# АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА САМОТНА ПЕЙКА

Наивното решение на задачата е да симулираме ситуацията – в едномерен масив с дължина n, в началото инициализиран с нули, пресмятаме кой човек сяда на съответното място, като на всяка стъпка намираме първата максимална последователност от свободни места. Ако на всяка стъпка започваме търсенето от началото, този подход има сложност $O(n^2)$, а ако на всяка стъпка намираме всички максимални последователности, и ги попълним последователно, сложността може да се покаже че е $O(n \log n)$. Това решение покрива първия (20%) вид тестове, където $n \leq 10000$.

В случая, при който $n = 2^k - 1$, задачата става напълно симетрична и може да се интерпретира като намиране на номерата в линеаризация на попълнено балансирано двоично дърво с $k$ слоя. Тази задача може да се реши по много начини – например с използване на рекурсивните зависимости между различните слоеве на дърво. Тук ще илюстрираме едно примерно решение, което използва главно побитови операции за $n = 7 = 2^3 - 1$:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 000 | 1000 | 100[0] | 100 | 100 | []100 | []100 | [0]100 | 100[0] | 000 |
| 001 | 1001 | 10[01] | 10 | 010 | [0]10 | [1]10 | [01]10 | 10[01] | 001 |
| 010 | 1010 | 101[0] | 101 | 101 | []101 | []101 | [0]101 | 101[0] | 010 |
| 011 | 1011 | 1[011] | 1 | 001 | [00]1 | [11]1 | [011]1 | 1[011] | 011 |
| 100 | 1100 | 110[0] | 110 | 110 | []110 | []110 | [0]110 | 110[0] | 100 |
| 101 | 1101 | 11[01] | 11 | 011 | [0]11 | [1]11 | [01]11 | 11[01] | 101 |
| 110 | 1110 | 111[0] | 111 | 111 | []111 | []111 | [0]111 | 111[0] | 110 |

1. *Записваме числата от 0 до $2^3 - 2$ в двоична бройна система (номерирали сме местата от пейката, започвайки от 0).*
2. *Добавяме „1“ отпред.*
3. *Образуваме група от десния край на числото до първата „0“ от дясно.*
4. *Изтриваме цифрите от групата.*
5. *Разглеждаме полученото като число в двоична бройна система – съответства на номера на човека, седнал на съответното място.*
6. *Образуваме група от левия край на числото до първата „1“ от ляво, без да я включваме в групата.*
7. *Заменяме в групата $0 \to 1$.*
8. *Добавяме „0“ отляво в групата.*
9. *Премества групата от дясната страна на числото.*
10. *Изтриваме „1“ отпред.*

Така получаваме алгоритъм със сложност $O(\log n)$ за този конкретен случай. Този алгоритъм ще се справи с втория (30%) вид тестове.

Да разгледаме сега задачата в общия случай.

При фиксирана големина на пейката $n$, да разгледаме намаляващата редицата $h_1, h_2, \ldots, h_n$, където $h_i$ е дължината на последователността от свободни места (която по-нататък ще наричаме „дупка“), в центъра на която сяда $i$-тия човек. Забелязваме, че $i$-тия човек сядайки в центъра на дупка с големина $h_i \geq 1$, я разделя на две дупки с големини:

$$l(h_i) \coloneqq \left\lfloor \frac{h_i - 1}{2} \right\rfloor \text{ и } r(h_i) \coloneqq \left\lceil \frac{h_i - 1}{2} \right\rceil.$$

Също така, всяка дупка в редицата е получена по този начин.

Нека с $H_s(n) = |\{i|h_i = s\}|$ да означим броят на дупките с големина $s$ в редицата. Тогава е в сила:

$$H_s(n) = \begin{cases} 0, \text{ако } n < s; \\ 1, \text{ако } n = s; \\ H_s\big(l(n)\big) + H_s\big(r(n)\big), \text{иначе.} \end{cases}$$

Основното наблюдение е, че по горната схема за пресмятането на $H_s(n)$ минаваме само през $O(\log n)$ различни други стойности на $H_s$. Това може да се докаже като се използва следното наблюдение: ако имаме три последователни естествени числа

$$1 < a < a + 1 < a + 2,$$

то прилагайки $l$ и $r$ към всяко получаваме стойности от интервала $[l(a), r(a + 2)]$, който съдържа най-много три последователни естествени числа. Така, „на всяка стъпка“, дължината на интервала остава ограничена от константата 3, а началото му се намалява двойно, значи общият брой цели числа, попадащи в някой от интервалите, е $O(\log n)$. Това показва също и че броят на различните цели числа измежду $h_1, h_2, \dots, h_n$, е най-много $3 \log n$ и че може да пресмятаме $H_s(n)$, за време $O(\log n)$. Понеже броят на различните големини на дупки е логаритмичен, можем да ги пресметнем, и да определим $h_i$ по дадено $i$ – това ще бъде е най-малката дължина на дупка $s$, за която броят на дупките $\sum_{s<t\leq n} H_t(n)$, с по-големи дължини е не по-голям от $i$.

След като локализирахме дължината на дупката на $i$-тия човек, остава да намерим и къде се намира на пейката. Нека с $B_s(n, j)$ означим началото (започвайки да броим от 0) на $j$-тата (започвайки от 0) дупка с дължина $s$ в пейка с $n$ места. Тогава имаме следната зависимост:

$$B_s(n, j) = \begin{cases} 0, \text{ако } s = n; \\ B_s(l(n), j), \text{ако } j < H_s\big(l(n)\big); \\ l(n) + 1 + B_s\Big(r(n), j - H_s\big(l(n)\big)\Big), \text{ако } j \geq H_s\big(l(n)\big). \end{cases}$$

По тази схема $B_s(n, j)$ се пресмята за време $O(\log^2 n)$. За да намерим на коя подред измежду всички дупки с дължина $h_i$ сяда $i$-тия човек, съобразяваме че всички хора преди него или са избирали по-големи дупки, или са избирали дупка със същата дължина, но започваща от място с по-малък индекс, т.е. $j = i - \sum_{s<t\leq n} H_t(n)$.

Знаейки дължината и началото на дупката, която е избрал $i$-тия човек, можем с проста аритметика да пресметнем мястото му върху пейката.

Да разгледаме втората подзадача – по дадено място на пейката $p$, търсим кой по ред човек сяда на него. Ако мястото е централно, е ясно, че първия човек сяда на него. В противен случай то е или вляво от центъра, или в дясно. Можем да си мислим че първия човек разделя пейката на две пейки с дължини $l(n)$ и $r(n)$.

Да разгледаме случая, в който мястото е вляво от центъра на пейката (другия случай е аналогичен). Рекурсивно решаваме задачата, и намираме, че $i_l$-тия човек сяда на място с номер $p$ в пейка с дължина $l(n)$. Сега можем, използвайки резултата от първата задача, да определим дължината $s$ и коя подред $k$ измежду дупките с една и съща дължина е дупката, в центъра на която е седнал $i_l$-тия човек в пейка с дължина $l(n)$.

Както в първата подзадача, за да намерим кой по ред човек сяда на мястото $p$ в пейка с $n$ места, определяме като:

$$k + \sum_{s<t\leq n} H_t(n).$$

Сложността в тази посока е $O(\log^3 n)$. Примерното решение използва memoization, и има сложност $O(\log^4 n)$.

*Автор: Красимир Георгиев*