INTELIGENTNI SISTEMI

as. ms Vladimir Jocović
as. ms Adrian Milaković

ALGORITMI PRETRAŽIVANJA

01

„Not all those who wander are lost.“
- J. R. R. Tolkien

DEFINISANJE PROBLEMA PRETRAŽIVANJA

Šta sve definiše problem pretraživanja?

- **Inicijalno stanje** – stanje predstavlja opis trenutne situacije u problemu. Startno stanje je ono od kojeg polazimo.
- **Operatori** – skup akcija koje je moguće preduzeti u određenom stanju i koje utiču na (eventualni) prelazak u drugo stanje.
- **Tranzicioni model** – opis efekata primene raspoloživih akcija u svakom mogućem stanju.

Inicijalno stanje, operatori i tranzicioni model zajedno definišu **prostor stanja problema**. On se može predstaviti grafom, pri čemu su stanja predstavljena čvorovima grafa, a tranzicije (operatori) granama grafa.

DEFINISANJE PROBLEMA PRETRAŽIVANJA

Šta sve definiše problem pretraživanja?

Putanja u prostoru pretraživanja predstavlja sekvencu stanja povezanih akcijama kojima se prelazi iz jednog stanja u naredno.

- **Ciljno stanje** – krajnje stanje u koje je potrebno doći.
- **Cena putanje** – dodeljena je svakoj putanji i utiče na cenu rešenja.
- **Pretraživanje** – proces utvrđivanja sekvence operatora (putanje) koja nas dovodi od startnog do ciljnog stanja. Takva sekvenca se naziva **rešenje problema**.
- **Optimalno rešenje problema** je ono čija je cena (putanje) najmanja.

Zadatak 1 – Hanojske kule

★☆☆☆☆

Hanojska kula je matematička igra koja se (u bazičnom slučaju) sastoji od 3 vertikalna štapa i 2 diska različitih poluprečnika. Diskovi se mogu stavljati na bilo koji štap. Igra počinje sa diskovima postavljenim na prvom štapu, po opadajućoj veličini diskova. Cilj igre je da se svi diskovi sa jednog štapa premeste na drugi poštujući sledeća pravila:

- Samo jedan disk može da se pomera istovremeno.
- Svaki potez se sastoji od uzimanja gornjeg diska sa jedne gomile i stavljanja tog istog diska na vrh druge gomile, odnosno disk može da se pomera samo ako je na poslednjem mestu na štapu.
- Nijedan disk ne sme biti smešten na manji disk na štapu.

Zadatak 1 – Rešenje

Kako predstaviti stanje problema?

- (x, y, z) – x, y i z predstavljaju broj diskova koji se nalaze na štapovima 1, 2 i 3 respektivno; *pr. (2, 0, 0)*
- (a, b, c, d, e, f) – a i b predstavljaju indikator prisustva većeg, odnosno manjeg diska na štapu 1, c i d na štapu 2, e i f na štapu 3; *pr. (1, 1, 0, 0, 0, 0)*
- (x : a-b, y : c-d) – disk x se nalazi kao a-ti po redu odozdo na štapu b, dok se disk y nalazi c-ti po redu odozdo na štapu d; *pr. (manji : 2 – 1, veći : 1 – 1)*
- (x, y) – x predstavlja broj štapa na kome se nalazi veći disk, dok y broj štapa na kome se nalazi manji disk; *pr. (1, 1)*
- *Da li svi predlozi daju potpune informacije? Dodatni predlog?*

Zadatak 1 – Rešenje

Usvojićemo notaciju **(x, y)** sa prethodnog slajda, gde **x** predstavlja broj štapa na kome se nalazi veći disk, dok **y** predstavlja broj štapa na kome se nalazi manji disk.

Potrebno je:

- Odrediti sva dozvoljena stanja u prostoru problema.
- Formirati tabelu dozvoljenih prelaza između svih stanja u prostoru problema.
- Prikazati kompletan graf pretrage za navedeni problem.
- Prikazati kompletno stablo pretrage za navedeni problem.

Zadatak 1 – Rešenje

Dozvoljena stanja u prostoru problema

Moguća stanja su sva ona stanja **(x, y)** gde **x, y ∈ { 1, 2, 3 }**. Takvih stanja ima ukupno 9 i to su:

(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3)

Startno stanje je **(1, 1)** dok stanje **(3, 3)** predstavlja ciljno stanje.

U slučaju stanja **(x, x)** podrazumevaćemo da se manji disk nalazi na većem, jer je to jedino dozvoljeno stanje (nije dozvoljeno da se veći disk nalazi na manjem).

Tabela dozvoljenih prelaza između stanja

U koja stanja je moguće preći iz stanja ***(1, 1)****?*

U stanje **(1, 2)** i u stanje **(1, 3)**. Veliki disk nije moguće pomerati.

U koja stanja je moguće preći iz stanja ***(1, 2)****?*

U stanje **(1, 1)**, stanje **(1, 3)** i u stanje **(3, 2)**.

Da li je moguće preći u stanje ***(2, 2)*** *iz stanja* ***(1, 2)****?*

Takvo stanje je dozvoljeno, ali nije moguće napraviti prelaz u njega iz stanja **(1, 2)**, jer se veliki disk ne može naći na malom disku.

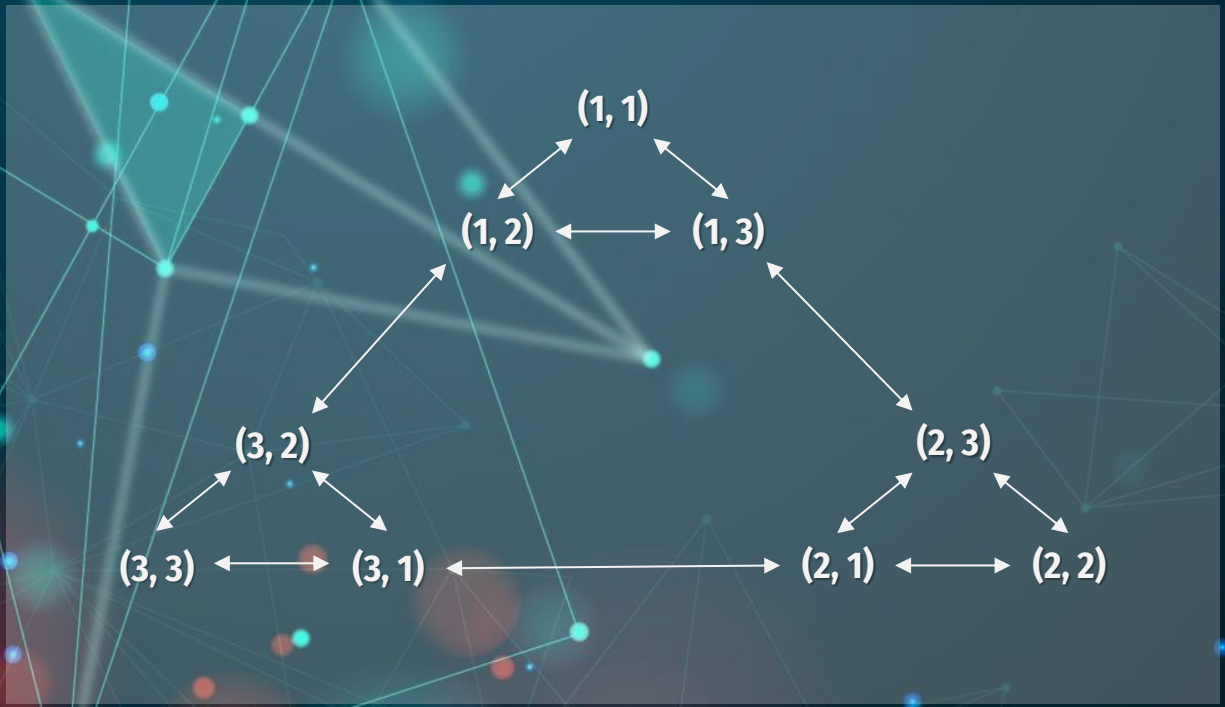
Tabela dozvoljenih prelaza između stanja

Prelaze je najlakše predstaviti i vizualizovati tabelom n *x* n, gde n predstavlja broj mogućih stanja u prostoru problema. Primetiti da je tabela u ovom slučaju simetrična u odnosu na dijagonale.

	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)	(3, 1)	(3, 2)	(3, 3)
(1, 1)		✓	✓						
(1, 2)	✓		✓					✓	
(1, 3)	✓	✓				✓			
(2, 1)					✓	✓	✓		
(2, 2)				✓		✓			
(2, 3)			✓	✓	✓				
(3, 1)				✓				✓	✓
(3, 2)		✓					✓		✓
(3, 3)							✓	✓	

Kompletan graf pretrage problema Hanojskih kula

Graf predstavlja prostor stanja problema i konstruiše se na osnovu tranzicione tabele (tabele prelaza). Akcije su predstavljene granama grafa, a stanja čvorovima grafa.



Zadatak 1 – Rešenje

Prikazati kompletno stablo pretrage

Nakon definisanja problema, potrebno ga je rešiti, odnosno pronaći sekvencu akcija koja vodi od startnog do ciljnog stanja. Moguće sekvence akcija koje započinju od startnog stanja formiraju stablo pretrage, gde startno stanje predstavlja koren takvog stabla. Ostala stanja su čvorovi stabla (unutrašnji ili listovi), a grane su akcije koje vode iz jednog stanja u drugo.

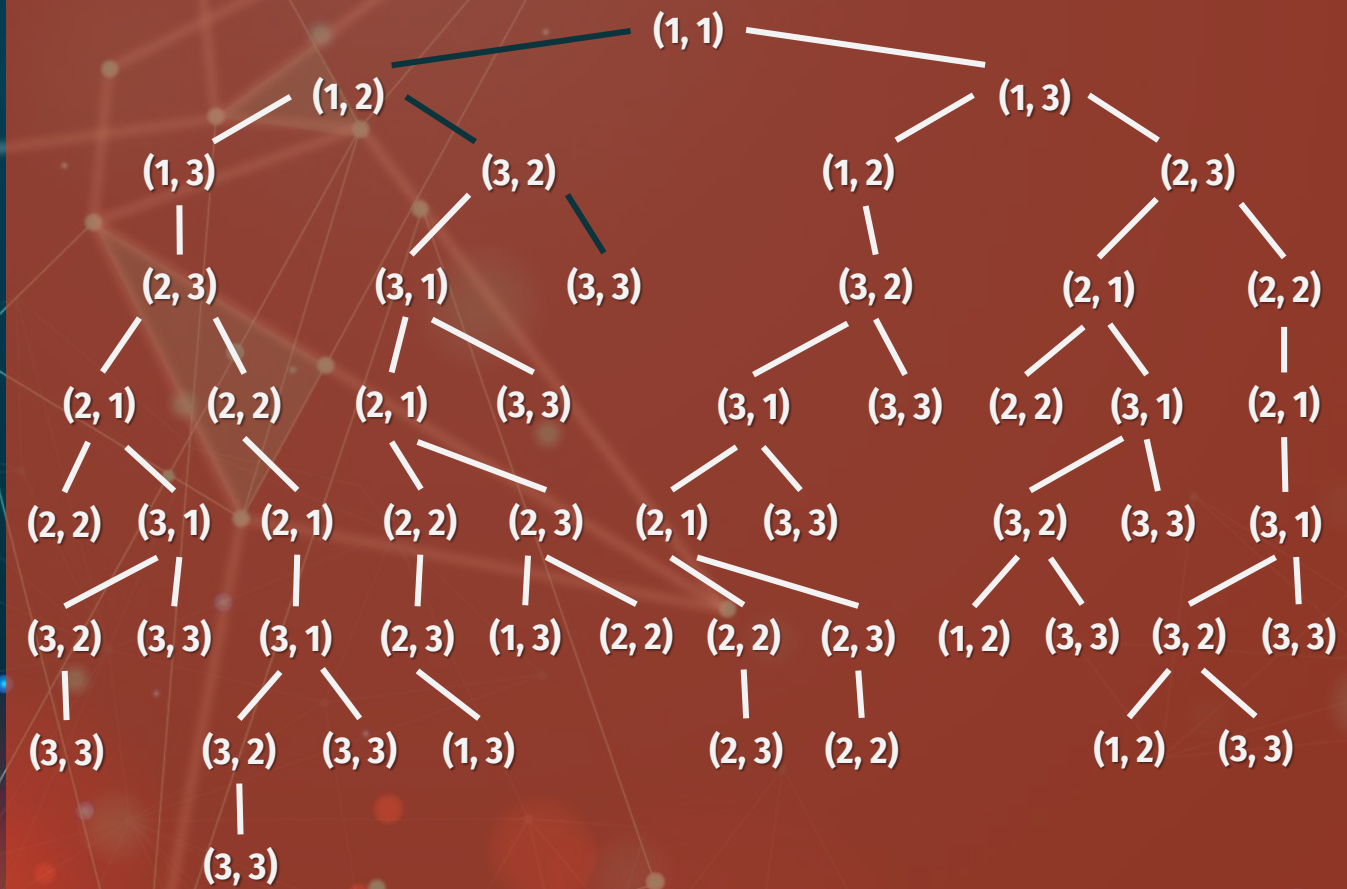
Proces generisanja novih čvorova (stanja) razmatranjem legalnih opcija (akcija) u tekućem čvoru (stanju) naziva se **ekspandovanje čvora**.

