**TEHNIČKO VELEUČILIŠTE U ZAGREBU**

**POLITEHNIČKI SPECIJALISTIČKI DIPLOMSKI STRUČNI STUDIJ**

SPECIJALIZACIJA INFORMATIKA

Stjepan Salopek

**RJEŠAVANJE LABIRINATA  
A\* ALGORITMOM PRETRAGE**

Zagreb, listopad 2023.

Uvod

Algoritmi pretrage pokazali su se vrlo korisnima u današnjem svijetu punom informacija, u kojem je iznimno bitno pronaći što točniju informaciju, u što kraćem roku, sa što manje resursa.

Postoji mnogo slučajeva gdje se algoritmi pretrage mogu koristiti, a najčešće se radi o pronalasku optimalnog puta, bio on stvarni ili virtualni. Primjerice, često se mogu pronaći u labirintima i društvenim igrama kao što je šah, u procesuiranju prirodnog jezika, usmjeravanju paketa u mreži i tako dalje.

Seminarski rad obraditi će najčešće algoritme pretrage i navesti njihove prednosti i mane. Nadalje, ući će se u određenu dubinu sa A\* algoritmom i obrazložiti razlog njegove kvalitete. Na praktičnom primjeru pokazati će se njegova jednostavnost i snaga. Nakon obrade algoritama, uvesti će se u metodu generiranja algoritama u programskom jeziku Python i načina na koji se prikazuju.

Imajući na umu informacije o algoritmima i način rada sa labirintima, pomoću A\* algoritma pretrage na jednostavan način će se pronaći najkraći put u labirintu.

Algoritmi pretrage

Algoritam pretrage je skup metoda i procedura namijenjenih istraživanju skupu mogućih rješenja koje nazivamo problemskim prostorom [1]. Često se koriste u problemima povezanim sa odabirom optimalnog ili najkraćeg puta, a mogu se podijeliti na informirane i neinformirane.

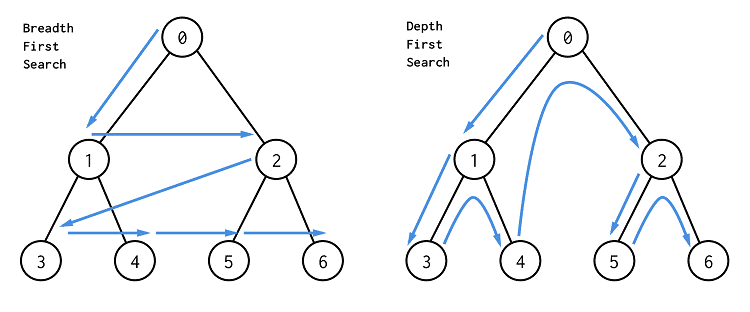
Neinformirani algoritmi imaju informaciju o oznaci cilja do kojeg trebaju doći, no ne znaju njegove koordinate. Informirani algoritmi imaju informaciju o oznaci cilja i njegovoj poziciji, a zadatak im je pronaći najkraći put.

BFS i DFS algoritmi pretrage

*Breadth-First Search* ili pretraživanje u širinu te *Depth-First Search* ili pretraživanje u dubinusu neinformirani algoritmi pretrage. Problemski prostor promatraju kao stablo odluka, u kojemu je korijenski čvor početno stanje algoritma, dok potomci korijena predstavljaju sve moguće opcije i putove kojima se može ići da bi se došlo do cilja.

Pretraživanje u širinu traži rješenje razinu po razinu, a koristi podatkovnu strukturu reda koji radi na principu *First In, First Out*. U početku svu djecu korijenskog čvora slijedno stavlja u red. Prilikom svake sljedeće iteracije, djecu prvog člana reda će staviti na kraj te ga nakon obrade izbaciti. Na ovaj način postiglo se pretraživanje u širinu.

Pretraživanje u dubinu istražuje jednu po jednu opciju, a na sljedeću prelazi tek kada dođe do slijepe ulice određenog puta. Koristi podatkovnu strukturu stoga koji radi na principu *Last In, First Out*. Uzimajući i obrađujući uvijek posljednjeg dodanog člana, postiže se pretraživanje u dubinu.



Slika 1 Usporedba DFS i BFS algoritama pretraživanja

A\* algoritam pretrage

Glavna mana kod DFS i BFS algoritama je njihova neinformiranost. S obzirom da to da nemaju uvid u lokaciju cilja, postoji velika vjerojatnost da će nepotrebno potrošiti mnogo računalnih resursa. Primjerice, prilikom traženja najkraćeg puta od Zagreba do Ljubljane, oba algoritma zanimati će i put koji vodi prema Slavoniji, iako bi bilo poželjno istovremeno ga ignorirati.

Da bi se izbjegao ovaj problem, koristi se A\* algoritam pretrage koji koristi lokaciju cilja kao nit vodilju za odabir najkraće rute. Na primjeru puta do Slovenije, algoritam bi odabrao i manje kvalitetnu cestu koja vodi prema Sloveniji umjesto kvalitetnije koja vodi prema Slavoniji, zato jer je preko funkcije određivanja kvalitete uočio da bi ga potonja udaljila od cilja.