Túto úlohu môžme riešiť nasledovne: Použijeme BFS na nájdenie najkratšej cesty. Vždy keď sa chceme pohnúť do nejakej strany, tak si vypočítame či tam môžme ísť. Toto však nezbehlo na 1 vstupe(neviem prečo len na jednom)(kôli TLE), preto treba vymyslieť niečo lepšie. Čo keby sme si všetky možné oblasti pre všetky možné pozície predrátali dopredu?? Koľko času by nás to stálo ??

Vieme to spraviť nasledovne:

Najskôr si vyrátame pre každý riadok súčet dĺžky a,b,c - toto vieme spraviť v lieárnom čase od dĺžky riadku. A potom vyrátame súčet pre každý obdĺžnik rozmerov ab,ba,ac,ca,bc,cb. Keď máme predrátané riadky, tak nás to bude stáť lineárny čas od dĺžky stĺpca. Takže celkové predrátanie nás bude stáť O(r.s) času. BFS nám zoberie tiež O(r.s), takže celková časová zložitosť bude O(r.s). Pamätať si musíme plánik, a pre každé políčko mriežky všetkých 6 súčtov jednotlivých polôh. Takže O(r.s) je pamäť.

Pozíciu kvádra si pamätám ako že pozíciu jeho ľavého horného rohu a šírny a dĺžky podstavy. Ak má kváder rozmery podstavy aktuálne a.b, tak ak ho posuniem hore, alebo dole, tak potom bude mať rozmery c.b a keď ho preklopím doľava, alebo doprava, tak bude mať rozmery a.c. Takže teraz viem jednoducho vypočítať ktorú oblasť mám kontroloívať či neobsahuje dieru.

Riešenie dá vždy správny výsledok, pretože určite nájdeme najkratšiu cestu, pretože používame BFS - ktoré nám zaručene nájde najkratšiu cestu.

Pseudo kód:

```
funkcia riadok(rozmer){
        for i:=1 to r do
                for j:=1 to s do
                        odcitaj pole[j-rozmer];
                        pricitaj pole[j];
                        zapis to do riadky[rozmer][i];

}
//pre kazdy rozmer kvadra si vypocitame sucty v riadkoch dlzky postupne a,b,c

funkcia sucet(pismeno1, pismeno 2)
        for i:=1 to s do
                for j:=1 to r do
                        odcitaj riadky[pismeno1][j-pismeno2];
                        pricitaj riadky[pismeno1][j];
                        zapis to do sucty[pismeno1][pismeno2];

//tu vypocitame uz s existujucich riadkov sucty stvorcov

riadok(a);
riadok(b);
riadok(c);

//takto nejak sa vola ta funkcia na riadky
sucet(a,b);
sucet(b,a);
sucet(a,c);
sucet(c,a);
```

```
sucet(b,c);
sucet(c,b);

// a takto nejak ta na sucty
```

//BFS

vlož do fronty pozíciu 0,0
skontroluj, či neleží na diere(ak plánik je menší ako podstava)

pokiaľ fronta nieje prázdna:

vyber z fronty pozíciu, skontroluj, či si neprišiel do cieľa, ak nie tak pohni sa na všetky miesta kam môžeš, ak ešte neboli navstívené a označ ich ako navšťívené.

vypíš výsledok