Zadatak 1 – Rešenje

Prikazati kompletno stablo pretrage

Kompletno stablo pretrage kreira se počevši od startnog čvora grafa pretrage, koje se dodaje u stablo kao njegov koren. Primenom svih legalnih akcija u tom stanju u stablo se dodaju čvorovi do kojih te akcije vode. Zatim se za novododate čvorove primenjuju sve legalne akcije za svaki od tih novododatih čvorova i dodaju se novi čvorovi u stablo. Postupak se ponavlja dok god se ne generiše/obiđe ciljni čvor ili dok god ima neekspandovanih čvorova.

Kompletno stablo pretrage ne sadrži petlje grafa na osnovu kojeg se ono kreira i čvorovi koji bi doveli do pravljenja petlje u stablu se ne dodaju u njega (nema *loopy* putanja).

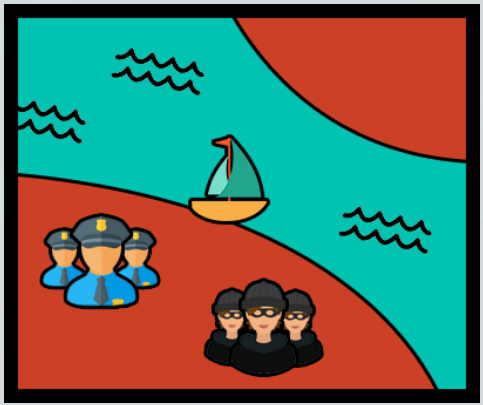

(1, 1)
(1, 2)
(1, 3)
(1, 3)
(3, 2)
(1, 2)
(2, 3)
(2, 3)
(3, 1)
(3, 3)
(3, 2)
(2, 1)
(2, 2)
(2, 1)
(2, 2)
(2, 1)
(3, 3)
(3, 1)
(3, 3)
(2, 2)
(3, 1)
(2, 1)
(2, 2)
(3, 1)
(2, 1)
(2, 2)
(2, 3)
(2, 1)
(3, 3)
(3, 2)
(3, 3)
(3, 1)
(3, 2)
(3, 3)
(3, 1)
(2, 3)
(1, 3)
(2, 2)
(2, 2)
(2, 3)
(1, 2)
(3, 3)
(3, 2)
(3, 3)
(3, 3)
(3, 2)
(3, 3)
(1, 3)
(2, 3)
(2, 2)
(1, 2)
(3, 3)
(3, 3)

Zadatak za samostalnu vežbu – Čamac

★☆☆☆☆

Tri policajca i tri huligana nalaze se na levoj obali reke koju treba da pređu. Na raspolaganju im je čamac u koji staju najviše dve osobe. Ako u nekom trenutku broj huligana nadmaši broj policajaca na levoj ili desnoj obali, huligani će se potući sa policajcima. Cilj je da svi bezbedno pređu reku.

- Predstaviti na pogodan način stanje problema. Smatrati da se čamac uvek nalazi na nekoj od obala.
- Koliki je ukupan broj stanja u prostoru problema?
- Koliko stanja je bezbedno po policajce i koja su to stanja?



Zadatak za samostalnu vežbu - Rešenje

Predstaviti na pogodan način stanje problema. Smatrati da se čamac uvek nalazi na nekoj od obala.

Stanje problema možemo definisati uređenom trojkom **(p, h, č)**, gde **p** predstavlja broj policajaca na levoj obali **{0, 1, 2, 3}**, **h** broj huligana na levoj obali **{0, 1, 2, 3}**, a **č** obalu na kojoj se čamac nalazi (**0 - leva, 1 - desna**). Nije potrebno pamtiti individualnosti osoba. Broj osoba na desnoj obali može se izračunati oduzimanjem broja osoba na levoj od ukupnog broja osoba. Broj osoba u čamcu je pridružen broju osoba na obali na kojoj se čamac nalazi.

Koliki je ukupan broj stanja u prostoru problema?

Ukupan broj stanja je **4 * 4 * 2 = 32**, jer p i h mogu uzeti neku od 4 različite vrednosti, a č neku od 2.

Zadatak za samostalnu vežbu - Rešenje

Koliko stanja je bezbedno po policajce i koja su to stanja?

Stanja u kojima su svi policajci na jednoj od obala:

(0, 0, 0), **(0, 0, 1)**, (0, 1, 0), (0, 1, 1), (0, 2, 0), (0, 2, 1), (0, 3, 0), (0, 3, 1) – svi policajci na desnoj obali.

(3, 0, 0), (3, 0, 1), (3, 1, 0), (3, 1, 1), (3, 2, 0), (3, 2, 1), **(3, 3, 0)**, **(3, 3, 1)** – svi policajci na levoj obali.

Stanja u kojima se policajci nalaze na obe obale, ali je njihov broj jednak broju huligana na svakoj od obala.

(0, 0, 0), **(0, 0, 1)**, (1, 1, 0), (1, 1, 1), (2, 2, 0), (2, 2, 1), **(3, 3, 0)**, **(3, 3, 1)**

Ukupno 20 različitih stanja.

Zadatak 2 – Putna mreža

★★☆☆☆

Na slici je prikazan deo putne mreže sa označenim dužinama puteva. Procenjena udaljenost od pojedinih gradova do ciljnog grada G data su u tabeli.

Zadatak 2 – Putna mreža

★★☆☆☆

Prikazati stablo pretrage i navesti redosled obilaženja čvorova prilikom pronalaženja putanje između gradova S i G, ukoliko se koriste sledeći algoritmi:

- Pretraga po dubini (*depth first search*)
- Pretraga po širini (*breadth first search*)
- Prvo najbolji (*best first search*)
- Grananje i ograničavanje (*branch and bound search*)
- A* (*A star*)
- ID A* (*Iterative deepening A**)

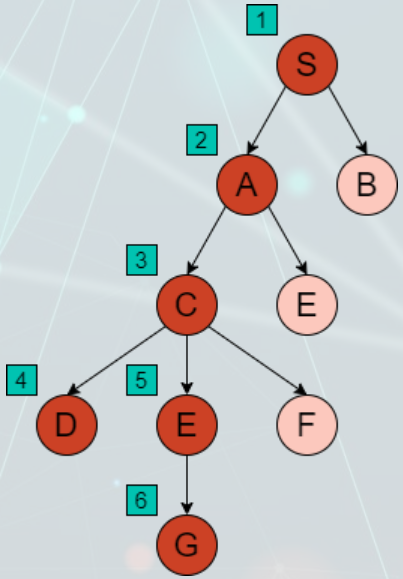
Zadatak 2 – Rešenje

Pretraga po dubini (*depth first search*):

- Lista čvorova sadrži startni čvor.
- Dokle god je lista čvorova neprazna:
  - Uklanja se čvor sa početka liste i ispituje se da li je ciljni.
    - Ako je u pitanju ciljni čvor, pretraga je završena.
    - Ako nije u pitanju ciljni čvor, dodati njegove sledbenike (ukoliko postoje) u odgovarajućem redosledu na **početak** liste.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 2 – Rešenje

Pretraga po dubini (*depth first search*):



putanja: SACEG
cena putanje: 17

Zadatak 2 – Rešenje

Pretraga po dubini (*depth first search*):

Da li je pronađena putanja optimalna?

Pronađena putanja nije optimalna i DFS ne garantuje pronalaženje optimalnog rešenja.

Koja struktura podataka odgovara DFS algoritmu?

LIFO (*Last in first out*) struktura podataka, odnosno stek.

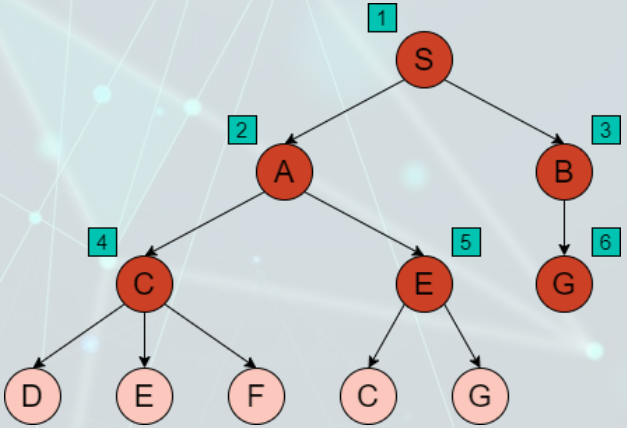
Zadatak 2 – Rešenje

Pretraga po širini (*breadth first search*):

- Lista čvorova sadrži startni čvor.
- Dokle god je lista čvorova neprazna:
  - Uklanja se čvor sa početka liste i ispituje se da li je ciljni.
    - Ako je u pitanju ciljni čvor, pretraga je završena.
    - Ako nije u pitanju ciljni čvor, dodati njegove sledbenike (ukoliko postoje) u odgovarajućem redosledu na **kraj** liste.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 2 – Rešenje

Pretraga po širini (*breadth first search*):



putanja: SBG
cena putanje: 15

Zadatak 2 – Rešenje

Pretraga po širini (*breadth first search*):

Da li je pronađena putanja optimalna?

Pronađena putanja je optimalna u broju koraka od startnog do ciljnog čvora (ukoliko bi svaka grana imala jednaku cenu). Ukoliko grane nemaju jednaku cenu, BFS ne garantuje pronalaženje optimalnog rešenja.

Koja struktura podataka odgovara BFS algoritmu?

FIFO (*First in first out*) struktura podataka, odnosno red.

Zadatak 2 – Rešenje

Prvo najbolji (*best first search*):

- Lista čvorova sadrži startni čvor.
- Dokle god je lista čvorova neprazna:
  - Uklanja se čvor sa početka liste i ispituje se da li je ciljni.
    - Ako je u pitanju ciljni čvor, pretraga je završena.
    - Ako nije u pitanju ciljni čvor, dodati njegove sledbenike (ukoliko postoje) u listu. Listu sortirati prema rastućoj vrednosti **heuristike** čvora.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 2 – Rešenje

Heuristika (funkcija procene) se može odrediti za svako stanje u prostoru problema i predstavlja estimaciju cene rešenja od tekućeg do ciljnog stanja, uzimajući trenutno stanje kao startno. Heuristika se može posmatrati kao dodatno znanje o problemu koje treba da poboljša proces pretraživanja.

Dobra heuristika **potcenjuje** cenu rešenja podproblema.

Šta se dešava ukoliko heuristika precenjuje rešenje?

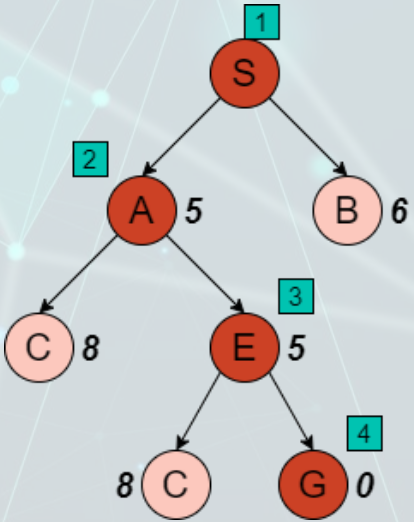
U tom slučaju nije garantovano optimalno rešenje problema.

Kako uporediti kvalitet dve heuristike koje vode do istog rešenja?

Jedno merilo bi bilo uporediti broj ekspandovanih čvorova (prosečan faktor grananja čvorova).

Zadatak 2 – Rešenje

Prvo najbolji (*best first search*):



putanja: SAEG
cena putanje: 13

Zadatak 2 – Rešenje

Prvo najbolji (*best first search*):

Da li je pronađena putanja optimalna?

Best first search ne garantuje pronalaženje optimalnog rešenja, jer rad algoritma zavisi isključivo od heurističke funkcije, koja može davati manju vrednost od stvarne cene parcijalne putanje.

Čak i da heuristička funkcija za svaki čvor potcenjuje preostalu cenu do rešenja, ovaj algoritam ne garantuje pronalaženje optimalnog rešenja, jer ne razmatra cene putanja između čvorova. Heuristička funkcija nekog čvora bi „jače“ mogla da potcenjuje cenu preostalog rešenja od nekog drugog čvora u kome heuristička funkcija takođe potcenjuje cenu preostalog rešenja pa bi „lošiji“ čvor bio izabran kao deo rešenja.

