{5/4} 1})$. זהו המקרה השני של משפט המאסטר ובו $T(n) = \Theta(n^0 \cdot \log n) = \boldsymbol{\Theta(\log n)}$.
3. נתבונן בנוסחאת הנסיגה $T(n) = 4T\left(\frac{n}{2}\right) + 2n$. כאן הקבועים $b = 2, a = 4, f(n) = 2n$ ומתקיים $2n = \Theta(n) = O(n^{1.5}) = O\left(n^{\log_2 4-\varepsilon}\right)$ בעבור $\varepsilon = 0.5 > 0$. זהו המקרה הראשון של משפט המאסטר ובו $\boldsymbol{T(n) = \Theta(n^2)}$.
4. נתבונן בפונקציה $T(n) = 3T\left(\frac{n}{4}\right) + n\log^3 n$. מתקיים $n\log^3 n = \Omega\left(n^{\log_4 3+\varepsilon} = \Omega\left(n^{0.8}\right)\right)$ עבור $\varepsilon = 0.8 - \log_4 3 \approx 0.0075 > 0$. זהו המקרה השלישי של משפט המאסטר. למקרה זה יש עוד תנאי, והוא להראות קיום $c < 1, n_0 \geq 0$ כך שלכל $n \geq n_0$ מתקיים $3\frac{n}{4}\log^3\left(\frac{n}{4}\right) \leq c \cdot n\log^3 n$. מחוקי לוגוריתמים:

$$\frac{3}{4}n\log^3\left(\frac{n}{4}\right) = \frac{3}{4}n\left((\log n - \log 4)^3\right) \leq \frac{3}{4}n\left(\log^3 n\right)$$

וסה"כ קיבלנו פסוק אמת בעבור $n - 3$. לכן נוכל להשתמש במקרה השלישי של המשפט ולקבל $\boldsymbol{T(n) = \Theta\left(n\log^3 n\right)}$.

. . . .

שחר פרץ, 2025

קומפל ב־LaTeX ונוצר באמצעות תוכנה חופשית בלבד