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).

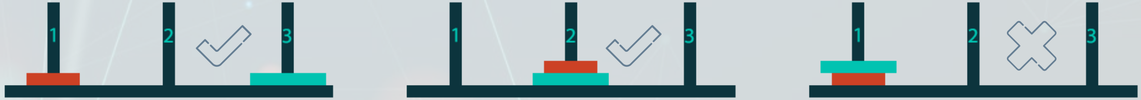
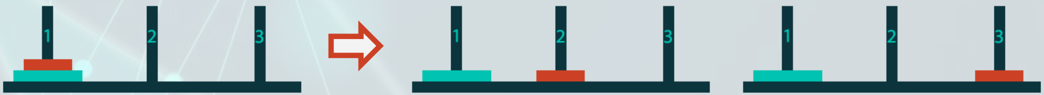


Tabela dozvoljenih prelaza između stanja

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.

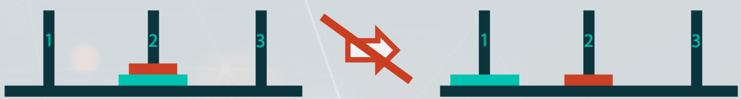


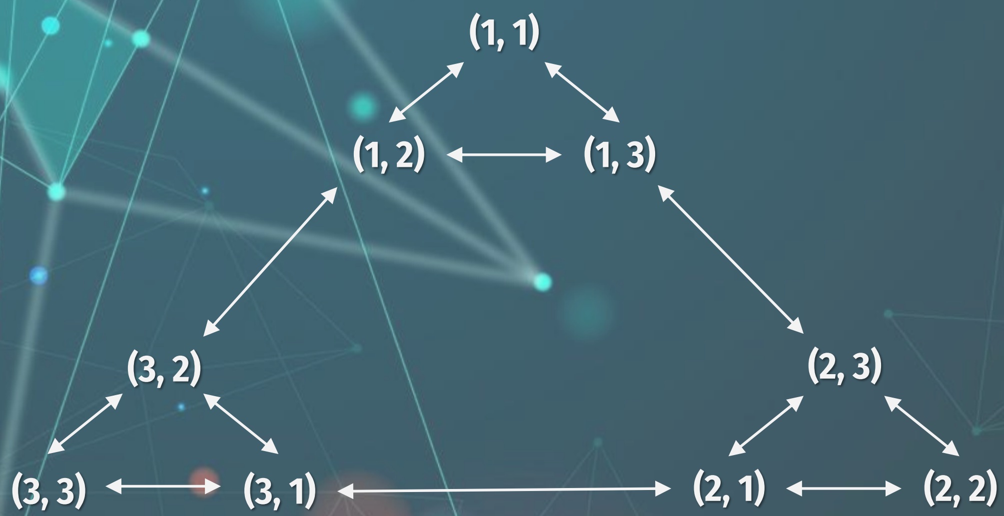
Tabela dozvoljenih prelaza između stanja

Prelaze je najlakše predstaviti i vizuelizovati 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 koje vode 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.