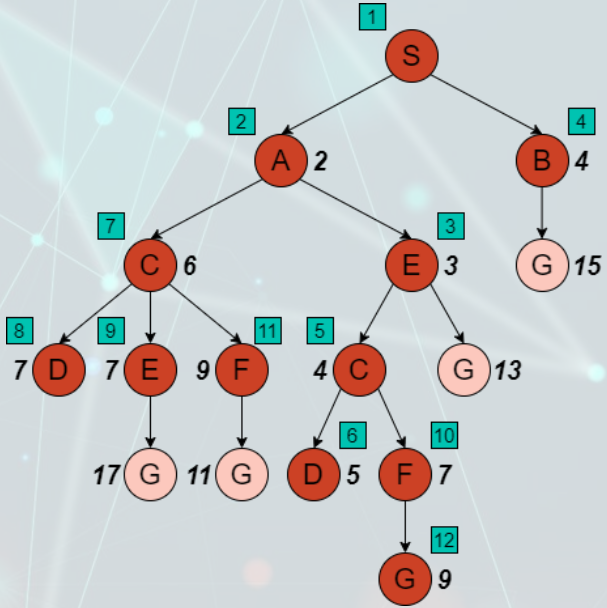
Zadatak 2 – Rešenje

Grananje i ograničavanje (*branch and bound search*):

- Lista parcijalnih putanja sadrži jednu putanju dužine 0 (startni čvor).
- Dokle god je lista parcijalnih putanja neprazna:
  - Uklanja se putanja sa početka liste.
    - Ako putanja dostiže ciljni čvor, pretraga je završena.
    - Za svakog sledbenika poslednjeg čvora na uklonjenoj putanji formira se po jedna nova parcijalna putanja.
    - Za svaku putanju izračunava se njena cena.
    - Nove putanje se dodaju u listu koja se održava rastuće prema ceni putanje.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 2 – Rešenje

Grananje i ograničavanje (*branch and bound search*):



putanja: SAECFG
cena putanje: 9

Zadatak 2 – Rešenje

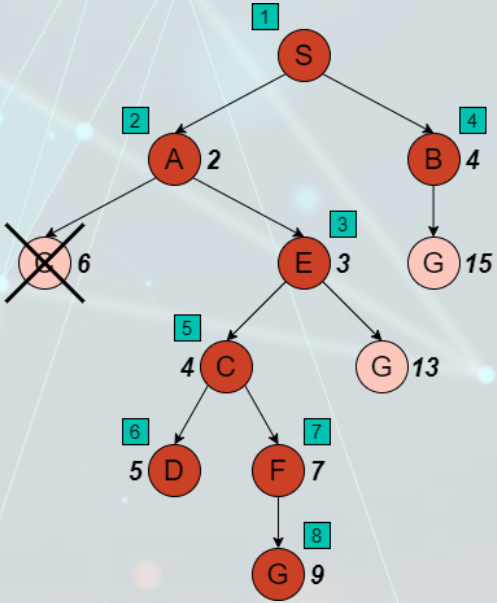
Grananje i ograničavanje (*branch and bound search*):

Da li je pronađena putanja optimalna?

Grananje i ograničavanje garantuje pronalaženje optimalne putanje. U momentu ekspanzije nekog čvora X, putanja od startnog čvora S do čvora X je već optimalna. Da to nije slučaj onda bi značilo da je čvor X već nekad ranije ekspandovan (označićemo ga sa X') i da je putanja S-X' sa manjom cenom, pa čvor X ne bi ni bio ekspandovan u ovom momentu. Takođe, dodavanjem novih čvorova na putanju one ne postaju kraće, jer su cene akcija/prelaza nenegativne.

Zadatak 2 – Rešenje

Grananje i ograničavanje (*branch and bound search*):



putanja: SAECFG
cena putanje: 9
dinamičko programiranje: DA

Zadatak 2 – Rešenje

Grananje i ograničavanje (*branch and bound search*):

Da li se test cilja primenjuje na generisani ili ekspandovani čvor?

Primenjuje se na ekspandovani čvor, jer bi se moglo desiti da se generisani ciljni čvor nalazi na suboptimalnoj putanji.

U slučaju BFS pretraživanja, da li se test cilja može primeniti na generisani čvor?

Može, jer BFS ne koristi cene putanja.

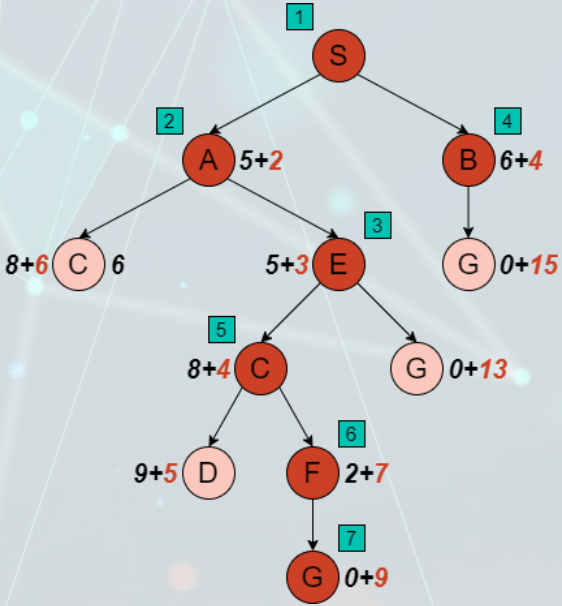
Zadatak 2 – Rešenje

A* (*A star search*):

- Lista parcijalnih putanja sadrži jednu putanju dužine 0 (startni čvor).
- Dokle god je lista parcijalnih putanja neprazna:
  - Uklanja se putanja sa početka liste.
    - Ako putanja dostiže ciljni čvor, pretraga je završena.
    - Za svakog sledbenika poslednjeg čvora na uklonjenoj putanji formira se po jedna nova parcijalna putanja.
    - Za sve putanje izračuna se **kumulativna** cena, koja predstavlja sumu cene parcijalne putanje od startnog do tekućeg čvora i funkcije procene putanje od tekućeg do ciljnog.
    - Nove putanje se dodaju u listu koja se održava rastuće prema ceni putanje.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 2 – Rešenje

A* (*A star search*):



putanja: SAECFG
cena putanje: 9

Zadatak 2 – Rešenje

A* (*A star search*):

Da li je pronađena putanja optimalna?

A* garantuje pronalaženje optimalne putanje, ukoliko heuristika potcenjuje cenu rešenja. Heuristička funkcija izračunava se kao

$$f(n) = g(n) + h(n)$$

gde g(n) predstavlja cenu putanje od startnog čvora do čvora n, a h(n) estimaciju cene putanje od čvora n do ciljnog čvora.

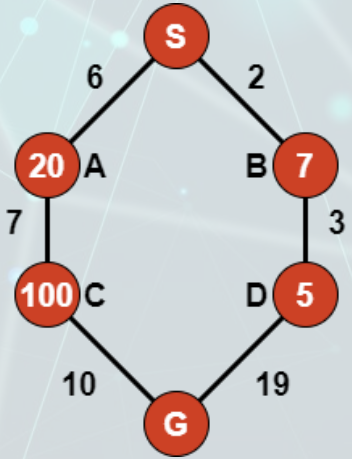
Šta bi bilo da je u prethodnom primeru put E-G bio 8 umesto 10? Da li bi tada A našao optimalno rešenje?*

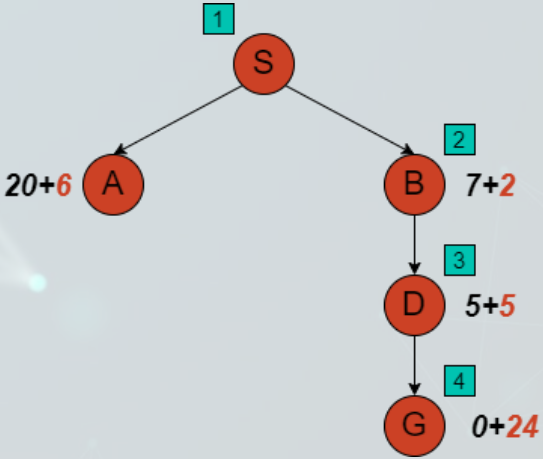
Ne, jer bi u koraku 6 čvor taj čvor G bio izabran za ekspanziju.

Zadatak 2 – Rešenje

Šta ako heuristika precenjuje?

A* ne garantuje pronalaženje optimalne putanje ...

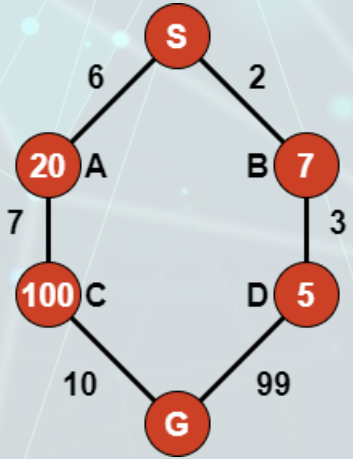


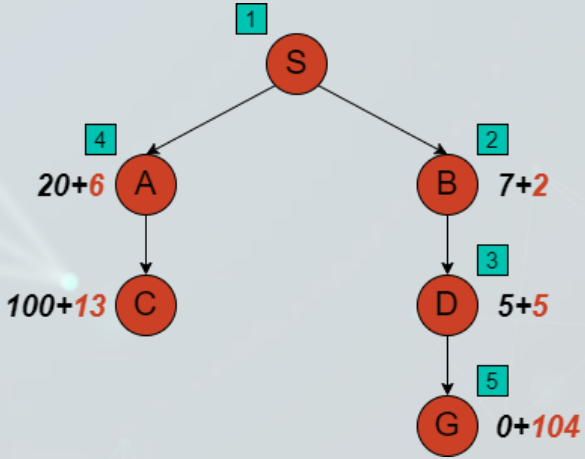


Zadatak 2 – Rešenje

Šta ako heuristika precenjuje?

A* ne garantuje pronalaženje optimalne putanje ...

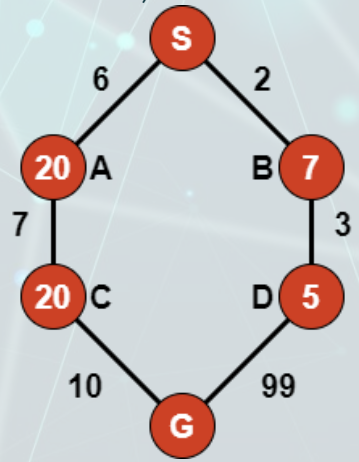


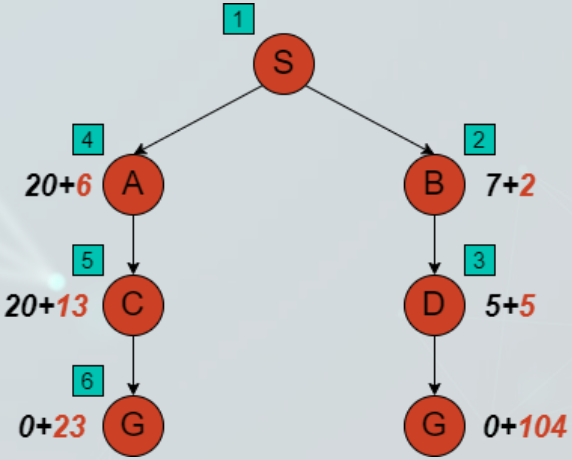


Zadatak 2 – Rešenje

Šta ako heuristika precenjuje?

... ali može da se desi i da optimalna putanja bude pronađena (na ovo ne treba računati).

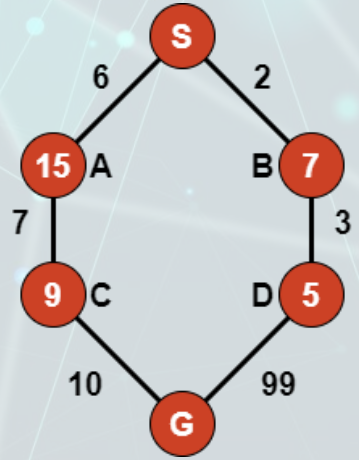


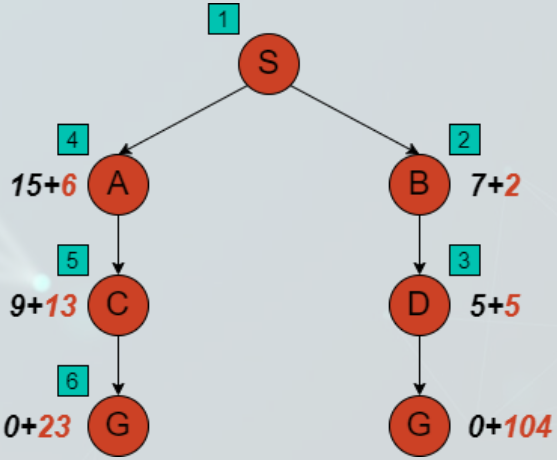


Zadatak 2 – Rešenje

Da li su sve heuristike koje potcenjuju iste?

Svaka heuristika koja potcenjuje rešenje daće garantovano optimalno rešenje, ali ne u najmanjem broju ekspanzija čvorova ...

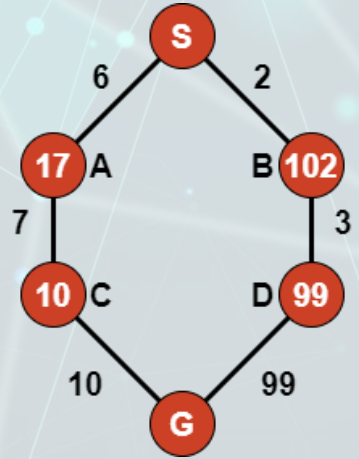


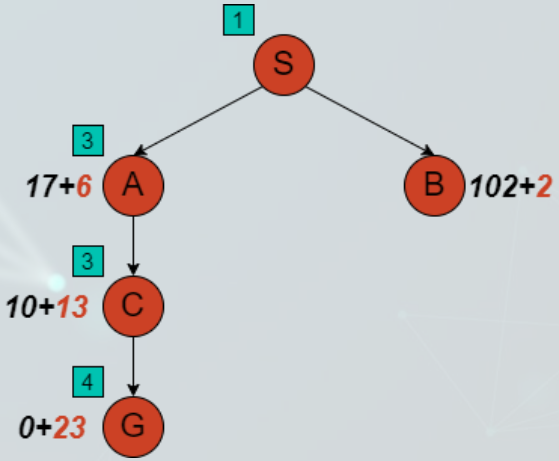


Zadatak 2 – Rešenje

Da li su sve heuristike koje potcenjuju iste?

