Υπογραμμικοί Αλγόριθμοι
Μάθημα 3 - 15/10/2019

Επιμέλεια σημειώσεων: Κωνσταντίνος Βροχίδης

1 Πρόβλημα Διακριτών Στοιχείων (Δ.Σ.)

Στο προηγούμενο μάθημα είχαμε αποδείξει ότι οποιοσδήποτε *ντετερμινιστικός* αλγόριθμος (είτε ακριβής είτε προσεγγιστικός) λύνει το πρόβλημα Δ.Σ. χρησιμοποιεί χώρο $\Omega(n)$.
Θα δείξουμε ότι το ίδιο ισχύει και για τους πιθανοτικούς αλγορίθμους, αποδεικνύοντας το ακόλουθο.

Θεώρημα 1. *Δεν υπάρχει πιθανοτικός ακριβής αλγόριθμος για το πρόβλημα διακριτών στοιχείων με χώρο $O(n)$.*

Για την απόδειξη θα βασιστούμε στη δομή της *πολυπλοκότητας επικοινωνίας*.

Πολυπλοκότητα Επικοινωνίας

Ας φανταστούμε το σενάριο στο οποίο υπάρχει μια συνάρτηση f και δύο άτομα, η Αλίκη και ο Μπάμπης. Η Αλίκη έχει ένα κομμάτι της εισόδου και ένα άλλο έχει ο Μπάμπης. Στόχος του είναι να υπολογίσουν *μαζί* την f στην είσοδο. Το ερώτημα που τίθεται είναι: Πόσα δυφία πρέπει να ανταλλάξουν για να υπολογίσουν την f;

Παράδειγμα 1.1.

(i) Η Αλίκη θέλει να στείλει έναν αριθμό στο $\{1, \ldots, N\}$. Δηλαδή, η Αλίκη και ο Μπάμπης θέλουν από κοινούν να υπολογίσουν την συνάρτηση $x \mapsto x$ με την Αλίκη να έχει όλη την είσοδο και τον Μπάμπη να μην έχει τίποτα. Προφανώς η Αλίκη πρέπει να στείλει όλη την είσοδό της στον Μπάμπη και ο Μπάμπης δεν χρειάζεται τίποτα στην Αλίκη. Κατά συνέπεια αρκούν $\log n$ δυφία.

(ii) Αν $x = (x_1, \ldots, x_N)$ και $f(x) = \left(\sum_{i=1}^{N} x_i\right) \mod p$ με την Αλίκη να έχει $x_S = (x_i)_{i \in S}$ για $S \subseteq \{1, \ldots, n\}$ και τον Μπάμπη να έχει $x_{\bar{S}}$ όπου $\bar{S} = \{1, \ldots, N\} \setminus S$, τότε αρκεί ο καθένας να στείλει $\log p$ δυφία στον άλλο.

Επομένως για να δείξουμε ότι δεν υφίσταται αλγόριθμος ροής με μικρό χώρο για κάποιο πρόβλημα χρησιμοποιούμε αναγωγές της μορφής

Αλγόριθμος ροής με χώρο S $\Longrightarrow$ Πρωτόκολλο επικοινωνίας για υπολογισμό κάποιας f με μέγεθος μηνύματος S

και εκμεταλλευόμαστε γνωστά κάτω φράγματα της πολυπλοκότητας επικοινωνίας.

Λήμμα 1. *Για να στείλει έναν αριθμό από το* $\{1, \ldots, N\}$ *η Αλίκη χρειάζεται* $\Omega(\log N)$ *δυφία.*

Σαν άσκηση, θα χρησιμοποιήσουμε το παραπάνω λήμμα για να αποδείξουμε ξανά το 1.
Μπορούμε να φανταστούμε έναν ντετερμινιστικό αλγόριθμο για το πρόβλημα διακριτών στοιχείων ως εξής:

- Υποθέτουμε μια αρίθμηση στα υποσύνολα του $[N]$. Η Αλίκη όταν δεχθεί σαν είσοδο το i, βάζει το i-οστο υποσύνολο στην δομή δεδομένων και στέλνει την κατάσταση της δομής στον Μπάμπη.