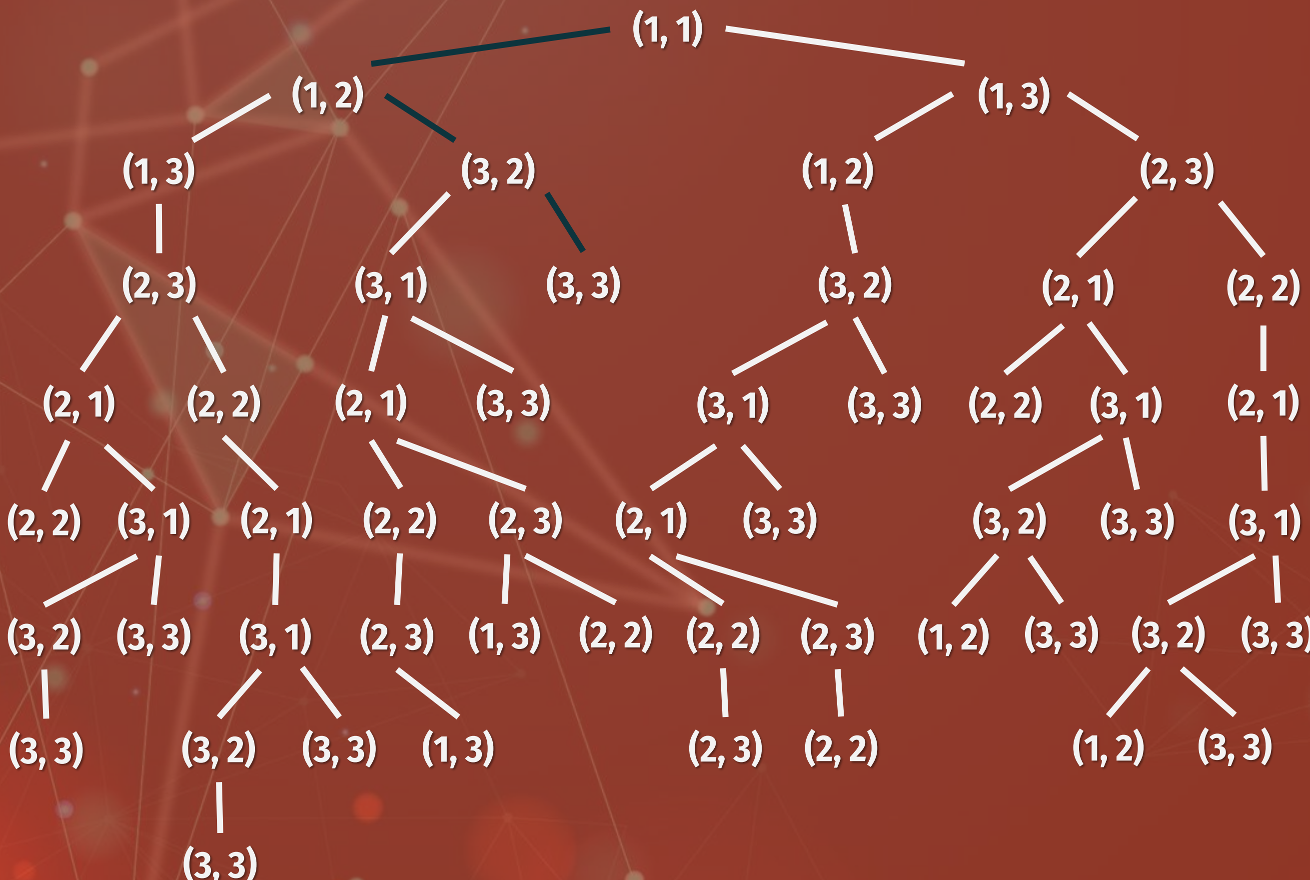
Proces generisanja novih čvorova (stanja) razmatranjem legalnih opcija (akcija) u tekućem čvoru (stanju) naziva se **ekspandovanje čvora**.

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).

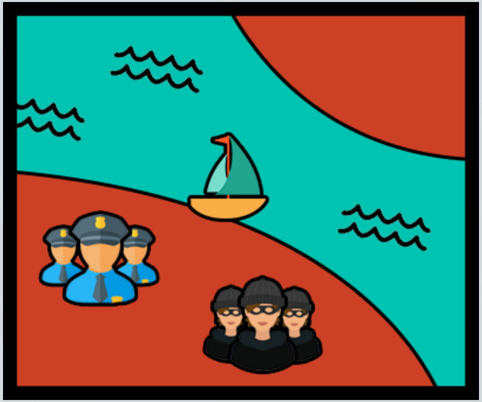

(1, 1)
(1, 2)
(1, 3)
(1, 3)
(3, 2)
(1, 2)
(2, 3)
(2, 3)
(3, 1)
(3, 3)
(3, 2)
(2, 1)
(2, 2)
(2, 1)
(2, 2)
(2, 1)
(3, 3)
(3, 1)
(3, 3)
(2, 2)
(3, 1)
(2, 1)
(2, 2)
(3, 1)
(2, 1)
(2, 2)
(2, 3)
(2, 1)
(3, 3)
(3, 2)
(3, 3)
(3, 1)
(3, 2)
(3, 3)
(3, 1)
(2, 3)
(1, 3)
(2, 2)
(2, 2)
(2, 3)
(1, 2)
(3, 3)
(3, 2)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

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

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.

