АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА ПОДЧИСЛО

Да си изясним как се търси отговорът на една заявка. Дадени са K и L – трябва да се намери L-тата цифра на X_K. Да означим с $F(left, right, K)$ най-голямото K-цифрено подчисло на числото, определяно от интервала $[left, right]$ на числото X (при това една или няколко последователни цифри от това число могат да бъдат нули). Да означим също с $G(left, right, K, L)$ L-тата цифра на числото $F(left, right, K)$. Тогава отговорът на поредната заявка е равен на $G(1,N,K,L)$.

Най-напред да създадем за всяка цифра масив, който съдържа позициите, на които тя се среща в X.

Ще намираме $G(left, right, K, L)$ по следния начин. На интервала $[left, right]$ намираме най-голямата цифра D. Тъй като цифрите са 10, то можем със сложност $O(logN)$ да намерим за всяка от тях колко пъти се среща в интервала $[left, right]$ на числото X (използва се двоично търсене). Да означим броя срещания на D с r, а позициите на които се среща с $p_1, p_2, \ldots, p_r$.

Ако $r \geq K$, то $F(left, right, K)$ се състои само от цифри D и $G(left, right, K, L)=D$.

Ако $r<K$, то всички цифри D ще влязат в $F(left, right, K)$ - това може да се докаже чрез допускане на противното. Да разгледаме числата $t_1 < t_2 < \ldots < t_r$, такива, че $G(left, right, K, t_j)=D$. Налице са четири случая:

- $t_1 > L$. Тогава $G(left, right, K, L) = G(left, p_1-1, t_1-1, L)$;
- $t_r < L$. Тогава $G(left, right, K, L) = G(p_r+1, right, K-t_r, L- t_r)$;
- $t_j = L$ за някое j. Тогава $G(left, right, K, L) = D$;
- $t_{j-1} < L < t_j$ за някое j. Тогава $G(left, right, K, L) = G(p_{j-1}+1, p_j-1, t_j-t_{j-1}-1, L-t_{j-1})$.

Да забележим, че след прехода към новата, по-малка задача, на новия интервал няма нито една цифра D. По този начин изчисляването на $G(left, right,K,L)$ се свежда до изчисляването на $G(left', right',K',L')$, като най-голямата цифра на интервала $[left', right']$ е по-малка от най-голямата цифра на интервала $[left, right]$. Тъй като първоначалната най-голяма цифра е не по-голяма от 9, то за изчисляването на $G(left, right, K,L)$ може да се наложи най-много 9 пъти да се прави преход към по-малка задача.

Да изясним как се намират позициите $t_1, t_2, \ldots, t_r$. Тъй като това са позиции, на които стои максималната цифра за интервала $[left, right]$, то трябва да ги търсим колкото се може по-наляво в числото $F(left, right, K)$.

t_j се определя от отношенията $t_j \geq j$ и $right - p_j \geq K-t_j$ (за да останат достатъчно цифри за остатъка от числото $F(left, right, K)$; при това нас ни интересува възможно най-малкото значение. От тук се получава $t_j = max(j, K + p_j - right)$. Използвайки двоично търсене, може със сложност $O(logN)$ да се определи кой от четирите случая, описани по-горе има място.

И така отговорът на една заявка може да бъде даден със сложност $O(9logN)=O(logN)$, а на всичките M със сложност $O(MlogN)$.