... a heuristika koja je po vrednosti jednaka tačnom rešenju će dati optimalno rešenje u najmanjem broju ekspandovanih čvorova.

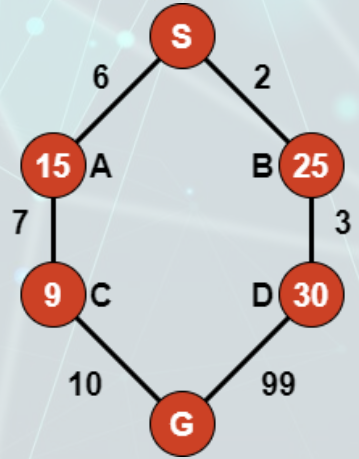


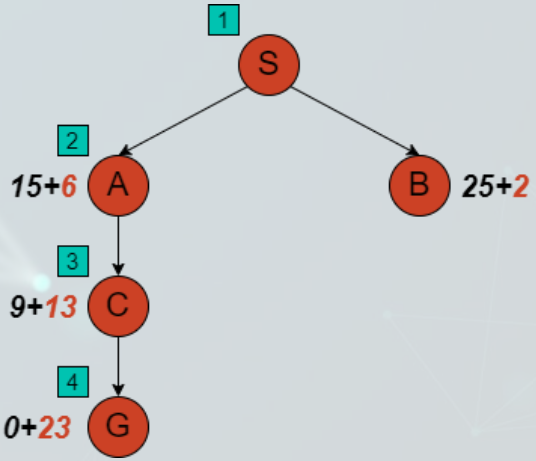


Zadatak 2 – Rešenje

Da li su sve heuristike koje potcenjuju iste?

Tačnu heuristiku je teško pronaći, ali treba težiti da ona bude što bliža stvarnoj preostaloj ceni da se ne bi nepotrebno ekspandovao veći broj čvorova.

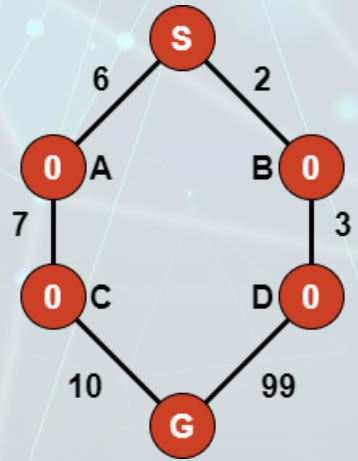


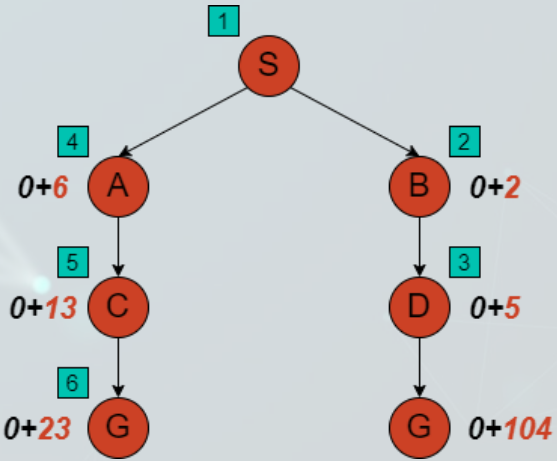


Zadatak 2 – Rešenje

Da li su sve heuristike koje potcenjuju iste?

U najgorem slučaju, kada je heuristika 0, A* algoritam se svodi na Branch & Bound.





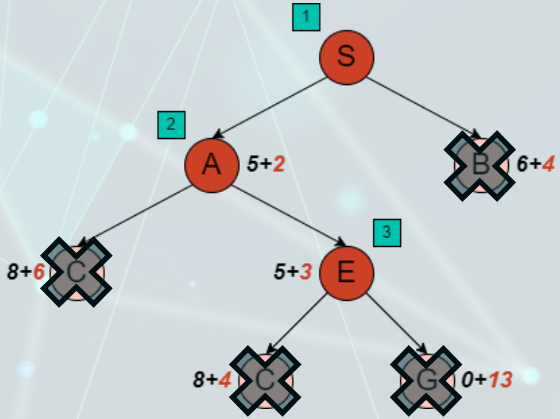
Zadatak 2 – Rešenje

ID A* (*Iterative Deepening A star search*):

- Lista čvorova sadrži startni čvor i njegovu **kumulativnu** cenu (za startni čvor se svodi na vrednost heurističke funkcije).
- Postavlja se prag prihvatanja na prethodno dobijenu vrednost.
- Dokle god je lista čvorova neprazna:
  - Uklanja se čvor sa početka liste i ispituje se da li je ciljni.
    - Ako je u pitanju ciljni čvor, pretraga je završena.
    - Ako nije u pitanju ciljni čvor, za njegove sledbenike (ukoliko postoje) izračunati njihove **kumulativne** cene i na **početak** liste dodati samo one sledbenike čija kumulativna cena ne prelazi postavljen prag prihvatanja, a ostale čvorove odbaciti.
- Ako je ciljni čvor pronađen, pretraga je uspešna.
- Ako ciljni čvor nije pronađen, prag prihvatanja se povećava na minimalnu vrednost kumulativne cene svih čvorova koji su bili odbačeni, a potom se ceo postupak ponavlja.

Zadatak 2 – Rešenje

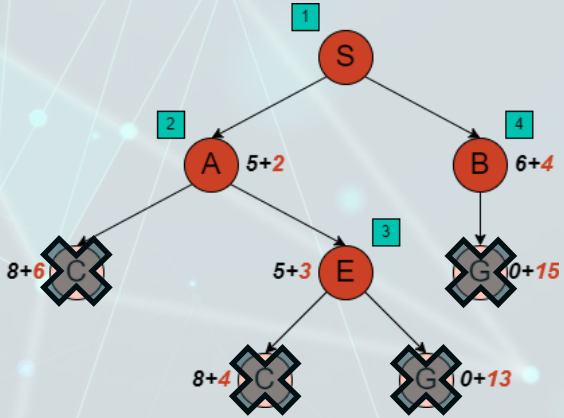
ID A* (*Iterative Deepening A star search*):



prag: 8

Zadatak 2 – Rešenje

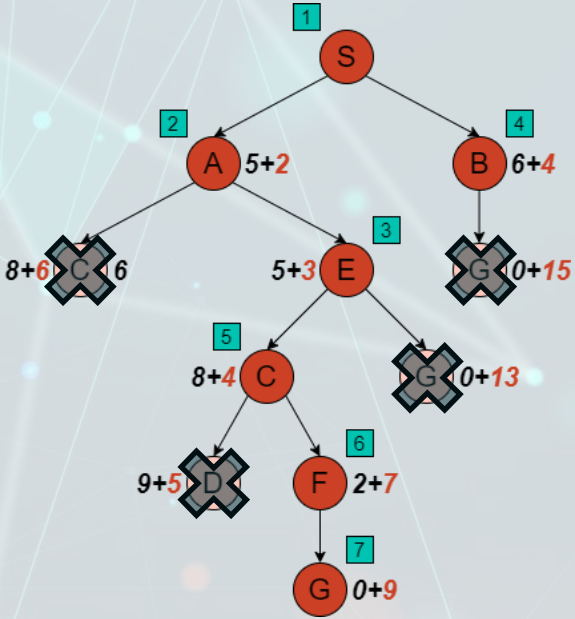
ID A* (*Iterative Deepening A star search*):



prag: 10

Zadatak 2 – Rešenje

ID A* (*Iterative Deepening A star search*):



prag: 12
putanja: SAECFG
cena putanje: 9

Zadatak 2 – Rešenje

ID A* (*Iterative Deepening A star search*):

Da li je pronađena putanja optimalna?

ID A* garantuje pronalaženje optimalne putanje, ukoliko heuristika potcenjuje cenu rešenja.

Koja je razlika između A i ID A*?*

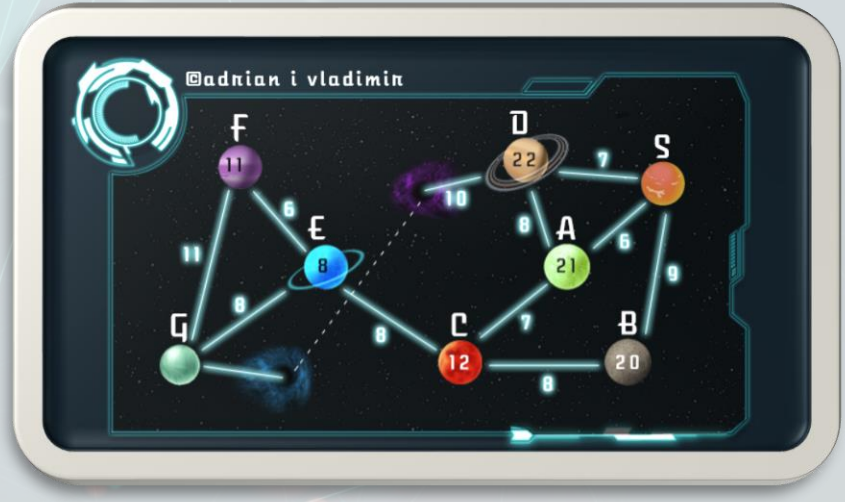
Algoritam ID A* zahteva manje memorije jer čuva samo jednu putanju od korena do čvora koji se trenutno razmatra, dok algoritam A* zahteva više memorije jer u memoriji čuva sve parcijalne putanje koje trenutno razmatra.

Algoritam ID A* često obilazi iste čvorove više puta, dok algoritam A* razmatra sve otvorene parcijalne putanje te ga dobra heuristika može voditi do rešenja u značajno manjem vremenu.

Zadatak za samostalnu vežbu - Svemir

★★☆☆☆

Na slici je prikazan deo putne mreže sa označenim dužinama puteva. Procenjena udaljenost od pojedinih planeta do ciljne planete G date su na planeti.

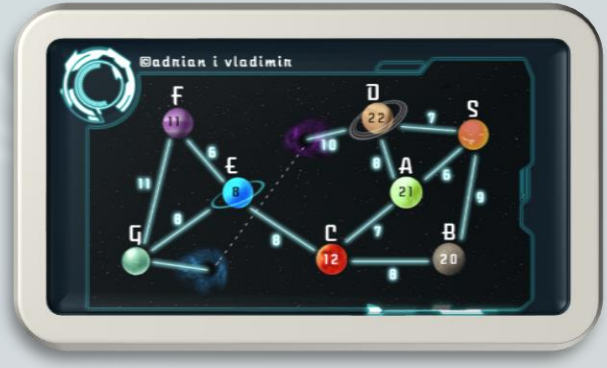


Zadatak za samostalnu vežbu - Svemir

★★☆☆☆

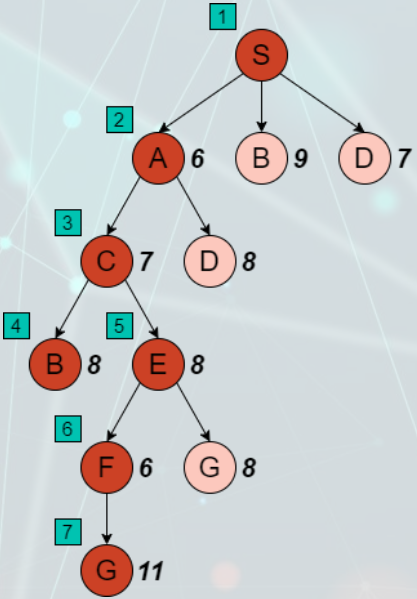
Prikazati stablo pretrage i navesti redosled obilaženja čvorova prilikom pronalaženja putanje između planeta S i G, ukoliko se koriste sledeći algoritmi:

- Pretraga po dubini (sledeća planeta za obilazak se bira kao najbliža planeta tekućoj)
- Pretraga po širini
- Prvo najbolji
- Grananje i ograničavanje
- A*

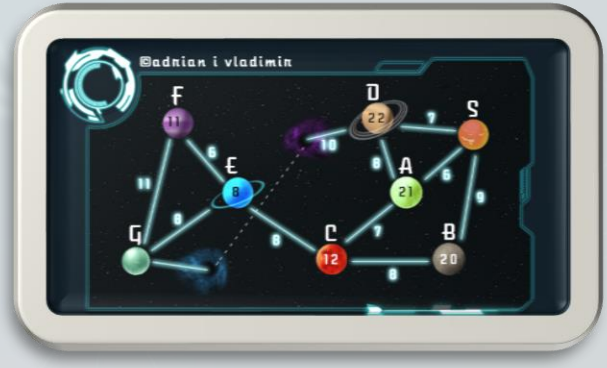


Zadatak za samostalnu vežbu – Rešenje

Pretraga po dubini sa datom heuristikom (*greedy local best-first*):