<u>Koliki je ukupan broj stanja u prostoru problema?</u>

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*)

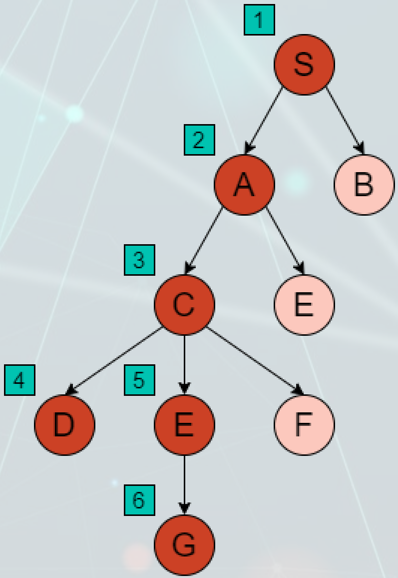
Zadatak 2 - Rešenje

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 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.

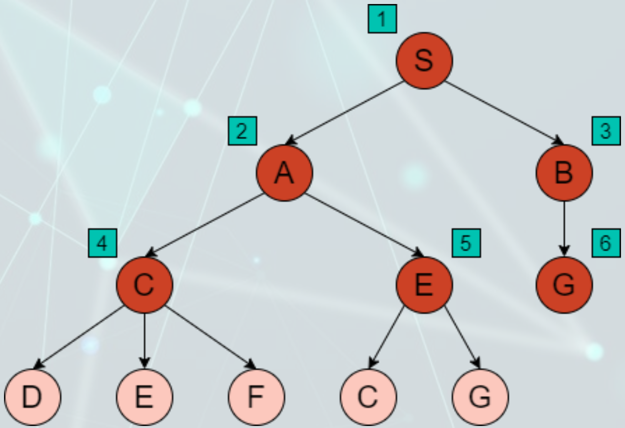
Zadatak 2 - Rešenje

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 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 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?

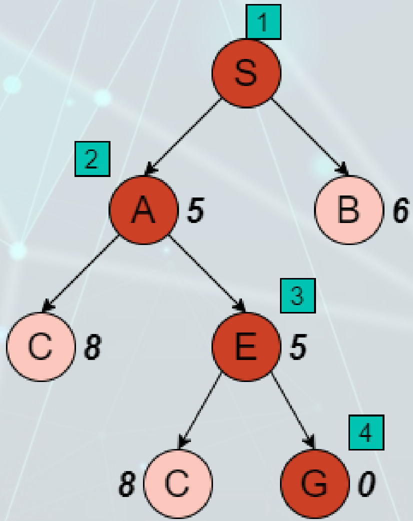
U tom slučaju nije garantovano optimalno rešenje problema.

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.

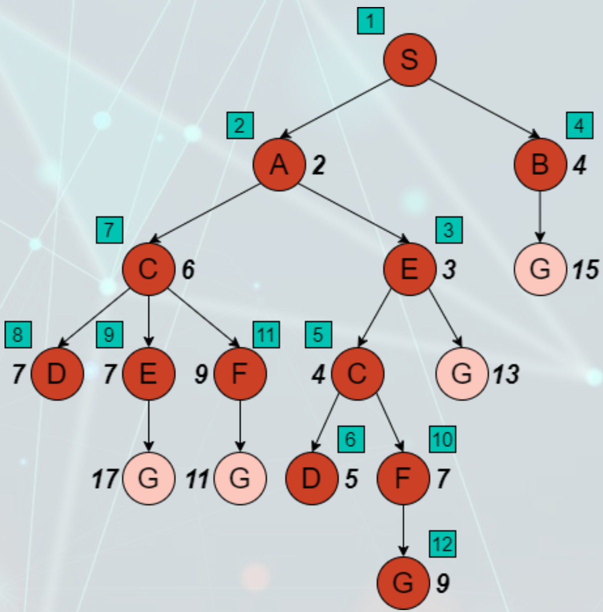
Zadatak 2 - Rešenje

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 sortira rastuće po 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

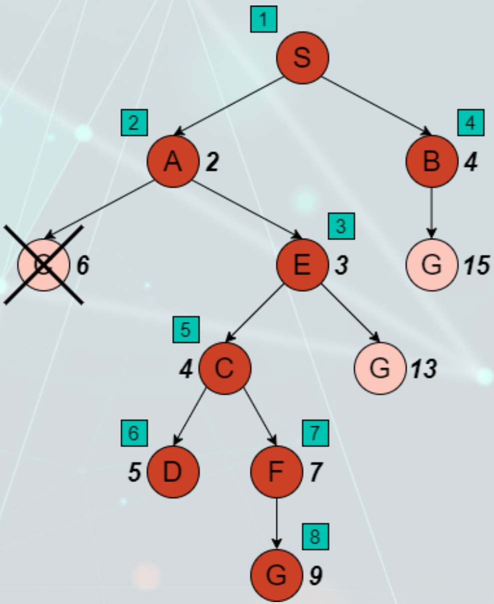
Grananje i ograničavanje (*branch and bound search*):

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.

