АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА КАПАЧКА

Листът хартия може да бъде представен чрез граф, чийто върхове са защрихованите квадратчета. Между два върха има ребро тогава и само тогава, когато съответните им квадратчета лежат на един ред или на един стълб.

Естествен подход при решаването на задачата е да търсим най-краткия път от началното защриховано квадратче до крайното такова чрез обхождане в ширина, започвайки от началното квадратче. Обхождането в ширина има сложност $O(V+E)$, където V е броя на върховете в графа, а E- брот на ребрата. В нашия случай броят на върховете не надвишава NM, но броят на ребрата може да достигне $NM(N+M-2)/2$. Така че този алгоритъм ще донесе 60 точки.

Първо решение за 100 точки.

Алгоритъмът може да се ускори значително, ако се приложи следното съображение: ако на някаква стъпка от обхождането в ширина посетим квадратче C, а преди това е било посетено поне едно квадратче D, което се намира в същия ред като квадратчето C, то няма смисъл да се разглеждат ребрата на графа, които свързват квадратчето C с други квадратчета от същия ред. Действително, нека D е първото защтриховано квадратче от даден ред, което е посетено при обхождането в дълбочина. Тогава в опашката на обхождането в дълбочина ще попаднат всички защриховани квадратчета от този ред. Когато разглеждаме някое квадратче C от тези квадратчета, то разглеждането на ребро (C,E) няма да добави нищо в опашката, тъй като или $E=D$, или E е било добавено в опашката по-рано, при разглеждането на реброто (D,E). Така че, можем да не разглеждаме голямо количество „допълнителни“ ребра – достатъчно е да помним квадратчетата, в които алгоритъмът е влизал в съответните редове. Естествено, същите съображения важат не само за редовете, но и за стълбовете на таблицата.

Така стигаме до извода, че за всеки ред ще трябва да се разгледат максимум $M-1$ ребра, а за всеки стълб максимум $N-1$ ребра (това са ребрата, които излизат от някое единствено квадратче на реда или стълба). Такъв алгоритъм разглежда $O(NM)$ ребра и съответно сложността му е $O(NM)$. Той води до решение за 100 точки..

Второ решение за 100 точки.

Отново се строи граф, но по различен начин. Нека върховете $a_1, a_2, \ldots, a_N$ съответстват на редовете $1, 2, \ldots, N$ на таблицата, а върховете $b_1, b_2, \ldots, b_M$ на стълбовете на таблицата. Между върховете a_i и b_j има ребро тогава и само тогава, когато квадратчето, което се намира на пресечната точка на съответните ред и стълб, е защтриховано. Други ребра в графа няма. Така че този граф има $N+M$ върха и не повече от NM ребра.

Да поставим в съответствие на хода от квадратче $(i,j1)$ в квадратче $(i,j2)$ придвижване по ребро (a_i,b_{j2}) на графа, а на хода от квадратче $(i1,j)$ в квадратче $(i2,j)$ придвижване по ребро (b_j,a_{i2}). Тъй като по най-краткия маршрут между квадратчета A и B не може да има два последователни хода в рамките на един ред или един стълб (тъй като те могат да бъдат заменени с един ход, който ще дава същия резултат), то на оптималния маршрут между квадратчетата A и B отговаря маршрут по ребрата на графа, който изглежда по следния начин: нека началното и крайното квадратчета имат следните координати A=($i1$,$j1$) и B=($i2$,$j2$).

Тогава пътят започва или във връх a_{i1}, или във връх b_{j1} и завършва с ребро, което свързва върховете a_{i2} и b_{j2} (това ребро се преминава или в посока от a_{i2} към b_{j2}, или обратно). И обратно, на всеки такъв път по ребрата на графа съответства маршрут със същата дължина между квадратчетата *A* и *B*.

И така, задачата се свежда до намирането на най-къс път, който започва в един от върховете a_{i1} или b_{j1} и завършва с ребро, което свързва върховете a_{i2} и b_{j2} (това ребро се преминава или в посока от a_{i2} към b_{j2}, или обратно). Ако означим с $d(x,y)$ дължината на най-късия път между върховете *x* и *y*, то търсената велина е равна на

$$min\{d(a_{i1}, a_{i2}), d(a_{i1}, b_{j2}), d(b_{j1}, a_{i2}), d(b_{j1}, b_{j2})\}+1$$

От тук се вижда, че търсената минимална дължина може да се намери с две обхождания в ширина – едното с начало във върха a_{i1}, а другото с начало във върха b_{j1}). Може да се мине и с едно обхождане, ако въведем един фиктивен връх *c* и го свържем с върховете a_{i1} и b_{j1}. Тогава задачата се свежда до намирането на по-късия от двата най-къси пътя – от *c* до a_{i2} и от *c* до b_{j2}, т.е. търси се $\min\{d(c, a_{i2}), d(c, b_{j2})$.

Сложността на този алгоритъм е $O(NM)$.