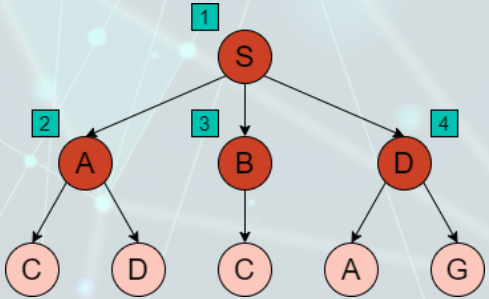


putanja: SACEFG
cena putanje: 38

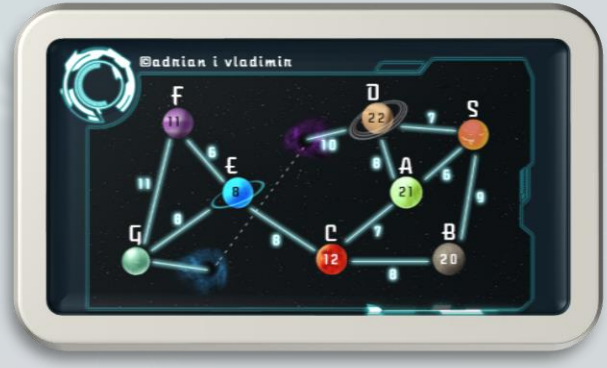


Zadatak za samostalnu vežbu – Rešenje

Pretraga po širini:

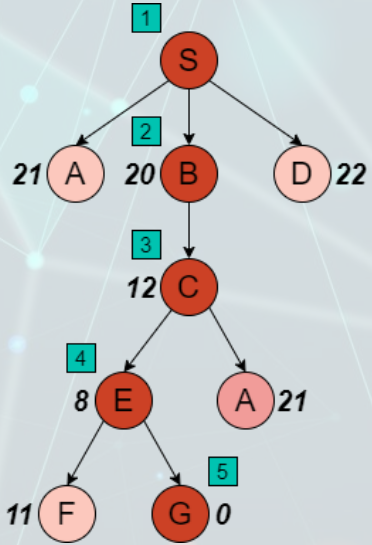


putanja: SDG
cena putanje: 17

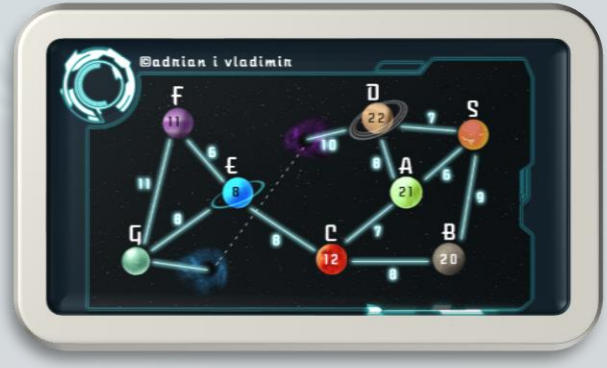


Zadatak za samostalnu vežbu – Rešenje

Prvo najbolji:

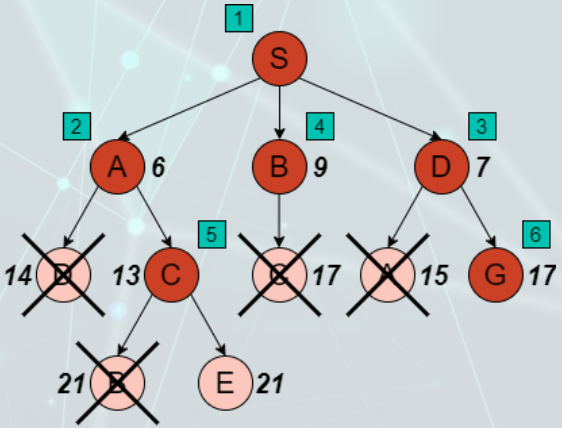


putanja: SBCEG
cena putanje: 33

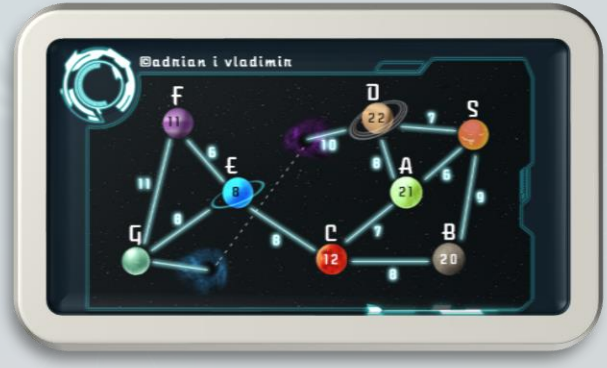


Zadatak za samostalnu vežbu – Rešenje

Grananje i ograničavanje:

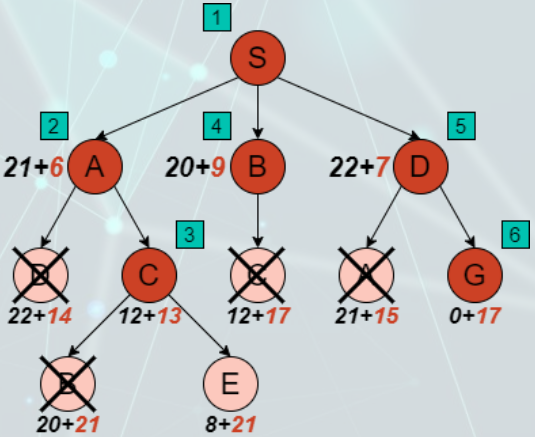


putanja: SDG
cena putanje: 17

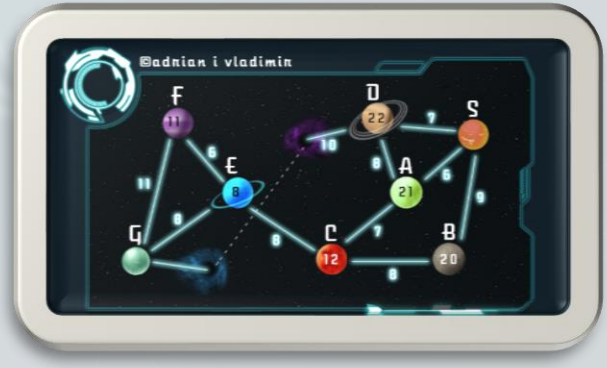


Zadatak za samostalnu vežbu – Rešenje

A*:



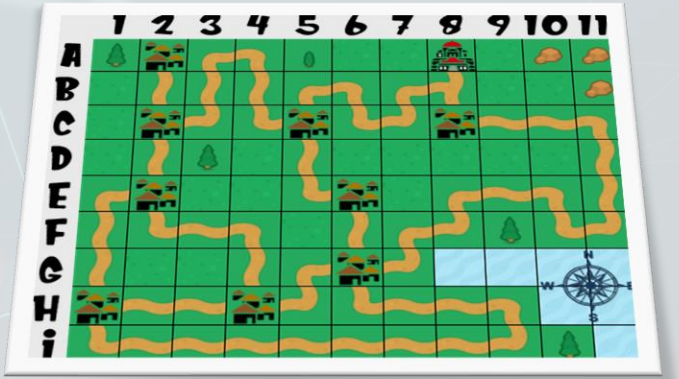
putanja: SDG
cena putanje: 17



Zadatak 3 – Rođendan kralja Artura

★★★☆☆

Mapa Arturovog kraljevstva se može videti na slici. Kralj Artur obeležava godišnjicu braka te je pozvao svoje vitezove na veliku proslavu u Kamelot. Svi vitezovi su pre toga bili kod Tristana na rođendanu i kreću iz sela koje se na mapi nalazi na poziciji (H, 1). Kamelot se na mapi nalazi na poziciji (A, 8). Putanja iz jednog sela ka drugom postoji, ukoliko su sela povezana obeleženom stazom. Cena putanje odgovara broju polja koje obuhvata staza između ta dva sela ne računajući polja na kojima se nalaze sela. Za svaki od sledećih obilazaka navesti algoritam pretraživanja koji se koristi, prikazati stablo pretrage, navesti redosled obilaska čvorova, dati putanju kojom će se vitezovi kretati i odrediti cenu dobijene putanje. Prilikom rešavanja stavke 1 koristiti dinamičko programiranje.

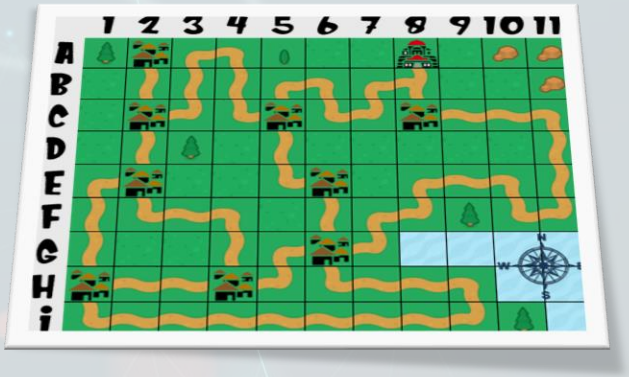


Zadatak 3 – Rođendan kralja Artura

★★★☆☆

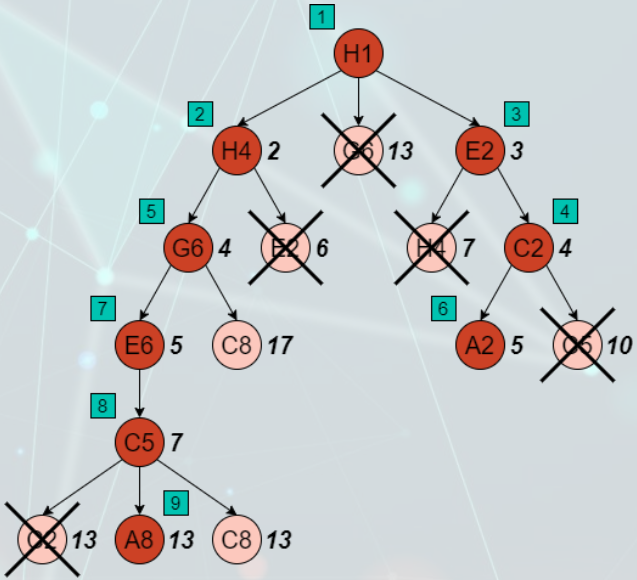
Vitez Lanselot je obećao Arturu i njegovoj supruzi Gvinervi da će im pomoći u dekoraciji svečane dvorane u kojoj će se održati proslava. Da bi se sve dekorisalo na vreme, Lanselotu je u interesu da stigne u Kamelot što pre. Lanselot kod sebe poseduje mapu Arturovog kraljevstva i spreman je da odvoji malo vremena da pronađe *najkraći mogući put*.

Koji algoritam pretraživanja koristi Lanselot?

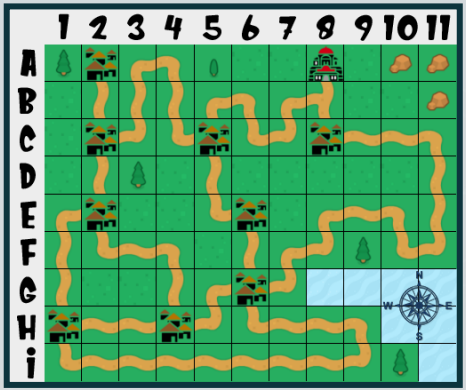


Zadatak 3 – Rešenje

Grananje i ograničavanje (*branch and bound search*):



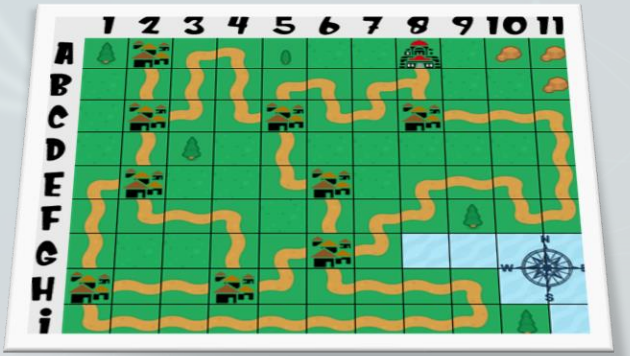
putanja:
H1-H4-G6-E6-C5-A8
cena putanje: 13



Zadatak 3 – Rođendan kralja Artura

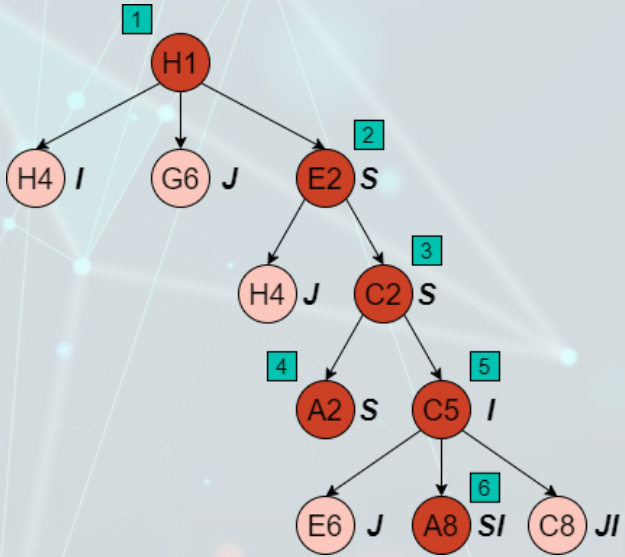
Vitezovi okruglog stola kreću sa Tristanovog rođendana ka Kamelotu odmah nakon Lanselota. Usled velike količine popijenog vina, nijedan vitez se nije setio da nabavi mapu. Jedino čega se sećaju jeste da se Kamelot u odnosu na Tristanovu kuću nalazi negde na severoistoku. Stoga, vitezovi odlučuju da *na raskrsnicama puteva najveću prednost daju putu koji ide ka severu, zatim putu koji ide ka istoku, zatim putu koji ide ka jugu i najmanju prednost daju putu koji ide ka zapadu*.