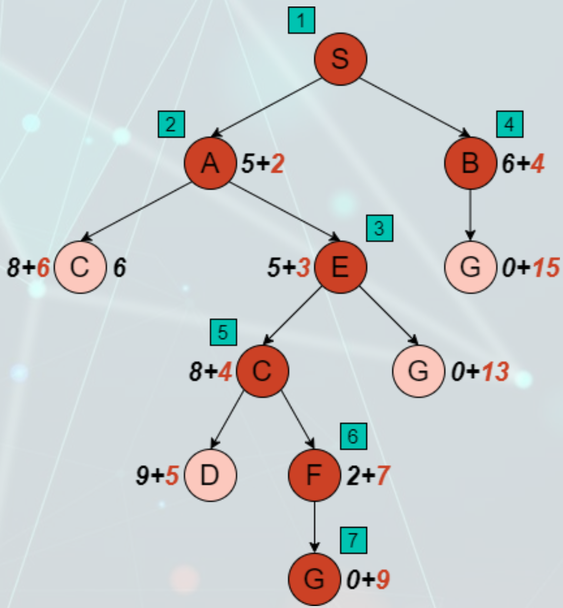
Zadatak 2 - Rešenje

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 svaku putanju izračunava 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 sortira rastuće po 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*):

<u>Da li je pronađena putanja optimalna?</u>

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

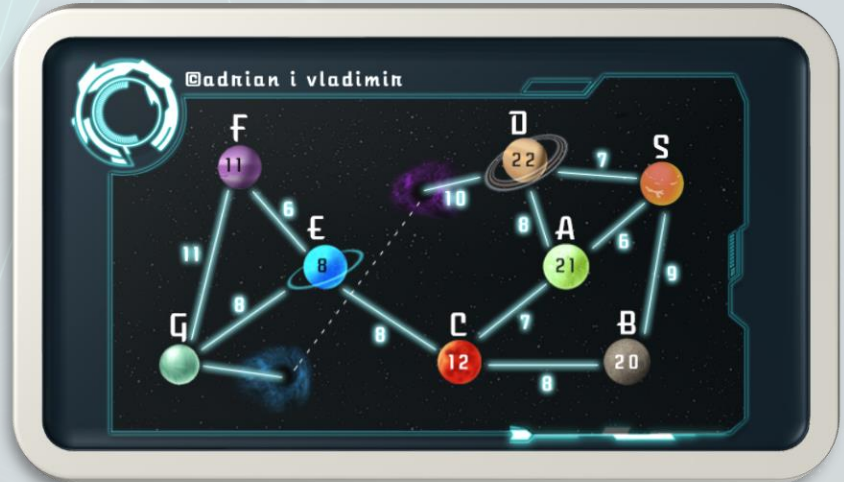
$$f(n) = g(n) + h(n)$$

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

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.

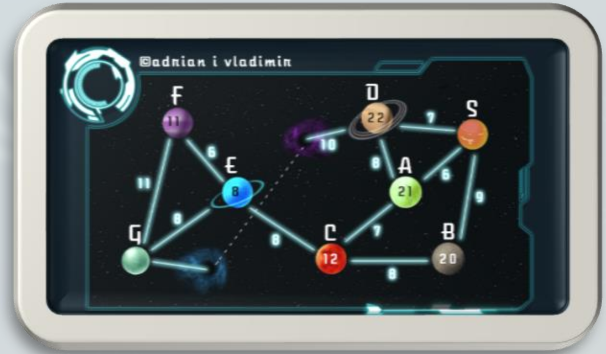


Zadatak za samostalnu vežbu - Svemir

★★☆☆☆

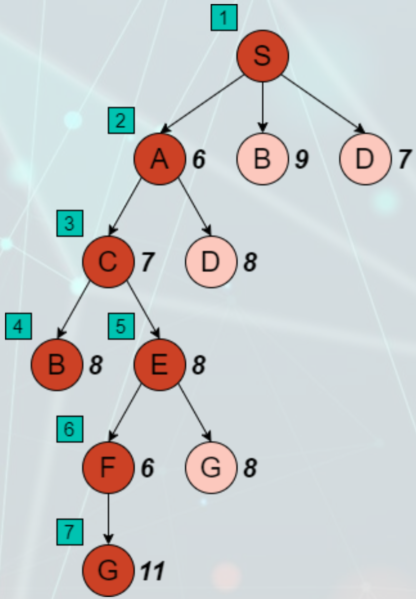
Prikazati stablo pretrage i navesti redosled obilaženja čvorova prilikom pronalaženja putanje između planeta S i G ukoliko se koriste sledeći algoritmi:

