АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА РОЗИ

Моделираме розовата градина с матрица A, имаща M реда и N колони. Всеки елемент $a_{i,j}$ на матрицата е количеството рози за храст от ред i и колона j в градината. Тогава задачата се свежда до намирането на

$$\sum_{\substack{i=1\\j=1}}^{\substack{i=M\\j=N}} a_{i,j} - \max\left(\forall\, sr \in \{1,2,\ldots,M-K+1\}, \forall\, sc \in \{1,2,\ldots,N-K+1\} \Rightarrow \sum_{\substack{r=sr\\c=sc}}^{\substack{r=sr+K-1\\c=sc+K-1}} a_{r,c} \right) \qquad (1)$$

Директно да се реализират сумиранията в (1) не е добра идея. Ако представим $K=C_K.N$ и $M=C_M.N$, то за намиране на първото събираемо в (1) са необходими $N.M=C_M.N^2$ събирания (сложност $\Theta(N^2)$), а за определяне на второто събираемо са необходими K^2 събирания за всеки от $(M-K+1).(N-K+1)$ варианта на разполагане на защитното покривало или $(C_K.N)^2.[(C_M-C_K).N+1].[(1-C_K).N+1]$ (сложност $\Theta(N^4)$) При стойности на K, M и N от порядъка 10^3 общото количество събирания в (1) има порядък 10^{12}, което предполага неприемливо време за решаване на задачата.

За да намалим сложността на решението прилагаме техника за бързо сумиране на елементите в подматрица на дадена матрица. Техниката се базира на следното:

Дефинираме матрица B, с елементи $b_{i,j} = \sum_{\substack{p=1\\q=1}}^{\substack{p=i\\q=j}} a_{p,q}$, $i \neq 0 \wedge j \neq 0$ и $b_{0,j} = b_{i,0} = 0$.

Тогава намирането на сумата от елементите за коя да е подматрица на матрицата A се свежда само до три събирания на определени елементи от матрицата B:

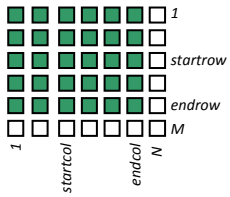
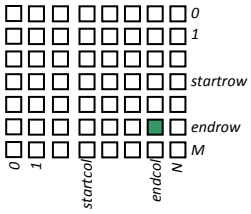
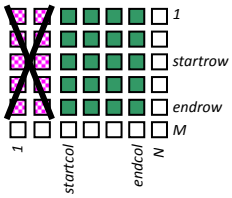
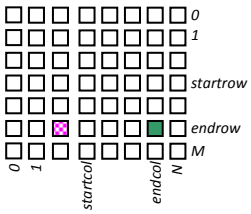
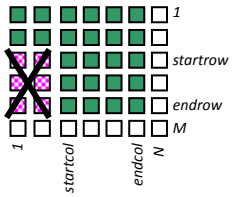
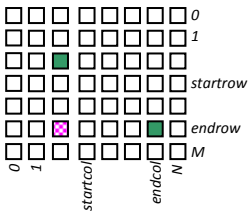
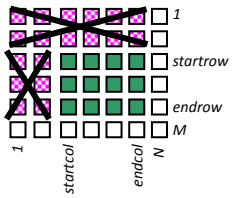
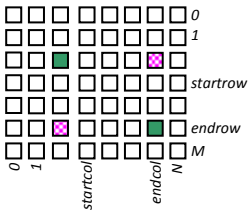
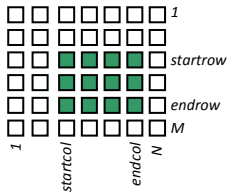
$$\sum_{\substack{p=startrow\\q=startcol}}^{\substack{p=endrow\\q=endcol}} a_{p,q} = b_{endrow,endcol} - b_{endrow,startcol-1} + b_{startrow-1,startcol-1} - b_{startrow-1,endcol} \qquad (2)$$

където $0 < startrow \leq endrow$, $0 < startcol \leq endcol$. Графична интерпретация на (2) е дадена в таблица по-долу. Понеже (2) има константна сложност, то сумиранията в (1) вече няма да зависят от N и общата сложност на решението се понижава до $\Theta(N^2)$, което изглежда приемливо при зададените ограничения.

Остава да реализираме ефективно определянето на $b_{i,j}$. Ще го правим „в движение" при четенето на $a_{i,j}$. Ако разгледаме следния вариант на (2)

$$a_{i,j} = \sum_{\substack{p=i\\q=j}}^{\substack{p=i\\q=j}} a_{p,q} = b_{i,j} - b_{i,j-1} + b_{i-1,j-1} - b_{i-1,j} \qquad (2')$$

като уравнение и го решим относно $b_{i,j}$ ще получим $b_{i,j} = a_{ij} + b_{i,j-1} - b_{i-1,j-1} + b_{i-1,j}$. Последното показва, че при реализиране на алгоритъма не е нужно да поддържаме матрицата A – нейните елементи при необходимост винаги може да се възстановят чрез (2').

Сумирани елементи от A в резултата	***Съответен елемент от B***	***Действие***
		Инициализация: $result \leftarrow b_{endrow,endcol}$
		$result \leftarrow resul - b_{endrow,startcol-1}$
		$result \leftarrow result + b_{startrow-1,startcol-1}$
		$result \leftarrow result - b_{startrow-1,endcol}$
	Окончателно: $result = \sum_{\substack{p=startrow \\ q=startcol}}^{\substack{p=endrow \\ q=endcol}} a_{p,q}$	

Автор: Евгений Василев,
Катедра Информатика, НПМГ