Koji algoritam pretraživanja koriste vitezovi okruglog stola?

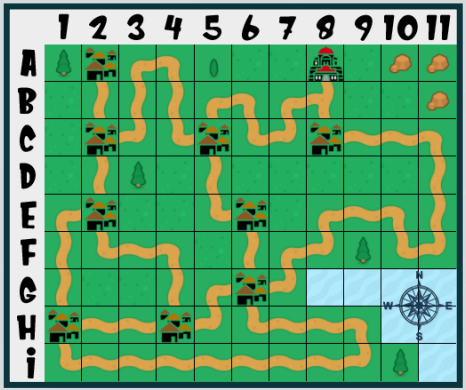


Zadatak 3 – Rešenje

Pretraga po dubini (*depth first search*):



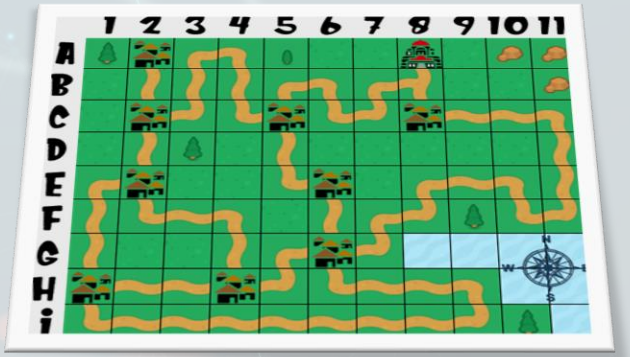
putanja:
H1-E2-**C2-A2-C2**-C5-A8
cena putanje: 18



Zadatak 3 – Rođendan kralja Artura

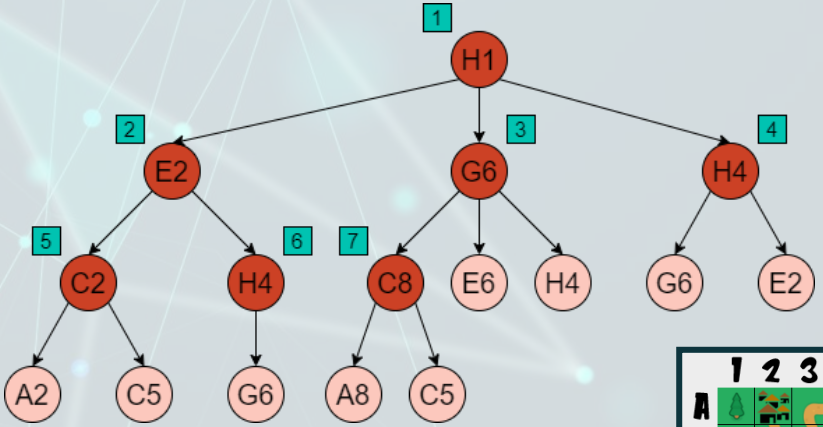
Vitez Tristan i njegova supruga Izolda kreću zajedno iz njihove kuće. Tristan ima običaj da se zadržava po selima, odmara i uživa u lokalnoj hrani i piću pre nego što nastavi put. Stoga, Izolda odlučuje da na osnovu mape osmisli poseban put do Kamelota za njih dvoje. Ona veruje da će brže stići u Kamelot krećući se *putem koji prolazi kroz najmanji mogući broj sela*.

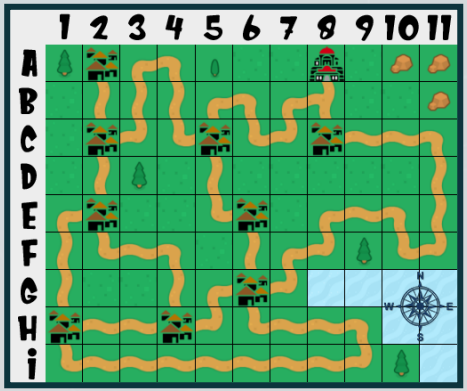
Koji algoritam pretraživanja koriste Tristan i Izolda?



Zadatak 3 – Rešenje

Pretraga po širini (*breadth first search*):





putanja:
H1-G6-C8-A8
cena putanje: 27

Zadatak za samostalnu vežbu - Planinari

★★★☆☆

Neda, Slavica i Dunjica planinare. Skica brda prikazana je na slici, pri čemu je slika orijentisana tako da je sever gore. Brojevi u svakom polju označavaju visinu datog polja. Po brdu je moguće kretati se isključivo jedno polje vertikalno ili horizontalno. Pri prelasku sa polja na polje potrebno je uložiti određeni napor koji odgovara visinskoj razlici između ta dva polja. Planinarke kreću polazeći od najniže tačke na brdu i cilj im je da se popnu na vrh brda (najviša tačka na brdu), pri čemu svaka planinarka koristi svoju strategiju pretrage (dozvoljeno je korišćenje dinamičkog programiranja i proveru ciljnog čvora na generisani čvor za svaki algoritam za koji to ima smisla). Koja planinarka će prva stići na vrh, ukoliko Neda pređe šest, Slavica četiri, a Dunjica tri jedinice visine za deset minuta?

	1	2	3
A	12	3	10
B	5	2	15
C	0	3	12

Zadatak za samostalnu vežbu - Planinari

★★★☆☆

Neda ne voli da osmišljava planove te prednost jednostavno daje poljima na severu, zatim na istoku, potom na jugu i na kraju na zapadu.

Slavica ne voli da se umara te prednost daje poljima sa najmanjom visinskom razlikom u odnosu na polje na kojem se u tom trenutku nalazi.

Dunjica je ponela mapu brda i unapred izračunala najlakši put do vrha koristeći isključivo podatke date na mapi. Dunjica želi i da iskoristi optimalnu, ali lako izračunljivu heuristiku da bi algoritam dao optimalno rešenje u manjem broju ekspandovanih čvorova nego algoritam koji ne koristi heuristiku.

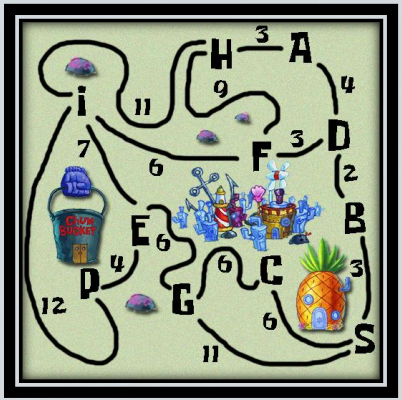
Za svaku planinarku navesti algoritam pretraživanja koji koristi, prikazati stablo pretrage, navesti redosled obilaska čvorova, dati putanju kojom će se planinarke kretati i odrediti cenu putanje.

Zadatak za samostalnu vežbu - Koralovo

★★★☆☆

Plankton je posle mnogo neuspelih pokušaja ukrao tajni recept Kebine pljeskavice sa kojim napokon može da uništi konkurenciju u prodaji brze hrane. Recept se nalazi u aktovci koju je Plankton ukrao sa dobro skrivenog mesta. Sve što Plankton treba da uradi jeste da stigne do kuće (mesto P) i polomi katanac na aktovci uz pomoć alata koji se tamo nalazi. Kada je Plankton ukrao recept, u Sunđer Bobovoj kući je zazvonio alarm. Sunđer Bob je hitro krenuo da traži lopova. Plankton i Sunđer Bob kreću iz različitih mesta u Koralovu i kreću se koristeći različite strategije. Mapa Koralova sa najvažnijim mestima i cenama puteva između mesta data je na slici.

Da li će Sunđer Bob uhvatiti Planktona?
Smatrati da Sunđer Bob hvata Planktona, ukoliko se u istom trenutku nalaze u istom mestu. Sunđer Bob ne može da uhvati Planktona na putu između dva mesta. Cena puteva sa slike odgovara broju minuta za koji se put može preći.



Zadatak za samostalnu vežbu - Koralovo

★★★☆☆

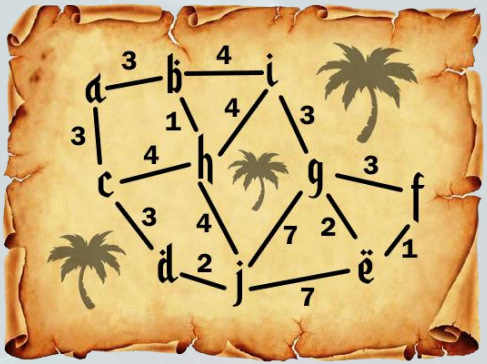
Plankton kreće iz mesta A i kreće se najbržim mogućim putem koji unapred sastavlja na osnovu plana i mape grada koju poseduje. Plankton ne želi da izgubi previše vremena pri kreiranju puta pa razmišlja da iskoristi tehniku dinamičkog programiranja i proveru ciljnog čvora na generisani čvor umesto na ekspandovani čvor, ukoliko navedene tehnike smeju da se koriste u odabranom algoritmu pretraživanja.

Sunđer Bob kreće iz mesta S u isto vreme kad i Plankton kreće iz mesta A. Sunđer Bob ne zna gde se nalazi lopov te odlučuje da luta gradom ne bi li naišao na nekoga ko nosi aktovku sa tajnim receptom. Na raskrsnicama puteva Sunđer Bob bira sledeće mesto za obilazak koje ima leksikografski najmanju oznaku pri čemu se nikad ne vraća u mesto koje je već obišao.

Zadatak za samostalnu vežbu - Agraba

★★★☆☆

Džafar je čuo da se na mapi Agrabe u mestu označenom sa F nalazi čarobna lampa u kojoj se nalazi duh koji može da ispunjava želje. Aladin, čuvši da Džafar traži lampu, brzo uzima stvar u svoje ruke i takođe kreće u potragu jer zna koje bi posledice mogle da budu, ukoliko se Džafar domogne lampe. Mapa Agrabe sa najvažnijim mestima i cenama puteva između mesta data je na slici. Aladin i Džafar kreću iz različitih lokacija i koriste različite strategije da bi pronašli lampu. Smatrati da cena puteva odgovara broju sati za koji se put može preći. U slučaju da u algoritmu nije moguće napraviti izbor sledećeg čvora za ekspandovanje, uzeti alfabetski poredak kao heuristiku (A B C D E F G H I J).



Zadatak za samostalnu vežbu - Agraba

★★★☆☆

Aladin kreće iz mesta B i brzo nabavlja mapu Agrabe. Nažalost, Džafar širom Agrabe ima špijune koji Aladinu mogu da stvaraju probleme na putu. Stoga, Aladin sastavlja put kojim bi prošao kroz najmanji broj mesta kako bi smanjio šansu za zasedu. Aladin ne želi da izgubi previše vremena pri kreiranju puta pa razmišlja da iskoristi tehniku dinamičkog programiranja i da proveru ciljnog čvora vrši na generisani čvor umesto na ekspandovani.

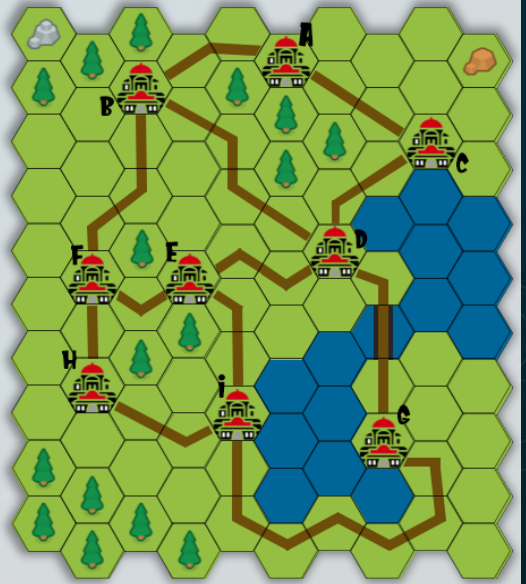
Džafar kreće iz mesta C i takođe nabavlja mapu Agrabe kako bi lakše pronašao put do lampe. Džafar je svestan da mu nije preostalo mnogo vremena pre nego što i ostali shvate njegov plan i spreče ga te na osnovu mape kreira najbrži put do lampe. Džafar ne želi da izgubi previše vremena pri kreiranju puta pa razmišlja da iskoristi princip dinamičkog programiranja i proveru ciljnog čvora na generisani čvor umesto na ekspandovani.

Zadatak za samostalnu vežbu - GoT

★★★★☆

Mapa dela Vesterosa se može videti na slici. Tajvin Lanister se kreće sa vojskom ka Kraljevoj Luci da bi služio kraljevstvu na mestu Kraljeve desnice.