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

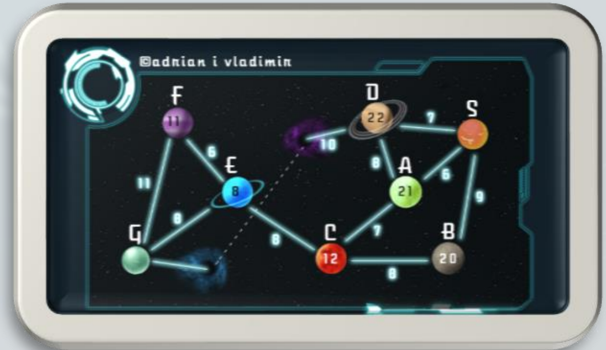


Zadatak za samostalnu vežbu - Rešenje

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

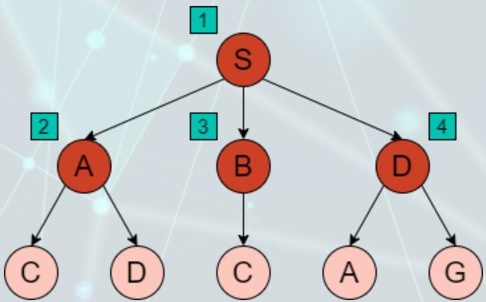


putanja: SACEFG
cena putanje: 38

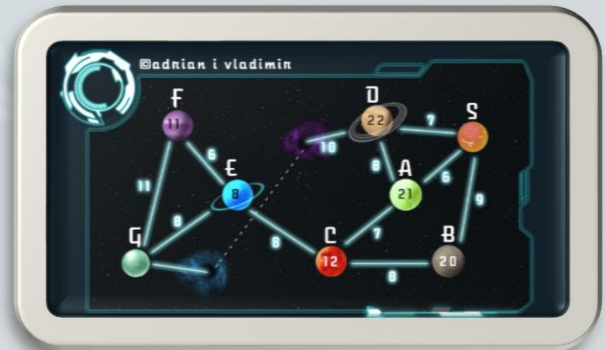


Zadatak za samostalnu vežbu - Rešenje

Pretraga po širini:

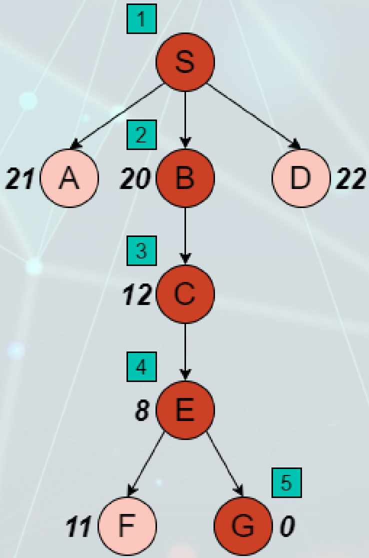


putanja: SDG
cena putanje: 17

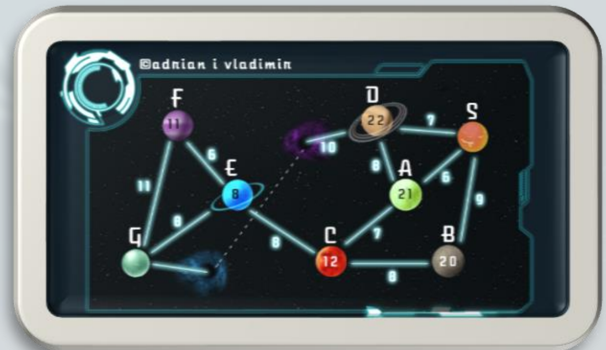


Zadatak za samostalnu vežbu - Rešenje

Prvo najbolji:

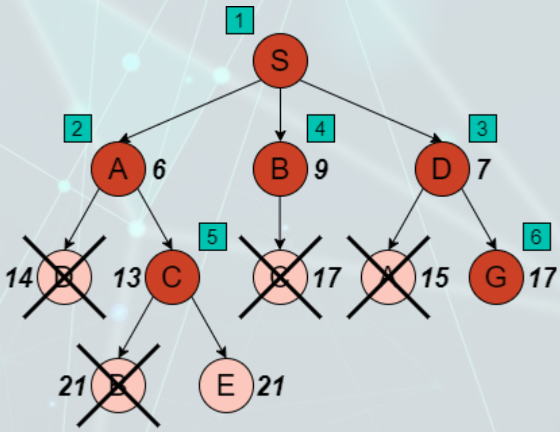


putanja: SBCEG
cena putanje: 33

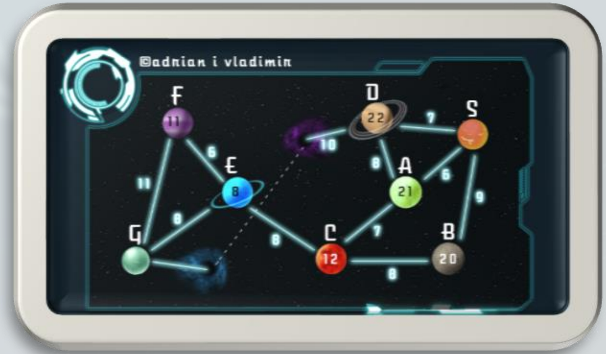


Zadatak za samostalnu vežbu - Rešenje

Grananje i ograničavanje:

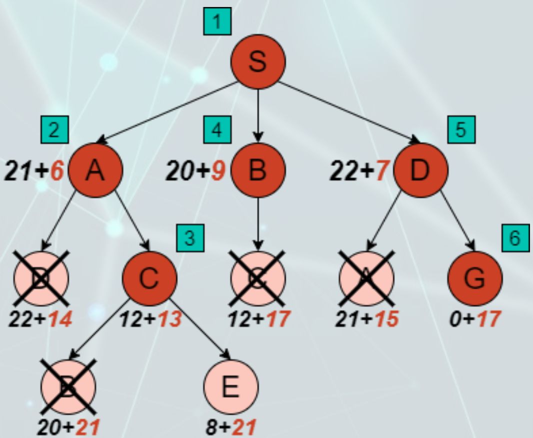


putanja: SDG
cena putanje: 17

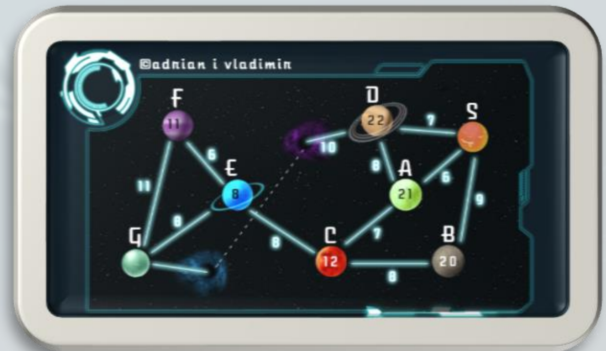


Zadatak za samostalnu vežbu - Rešenje

A*:



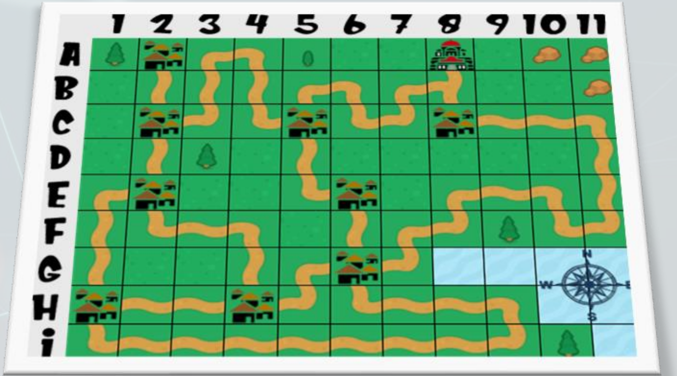
putanja: SDG
cena putanje: 17



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.

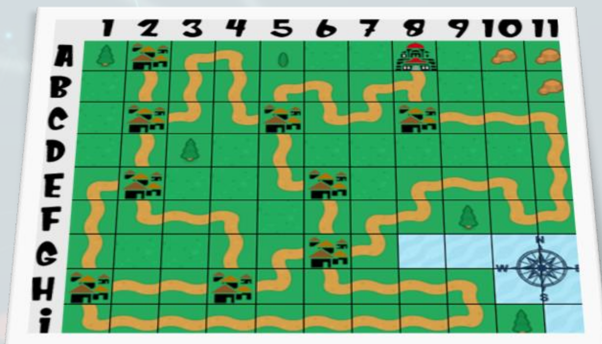


Zadatak 3 – Rođendan kralja Artura

★★★☆☆

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?