- Ο Μπάμπης θέλει να βρει το i. Αν είναι σε θέση να το κάνει τότε από το 1 το μέγεθος της δομής είναι $\Omega(N)$. Πράγματι δεδομένου ότι ο αλγόριθμος είναι ακριβής, ο Μπάμπης μπορεί να βρει το i με τον ακόλουθο τρόπο. Για κάθε $j \in [N]$, εισάγει το στοιχείο j στην δομή που έλαβε από την Αλίκη και ελέγχει αν αυξήθηκε το πλήθος των διακριτών στοιχείων. Η απάντηση είναι αρνητική αν και μόνο αν το j ανήκει στο i-οστο σύνολο.

Θα αποδείξουμε το 1 με μια αντίστοιχη αναγωγή και βασιζόμενοι στο παρακάτω κάτω φράγμα για πιθανοτικά πρωτόκολλα επικοινωνίας.

Λήμμα 2. *Κάθε πιθανοτικό πρωτόκολλο επικοινωνίας μεταξύ Αλίκης και Μπάμπη, οπού η Αλίκη θέλει να στείλει έναν αριθμό στο* $[N]$ *στον Μπάμπη, και το οποίο πετυχαίνει με πιθανότητα* $2/3$*, χρειάζεται* $\Omega(\log N)$ *δυφία.*

Θα χρειαστούμε το παρακάτω αποτέλεσμα που αποδείξαμε σε προηγούμενο μάθημα μέσω της πιθανοτικής μεθόδου.

Λήμμα 3 (Κώδικας επιδιόρθωσης λαθών)**.** *Υπάρχουν* 2^{cn} *(με* $c \approx 1/500$*) υποσύνολα του* $[n]$ $A_1, A_2, \ldots$ *τέτοια ώστε* $|A_i| = \frac{n}{10}$ *για κάθε* i *και* $|A_i \cap A_j| \leq \frac{2n}{100}$ *για κάθε* $i \neq j$*.*

Απόδειξη Θεωρήματος 1. Πρώτα απ'όλα, επαναλαμβάνοντας με καινούρια τυχαία δυφία τη δομή δεδομένων, μπορούμε την πιθανότητα αποτυχίας να την κάνουμε οσοδήποτε μικρή, ας πούμε $\gamma = 10^{-9}$. Απο εδώ και στο εξής, θα υποθέσουμε ότι η πιθανότητα αποτυχίας της δομής σε ερώτηση είναι γ. Όπως και πριν, θεωρούμε ότι η Αλίκη θέλει να στείλει έναν αριθμό από το $[2^{cn}]$ στον Μπάμπη. Και οι δύο γράφουν όλα τα σύνολα $A_1, A_2, \ldots$. Όταν βλέπει το i η Αλίκη βάζει το σύνολο A_i στη δομή δεδομένων που λύνει το πρόβλημα Δ.Σ., και στέλνει την κατάσταση της μνήμης της δομής στον Μπάμπη.

Ο Μπάμπης θέλει να πάρει το i με πιθανότητα $2/3$. Έτσι όταν έχει πάρει μια δομή δεδομένων στην οποία η Αλίκη έχει εισάγει κάποιο σύνολο A_i κάνει το εξής. Εισάγει σε αυτήν ένα ένα τα στοιχεία του $[n]$, κάνοντας κάθε φορά ερώτηση στη δομή αν αυξήθηκε ή όχι το πλήθος το διακριτών στοιχείων. Έτσι παίρνει ότι:

- $j \in A_i \Rightarrow$ με πιθανότητα $1 - \gamma$ το πλήθος των Δ.Σ. δεν αυξήθηκε.
- $j \notin A_i \Rightarrow$ με πιθανότητα $1 - \gamma$ το πλήθος των Δ.Σ. αυξήθηκε.