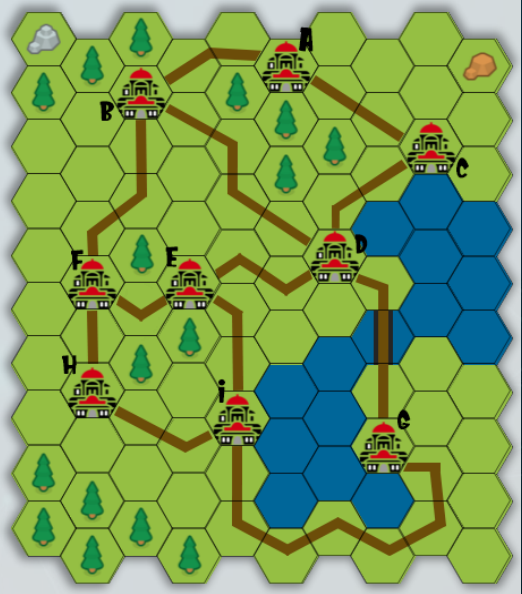
Lord Varis, deo Kraljevog malog veća, želi da sazna što više o Tajvinu i vojsci koju dovodi u prestonicu te šalje svoje špijune u bitne delove Vesterosa kako bi oni sakupili što više informacija. Tajvin i njegova vojska se nalaze u selu na poziciji čvora `A` u trenutku `t = 0` i kreću se putanjom `A – B – D – G – I`. Varis ne zna putanju kojom će se kretati Tajvin i njegova vojska te osmišljava strategiju kojom će se kretati njihovi špijuni.



Zadatak za samostalnu vežbu - GoT

★★★★☆

Putanja iz jednog mesta ka drugom postoji, ukoliko su mesta povezana obeleženom stazom. Cena putanje odgovara broju polja koje obuhvata staza između ta dva mesta ne računajući polja na kojima se nalaze mesta. Ukoliko se u bilo kom trenutku Tajvin i neki od špijuna nađu u istom mestu, tada Varis saznaje jednu novu informaciju od pijanih vojnika. Za sledeći obilazak navesti algoritam pretraživanja koji se koristi, prikazati stablo pretrage, navesti redosled obilaska čvorova, dati putanju kojom će se špijuni kretati i odrediti koliko informacija će saznati Varis.



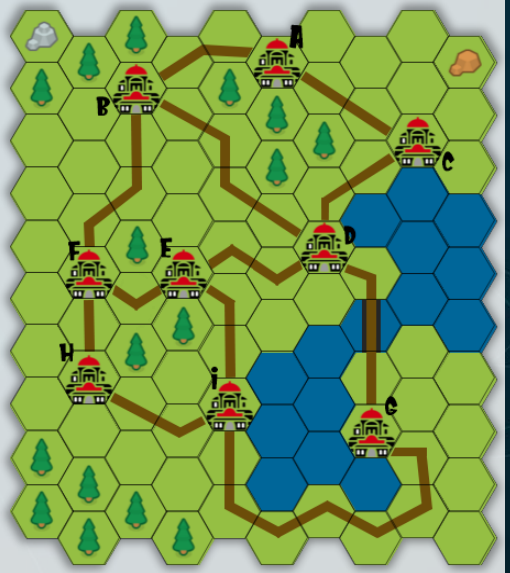
Zadatak za samostalnu vežbu - GoT

★★★★☆

Smatrati da se po putevima svi kreću brzinom od jednog polja po danu, ne računajući polja sa mestima (npr. ukoliko je neko u mestu A bio u trenutku t = 0, onda će u mestu B biti u trenutku t = 2).

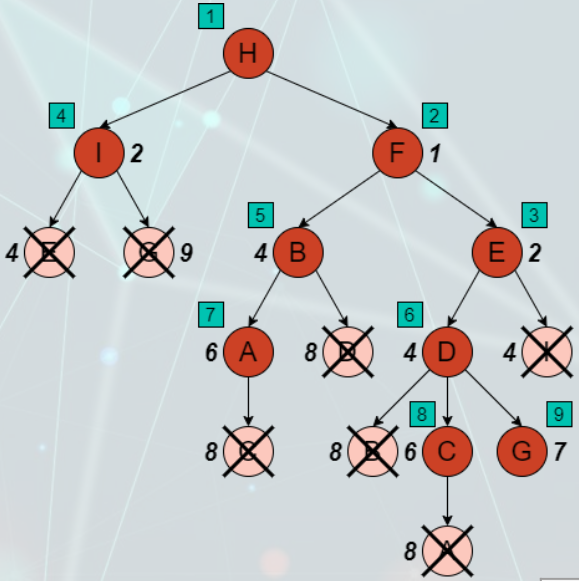
Varis šalje svoje špijune iz tačke H tako da u što kraćem vremenskom periodu zaposednu sva mesta na mapi deleći se u manje grupe po potrebi. Jednom kada zaposednu mesto, jedan od špijuna ostaje u tom mestu.

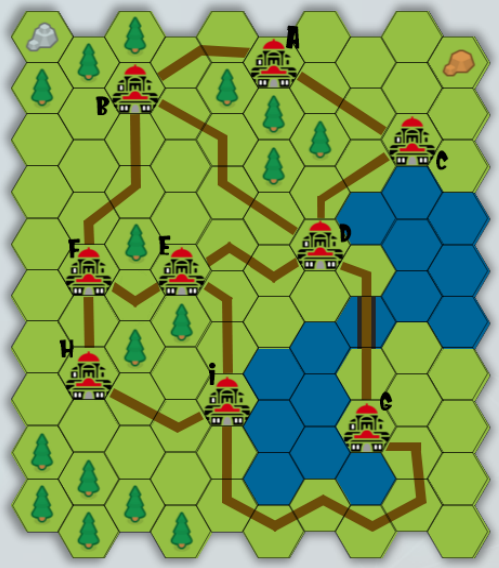
Smatrati da postoji dovoljan broj špijuna.



Zadatak za samostalnu vežbu - Rešenje

Grananje i ograničavanje:





broj sakupljenih informacija: 3

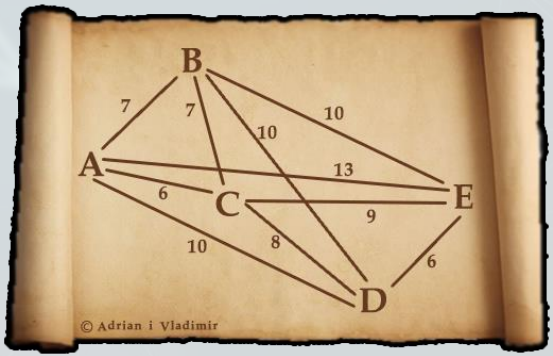
t	A	B	C	D	E	F	G	H	I
Tajvin	0	2	-	6	-	-	9	-	16
Varis	6	4	6	4	2	1	7	0	2

Zadatak 4 – Problem trgovačkog putnika

★★★★☆

Hermiona se bori za prava kućnih vilenjaka i želi da postavi postere koji podižu svest o pravima vilenjaka u svakoj od 5 učionica sa slike. Između svake dve učionice postoji put koji je moguće preći za vreme naznačeno na slici. Stoga, želi da izgubi što manje vremena u putovanju između učionica. Polazeći od učionice A treba pronaći minimalno vreme koje je potrebno da Hermiona poseti svaku učionicu tačno jednom i vrati se u učionicu A. Predložiti dve heurističke funkcije. Primeniti sledeće algoritme pretrage:

- *Greedy local best first search*
- *A**



Zadatak 4 – Rešenje

Problem trgovačkog putnika – NP-težak problem (*NP hard*)

- Jako teški za rešavanje, tj. najmanje su teški kao najteži problemi u NP klasi. Ne postoji poznat algoritam za njihovo optimalno rešavanje u svim slučajevima u polinomijalnom vremenu.
- Rešenje se brzo i lako može verifikovati u polinomijalnom vremenu u NP klasi problema, dok u NP-hard klasi to nije uvek slučaj.
- *Bruteforce* (permutacija broja gradova, *n!*), *Branch and bound*

Heuristička rešenja nisu uvek optimalna, ali mogu pružiti dobro rešenje u razumnom vremenu.

Greedy local best first search bira lokalno najbolji čvor. U ovom slučaju prioritet se daje najbližem neobiđenom čvoru.

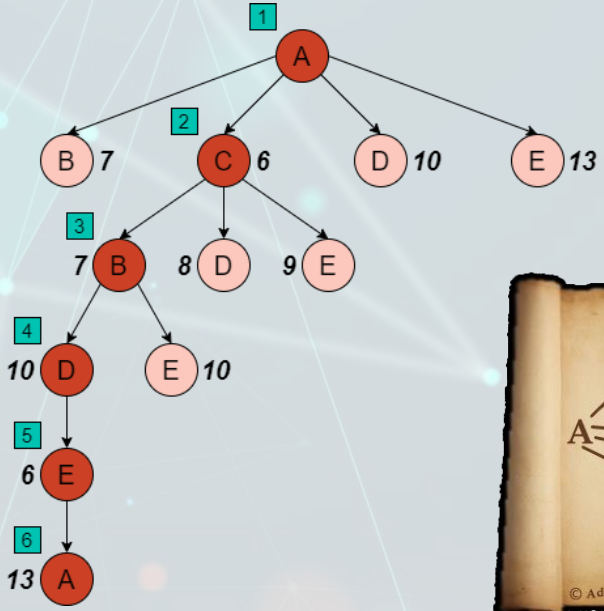
Zadatak 4 – Rešenje

Lokalno prvo najbolji (*greedy local best first search*):

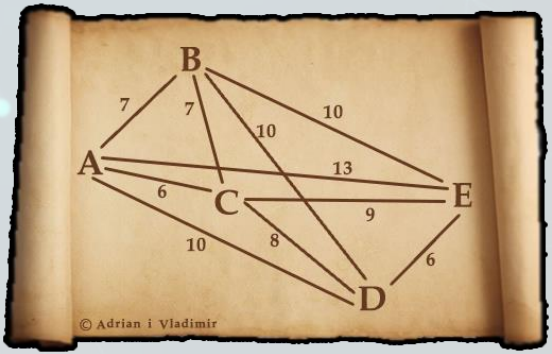
- Lista čvorova sadrži startni čvor.
- Dokle god je lista čvorova neprazna:
  - Uklanja se čvor sa početka liste i ispituje se da li je ciljni.
    - Ako je u pitanju ciljni čvor, pretraga je završena.
    - Ako nije u pitanju ciljni čvor, dodati njegove sledbenike (ukoliko postoje) u **sortiranom poretku prema rastućoj vrednosti heuristike čvora na početak liste**.
- Ako je ciljni čvor pronađen, pretraga je uspešna.

Zadatak 4 – Rešenje

Greedy local best first search:



putanja: ACBDEA
cena putanje: 42



Zadatak 4 – Rešenje

U slučaju algoritma A* kao ***heuristička funkcija*** preostale parcijalne putanje od ekspandovanog do ciljnog čvora mogla bi da se primeni cena ***minimalnog razapinjućeg stabla*** koje obuhvata sve čvorove grafa osim čvorova koji čine tu parcijalnu putanju (startni i ciljni čvor parcijalne putanje se ne uklanjaju).

Šta je minimalno obuhvatno stablo (Minimum Spanning Tree)?

To je podskup grana grafa koje povezuju sve čvorove grafa tako da nema petlji i čija je cena (zbir cene grana) najmanja.

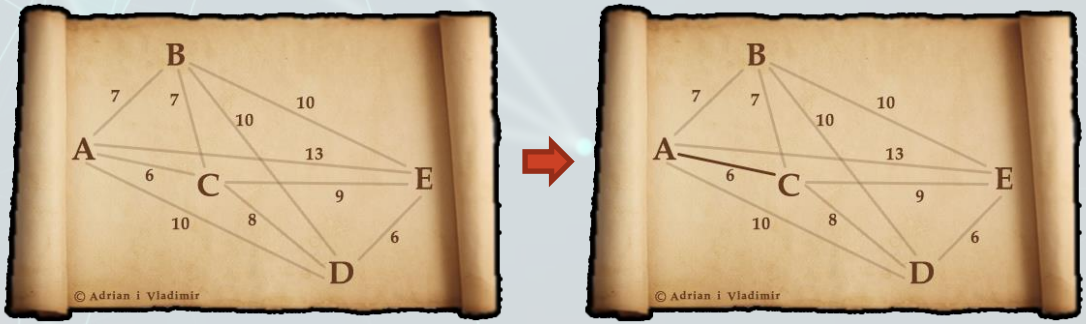
Kako se kreira MST?

Npr. koristeći Kruskalov algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.

Zadatak 4 – Rešenje

Kako se kreira MST?

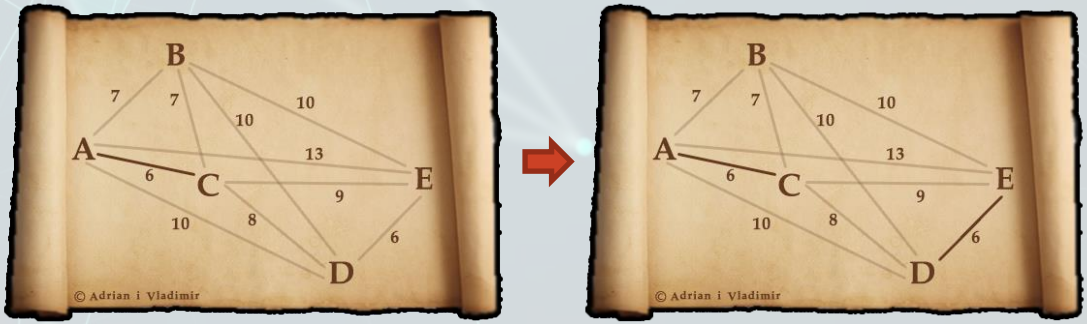
Npr. koristeći Kruskalov algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.