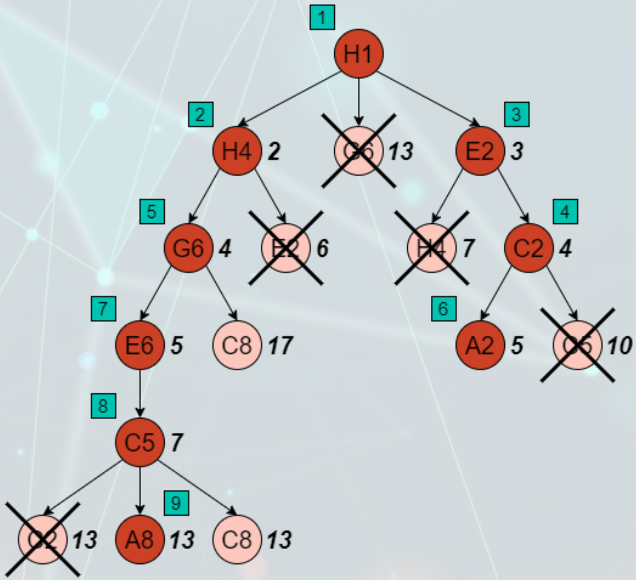
Zadatak 3 - Rešenje

Dinamičko programiranje

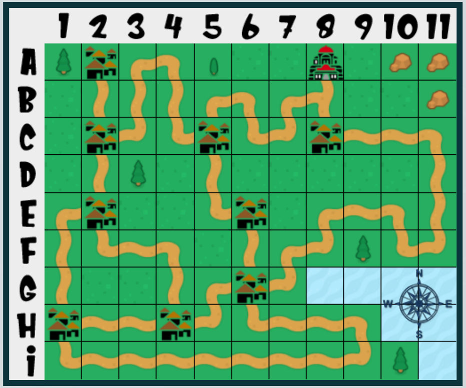
Predstavlja metod rešavanja problema svođenjem istog na njegov podproblem, a zatim rešavanjem tog podproblema po istom principu ili na trivijalan način. Svaki podproblem se tipično rešava samo jednom. Originalni problem se rešava kombinovanjem rešenja podproblema.

Zadatak 3 - Rešenje

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



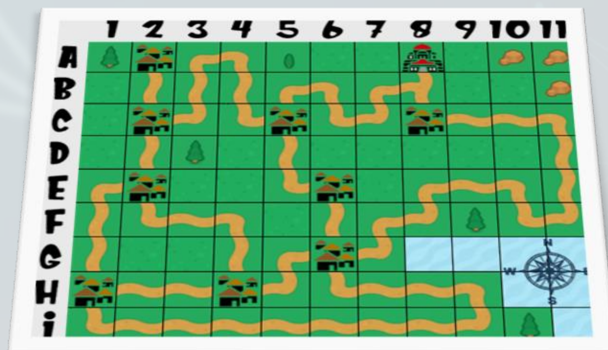
putanja:
H1-H4-G6-E6-C5-A8
cena putanje: 13



Zadatak 3 – Rođendan kralja Artura

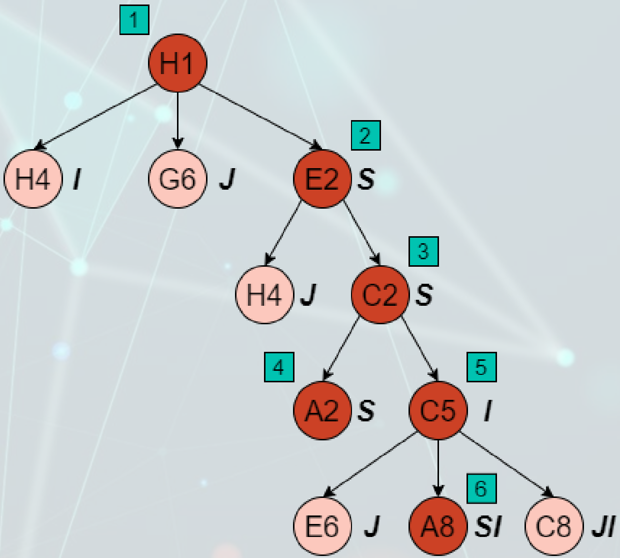
Vitezovi okruglog stola kreću sa Tristanovog rođendana ka Kamelotu odmah nakon Lanselota. Usled velike količine popijenog vina, nijedan vitez se nije setio da nabavi mapu. Jedino čega se sećaju jeste da se Kamelot u odnosu na Tristanovu kuću nalazi negde na severoistoku. Stoga, vitezovi odlučuju da ***na 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?