Αν καλέσουμε X την τυχαία μεταβλητή που μετρά το πλήθος των στοιχείων j για τα οποία ο Μπάμπης πήρε το λάθος αποτέλεσμα από την δομή (δηλαδή είτε πήρε ότι το j ανήκει στο A_i ενώ $j \in A_i$, είτε πήρε ότι j δεν ανήκει στο A_i, ενώ $j \in A_i$ έχουμε

$$\mathbb{E}[X] \leq \gamma n.$$

Από την ανισότητα του Markov έχουμε

$$\mathbb{P}[X \geq 5\gamma n] \leq \frac{\mathbb{E}[X]}{5\gamma n} = 1/5.$$

Άρα έχει βρει το A_i συν/εκτός από $5\gamma n \approx n/10^9$ στοιχεία με πιθανότητα μεγαλύτερη του $4/5$. Δηλαδή αν S είναι η εικόνα που έχει ο Μπάμπης για το A_i, τότε $|S \triangle A_i| \leq \frac{5n}{10^9}$.

Όμως έχουμε $|A_i \triangle A_j| \geq \frac{16n}{100}$ και κατά συνέπεια $|S \triangle A_j| \geq \frac{16n}{100} - \frac{2n}{10^9} \gg |S \triangle A_i|$ για κάθε $j \neq i$. Άρα αν ο Μπάμπης διαλέξει το σύνολο που έχει τη μικρότερη συμμετρική διαφορά με το S ανακτά το A_i και κατά συνέπεια το i. Αυτό μπορεί να το κάνει με πιθανότητα $4/5$.

Από το Λήμμα 2 η δομή δεδομένων πρέπει να έχει μέγεθος $\Omega(\log 2^{cn}) = \Omega(n)$. □

2 Πρόβλημα ℓ_1 Βαρέων Στοιχείων (ℓ_1 Heavy Hitters)

Έστω $x = (x_1, \ldots, x_n)$, με $\|x\|_1 = \sum_{i\in[n]} |x_i|$. Στο πρόβλημα των βαρέων στοιχείων έχουμε ένα διάνυσμα $x \in \mathbb{R}^n$ και θέλουμε να διατηρήσουμε μια δομή δεδομένων που απαντάει στην παρακάτω ερώτηση: βρες μου εκείνα τα $i \in [n]$ ώστε $|x_i| \geq \epsilon\|x\|_1$. Δηλαδή, θέλουμε να υλοποιήσουμε τις παρακάτω μεθόδους. Ουσιαστικά ενδιαφερόμαστε για όλα εκείνα τα στοιχεία x_i του

Algorithm 1 Heavy Hitters

`initialize()`: $x \leftarrow 0$

`update`(i, Δ): $x_i \leftarrow x_i + \Delta$

`query()`: $\{i \in [n] : |x_i| \geq \varepsilon\|x\|_1\}$

διανύσματος x που αποτελούν ένα σημαντικό ποσοστό (καθορίζεται από το ε) της μάζας του x. Παρατηρούμε ότι το πλήθος αυτών των στοιχείων δεν ξεπερνά το $1/\varepsilon$. Πράγματι, έχουμε

$$\|x\|_1 \geq \sum_{i\in\text{output}} |x_i| \geq \sum_{i\in\text{output}} \varepsilon\|x\|_1 = \varepsilon\|x\|_1 \cdot |\text{output}| \Longrightarrow |\text{output}| \leq 1/\epsilon.$$

Επιπλέον διακρίνουμε τρία μοντέλα του προβλήματος:

(1) $\Delta = 1$ (2η σειρά ασκήσεων) (cash-register model).

(2) $\Delta =$ οτιδήποτε, αλλά όταν γίνεται ερώτηση $x_i \geq 0$ (strict turnstile).

(3) $\Delta =$ οτιδήποτε (turnstile).

