Keď sa pozrieme na nejaký potok a chceme ho preskočiť, tak vidíme, že nemá význam skákať z políčka váčšieho ako a, alebo menšieho ako b-p. Inak by sme potok nepreskočili. Takže stačí vždy skontrolovať pre každý potok týchto b-p políčok a spraviť skoky z týchto políčok. Na tieto políčka sa vieme dostať bud nejakým skokom(posledný pohyb), alebo niekoľkými krokmi z predchádzajúcich políčok, ale iba z tých ktoré sú väčšie ako b minulého potoka,a lebo 0. Na políčka z intervalu b minulého potoka a a súčasného na ktoré sa vieme dostať posledným pohybom skokom, prejdeme práve raz, pretože v inom intrevale nás už netrápia, pretože sa z nich iným pohybom ako skokom nedokážene dostať.

Takže stačí nám jedna queue do ktorej hádžeme naše pozície z políčok max(b minulého, b-p) až a-1. Pre dané políčko si zistíme na koľko najmenej skokov sa tam vieme dostať tak, že je to minimum z intrevalu b minulého až pozícia políčka, ktorú si vieme jednoducho vyrátať minimum z vyhádzania nejakých prvkov z fronty ktoré majú pozície menšie alebo rovné a z predchádzajúceho políčka. Potom ešte vyhádžeme všetky políčka kde sa viem dostať z potoka a môžem ísť na ďalší potok. Pre každý potok prejdem p políčok, takže časová zložitosť je O(n*p) - čo je menšie ako 10^8 takže to stíham a pamäťová je O(p) - pretože ak by bolo vo fronte viac políčok/prvkov, tak potom som tam nemohol pushnuť ten posledný, lebo ten je vzdialený od prvého aspoň o p, takže ten prvý by som musel vymazať. Takže O(p).

Riešenie dá určite správny výsledok, pretože iná lepšia trasa nemôže existovať, pretože spontrolujem všetky políčka z ktorých by sa optimálna trasa skladala, takže nemôže byť lepšia.

Pseudo kód

```
pre kazdy potok i
        pre kazde policko j od max(minule_b,b-p)
                fronta.push(j+p,min(mensie policka
                 vo fronte ako ja,pocet minuleho policka))

                vypushuj vsetky policka medzi a a b


prejdi vsetky policka za poslednym potokom a najdi minimum,
 ak neexituje, vypis -1
```