Zadatak 4 – Rešenje

Kako se kreira MST?

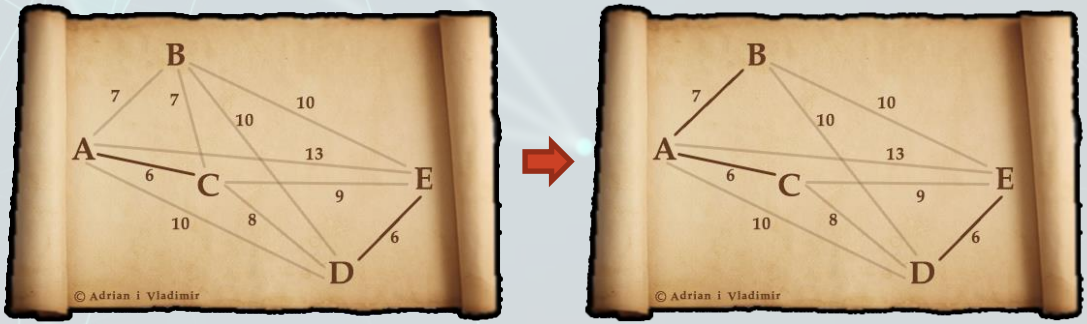
Npr. koristeći Kruskalov algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.



Zadatak 4 – Rešenje

Kako se kreira MST?

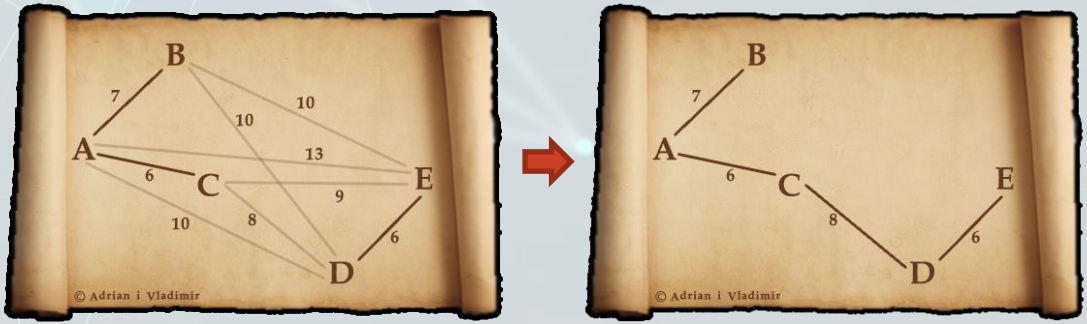
Npr. koristeći Kruskalov algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.



Zadatak 4 – Rešenje

Kako se kreira MST?

Npr. koristeći Kruskalov algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.



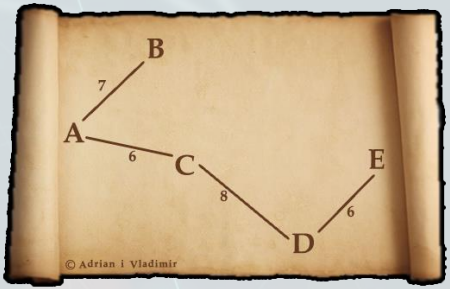
Zadatak 4 – Rešenje

Koja je motivacija za izbor ove heuristike?

MST je najmanja moguća procena preostalog dela puta koji treba preći, odnosno nije moguće povezati neposećene gradove i start i cilj posmatrane parcijalne putanje putem čija je cena manja od cene MST-a.

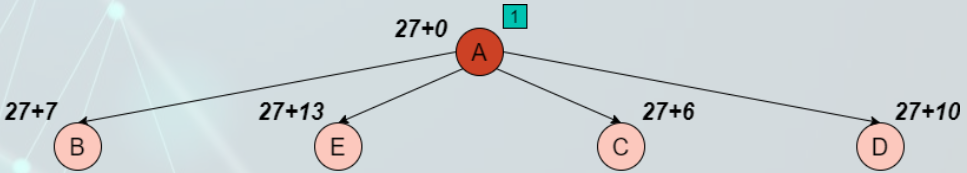
Zašto i startni i ciljni čvor parcijalne putanje čine MST?

Iako posećeni, oni povezuju parcijalnu putanju sa preostalim neposećenim čvorovima (krećemo od cilja parcijalne putanje, a u start moramo da se vratimo).

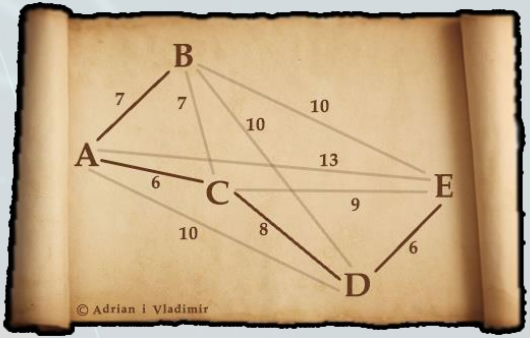


Zadatak 4 – Rešenje

A*:

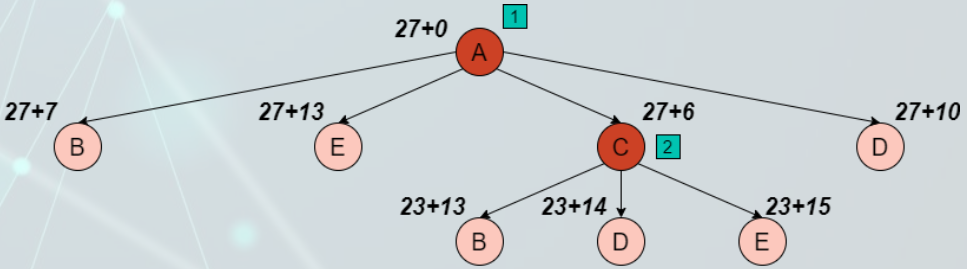


MST(BCDEA):
h = 27

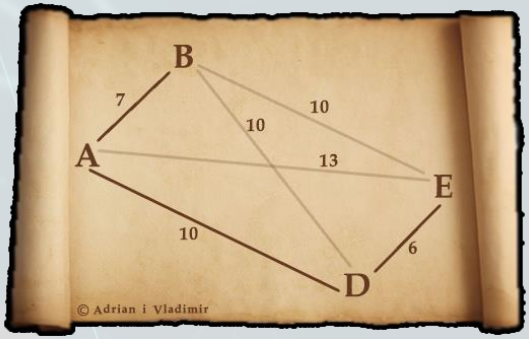


Zadatak 4 – Rešenje

A*:

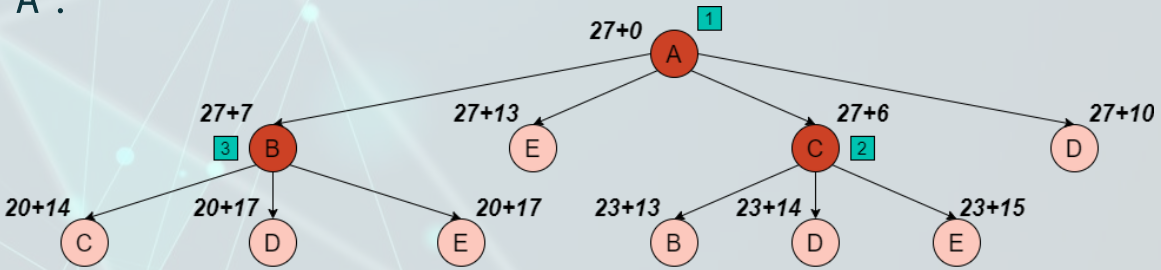


MST(BDEA):
h = 23

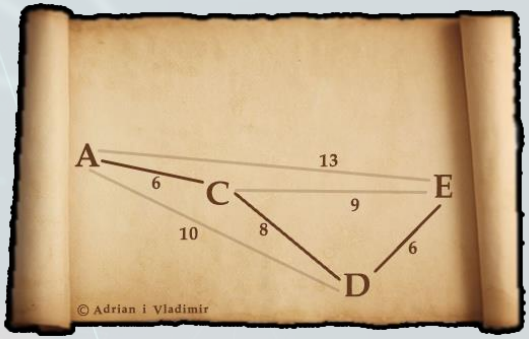


Zadatak 4 – Rešenje

A*:

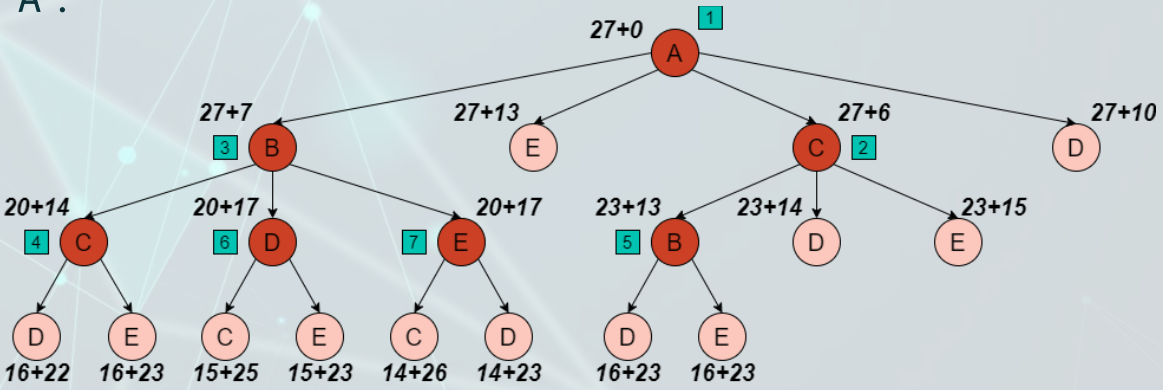


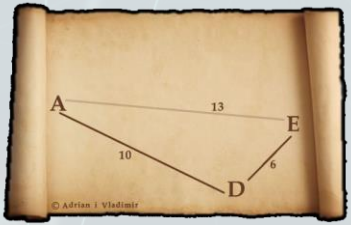
MST(CDEA):
h = 20

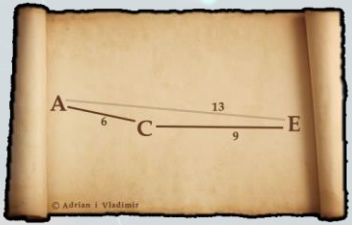


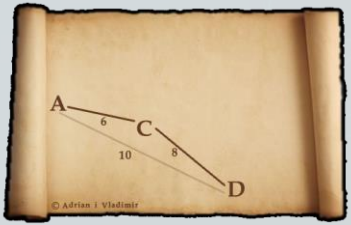
Zadatak 4 – Rešenje

A*:



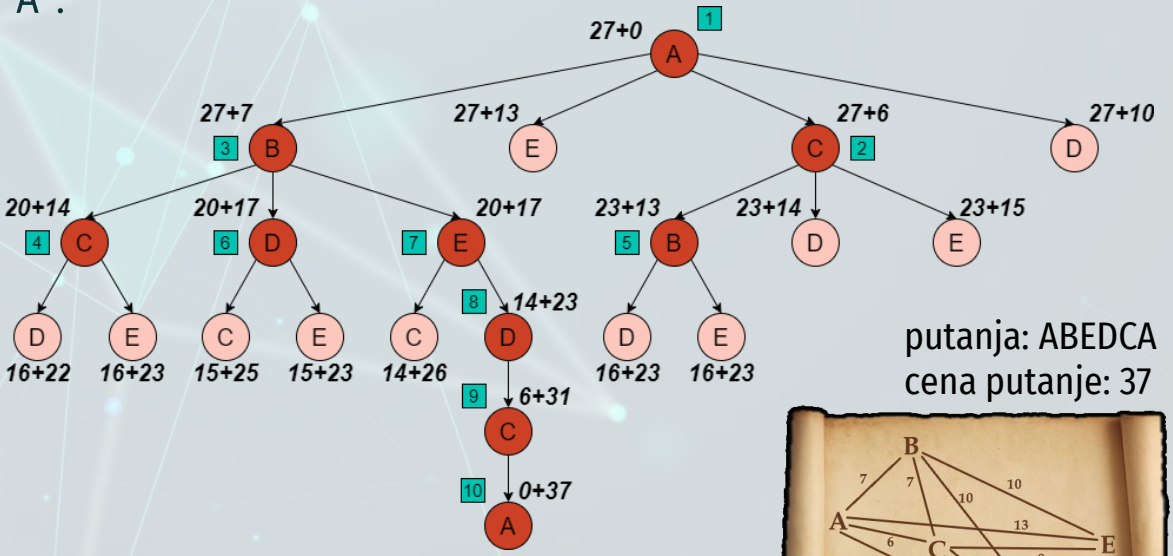






Zadatak 4 – Rešenje

A*:



putanja: ABEDCA
cena putanje: 37

PITANJA?

http://ri4es.etf.rs/