CountMin Σκιαγράφημα

Περιοριζόμαστε στο μοντέλο strict turnstile, κατά συνέπεια $|x_i| = x_i$. Επιπροσθέτως, για να μπορέσουμε να σχεδιάσουμε υπογραμμικό αλγόριθμο θα τροποποιήσουμε το ζητούμενο, γιατί η απαίτηση να επιστρέφονται αποκλειστικά και μόνο τα στοιχεία με βάρος τουλάχιστον $\epsilon\|x\|_1$ είναι αρκετά ισχυρή. Συγκεκριμένα, θέλουμε αλγόριθμο που να επιστρέφει όλα τα $i \in [n]$ τέτοια ώστε $x_i \geq \varepsilon\|x\|_1$ και κανένα i με $x_i \leq \frac{\varepsilon}{2}\|x\|_1$. Όπως είδαμε υπάρχουν το πολύ $2/\varepsilon$ τέτοια στοιχεία.

2.0.1 Περιγραφή Αλγόριθμου

Παίρνουμε R ανά 2-ανεξάρτητες συναρτήσεις $h_r : [n] \to [\beta/\varepsilon]$ και κρατάμε $R \cdot \beta/\varepsilon$ μετρητές $C_{k,j}$ με $k \in [R]$, $j \in [\beta/\varepsilon]$, όπου β σταθερά.

$$\begin{matrix} C_{1,1} & C_{1,2} & \cdots & C_{1,\beta/\epsilon} \\ \vdots & \ddots & & \vdots \\ C_{R,1} & C_{R,2} & \cdots & C_{R,\beta/\epsilon} \end{matrix}$$

Όταν έρχεται μια ανανέωση (i, Δ) ο αλγόριθμος επαναλαμβάνει το ακόλουθο για κάθε $k \in [R]$

$$C_{k,h_k(i)} \leftarrow C_{k,h_k(i)} + \Delta.$$

Κατά συνέπεια έχουμε:

- Χώρος: $\overbrace{\left(R \cdot \frac{\beta}{\varepsilon}\right)}^{\text{μετρητές}} \log n + \overbrace{R \cdot O(\log n)}^{\text{'τυχαίες' συναρτήσεις}} = R \cdot O\left(\varepsilon^{-1} \log n\right)$.
- Χρόνος ανανέωσης: $R \cdot O(1) = O(R)$.

Στη συνέχεια περιγράφουμε το query του εν λόγω αλγορίθμου.

---

for $i \in [n]$ **do**
  if $C_{r,h_r(i)} \geq \varepsilon \|x\|_1 \ \forall r \in [R]$ **then**
    return i

Η τιμή $\|x_1\|_1$ είναι διαθέσιμη όντας το άθροισμα των τιμών των μετρητών μίας οποιαδήποτε γραμμής $i \in [R]$.

Ο χρόνος ερώτησης είναι $R \cdot O(n)$.

Ανάλυση CountMin Σκιαγραφήματος

Για δεδομένα $i \in [n]$ και $r \in [R]$ ορίζουμε την τυχαία μεταβλητή δ_i με

$$\delta_j = \begin{cases} 1, & \text{αν} \quad h_r(j) = h_r(i) \\ 0, & \text{αν} \quad h_r(j) \neq h_r(i) \end{cases}.$$

Έχουμε

$$\begin{aligned} \mathbb{E}\left[C_{r,h_r(i)}\right] &= \mathbb{E}\left[\sum_{j:h_r(j)=h_r(i)} x_j\right] = x_i + \mathbb{E}\left[\sum_{\substack{j \neq i \\ h_r(j)=h_r(i)}} x_j\right] + x_i + \mathbb{E}\left[\sum_{j \neq i} \delta_j x_j\right] \\ &= x_i + \sum_{j \neq i} \mathbb{E}\left[\delta_j\right] x_j = x_i + \sum_{j \neq i} \mathbb{P}\left[\delta_j = 1\right] x_j = x_i + \sum_{j \neq i} \frac{\varepsilon}{\beta} x_j \\ &\leq x_i + \frac{\varepsilon}{\beta} \|x\|_1. \end{aligned}$$

Έπεται ότι

$$\mathbb{E}\left[C_{r,h_r(i)} - x_i\right] \leq \frac{\varepsilon}{\beta} \|x\|_1$$

και από Markov

$$\mathbb{P}\left[C_{r,h_r(i)} - x_i \geq \frac{\varepsilon}{2} \|x\|_1\right] \leq \frac{2}{\beta}.$$

Επομένως έχουμε ότι

- Αν $x_i \geq \varepsilon\|x\|_1$, τότε $C_{r,h_r(i)} \geq \varepsilon\|x\|_1$.
- Αν $x_i \leq \varepsilon\|x\|_1$, τοτε με πιθανότητα $1 - \frac{2}{\beta}$ ισχύει ότι $C_{r,h_r(i)} < \frac{\varepsilon}{2}\|x\|_1 + \frac{\varepsilon}{2}\|x\|_1 = \varepsilon\|x\|_1$.

Τώρα μπορούμε να φράξουμε την πιθανότητα αποτυχίας του αλγορίθμου.

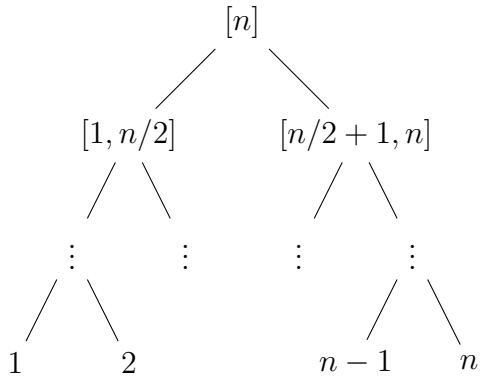
$$\begin{aligned}
&\mathbb{P}\left[\exists i \in [n] \text{ τέτοιο ώστε } x_i \leq \frac{\varepsilon}{2}\|x\|_1 \text{ και } \forall r \in [R]\ C_{r,h_r(i)} \geq \varepsilon\|x\|_1\right] \leq \\
&\leq \sum_{i\in[n]} \mathbb{P}\left[x_i \leq \frac{\varepsilon}{2}\|x\|_1 \text{ και } \forall r \in [R]\ C_{r,h_r(i)} \geq \varepsilon\|x\|_1\right] \\
&= n \cdot \left(1 - \frac{2}{\beta}\right)^R \overset{\beta=4}{=} \frac{n}{2^R}.
\end{aligned}$$

Επιλέγοντας το πλήθος των συναρτήσεων κατακερματισμού R ίσο με $\log(1/\delta)$ για $\delta = 1/10n$ (δηλαδή $R = O(\log n)$) η πιθανότητα αποτυχίας είναι το πολύ $1/10$. Κατά συνέπεια ο αλγόριθμος έχει πιθανότητα επιτυχίας $9/10$. Συνολικά έχουμε

- Χώρος: $O(\varepsilon^{-1}\log^2 n)$.
- Χρόνος ανανέωσης: $O(\log n)$.
- Χρόνος ερώτησης: $O(n \log n)$.

Ανταλλάσσοντας χώρο για χρόνο

Ο προηγούμενος αλγόριθμος έχει χρόνο ερώτησης $O(n \log n)$. Θα δούμε έναν τρόπο με τον οποίο μπορούμε να μειώσουμε τον χρόνο ερώτησης σε $O(\varepsilon^{-1}\log n\,(\log 1/\varepsilon + \log\log n))$ αυξάνοντας τον χώρο σε $O(\varepsilon^{-1}\log n\,(\log 1/\varepsilon + \log\log n))$.
Κατασκευάζουμε ένα δυαδικό δέντρο στο σύνολο $[n]$. Κάθε κόμβος του δέντρου αντιστοιχεί σε όλους τους κόμβους του αντίστοιχου υποδέντρου.



Θα κρατήσουμε ένα αντίγραφο του CountMin σκιαγραφήματος για κάθε ένα από τα $\log(n)$ επίπεδα επίπεδα του δέντρου. Ενδεικτικά για τα πρώτα δύο επίπεδα έχουμε

$$x^{(1)} = \left(\sum_{i\in[1,n/2]} x_i, \sum_{i\in[n/2+1,n]} x_i \right),$$
$$x^{(2)} = \left(\sum_{i\in[1,n/4]} x_i, \sum_{i\in[n/4+1,n/2]} x_i, \sum_{i\in[n/2+1,3n/4]} x_i, \sum_{i\in[3n/4+1,n]} x_i \right).$$

Επομένως ο χώρος που απαιτείται είναι $\sum_{i=1}^{\log n} \varepsilon^{-1} \log^2(n/2^i) = \Theta(\varepsilon^{-1} \log^3 n)$. Ο αλγόριθμος που ακολουθούμε είναι ο εξής:

- Τρέχουμε το σκιαγράφημα από στην ρίζα του δέντρου προς τα κάτω.
- Σε κάθε επίπεδο συνεχίζουμε να εξερευνούμε ένα υποδέντρο αν και μόνο αν ο αντίστοιχος κόμβος είναι βαρύς.
- Φτάνοντας στο τελευταίο επίπεδο επιστρέφουμε όσα φύλλα είναι βαριά.

Η ορθότητα του αλγόριθμου βασίζεται στις παρακάτω παρατηρήσεις.

- Για κάθε επίπεδο έχουμε $\|x^{(i)}\|_1 = \|x\|_1$ αφού η μάζα κάθε γονιού μοιράζεται στα παιδιά του.
- Βρισκόμενοι στο μοντέλο strict turnstile έχουμε ότι αν το παιδί είναι βαρύ τότε θα είναι κι ο γονιός.
- Από τα δύο προηγούμε παίρνουμε ότι σε κάθε επίπεδο υπάρχουν το πολύ $2/\varepsilon$ κόμβοι με μάζα μεγαλύτερη από $\varepsilon/2$.

Κατά συνέπεια ο μέγιστος αριθμός κόμβων για τους οποίους θα γίνει ερώτηση είναι $\frac{2}{\varepsilon}\log n = O(\varepsilon^{-1}\log n)$. Η κάθε ερώτηση παίρνει χρόνο R οπότε το συνολικό πλήθος ερωτήσεων είναι $R \cdot O(\varepsilon^{-1}\log n)$.

Αναγκαία συνθήκη για την αποτυχία του αλγόριθμου είναι η αποτυχία ενός ερωτήματος σε κάποιο από τα $4\varepsilon^{-1}\log n$ ερωτήματα που θα τεθούν για το αν ένας κόμβος είναι βαρύς ή όχι. Αν πετυχαίναμε πιθανότητα αποτυχίας κάθε ερωτήματος μικρότερη από $\frac{1}{100\varepsilon^{-1}\log n}$, τότε ο αλγόριθμος θα πετύχαινε με πιθανότητα $24/25$.

Η πιθανότητα αποτυχίας του κάθε ερωτήματος, όπως είδαμε, είναι μικρότερη από $1/2^R$ επομένως αρκεί να πάρουμε $4/\varepsilon R = O(\varepsilon^{-1}(\log(1/\varepsilon) + \log\log n))$ μετρητές σε κάθε επίπεδο. Αφού έχουμε $\log n$ επίπεδα ο συνολικός χώρος (σε μετρητές) γίνεται

$$O(\varepsilon^{-1}\log n\,(\log(1/\varepsilon) + \log\log n)).$$

Ο συνολικός χρόνος, δεδομένου ότι $R = O(\log(1/\varepsilon) + \log\log n)$, είναι

$$O(\varepsilon^{-1} \log n \, (\log(1/\varepsilon) + \log\log n))$$