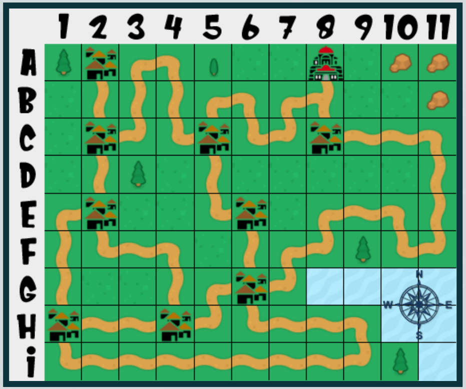


Zadatak 3 - Rešenje

Pretraga po dubini (*depth first search*):



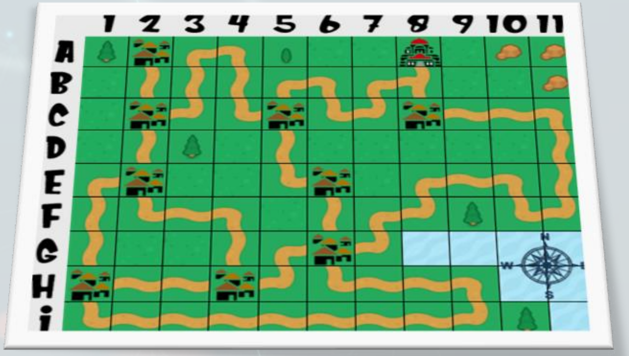
putanja:
H1-E2-**C2-A2-C2**-C5-A8
cena putanje: 18



Zadatak 3 – Rođendan kralja Artura

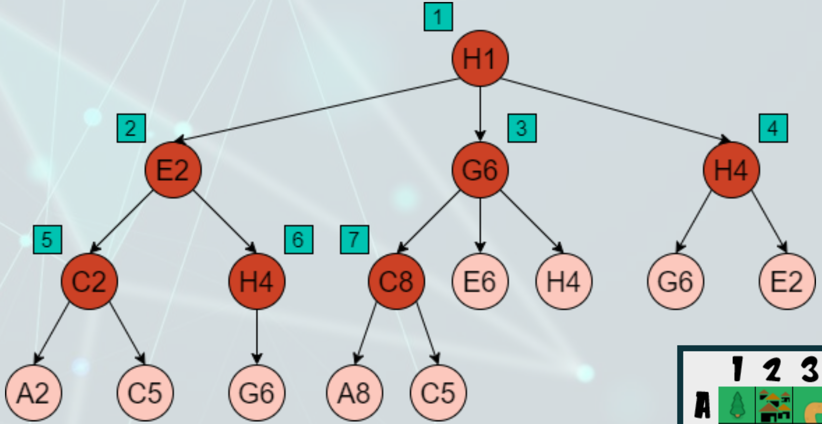
Vitez Tristan i njegova supruga Izolda kreću zajedno iz njihove kuće. Tristan ima običaj da se zadržava po selima, odmara i uživa u lokalnoj hrani i piću pre nego što nastavi put, stoga Izolda odlučuje da na osnovu mape osmisli poseban put do Kamelota za njih dvoje. Ona veruje da će brže stići u Kamelot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

Koji algoritam pretraživanja koriste Tristan i Izolda?

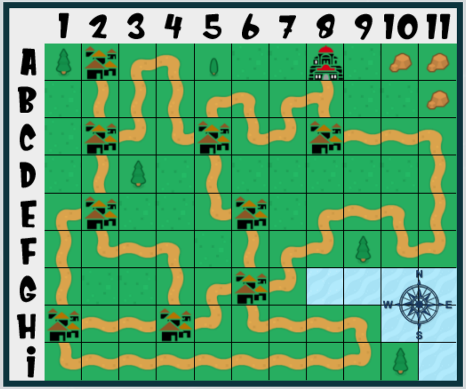


Zadatak 3 - Rešenje

Pretraga po širini (*breadth first search*):



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

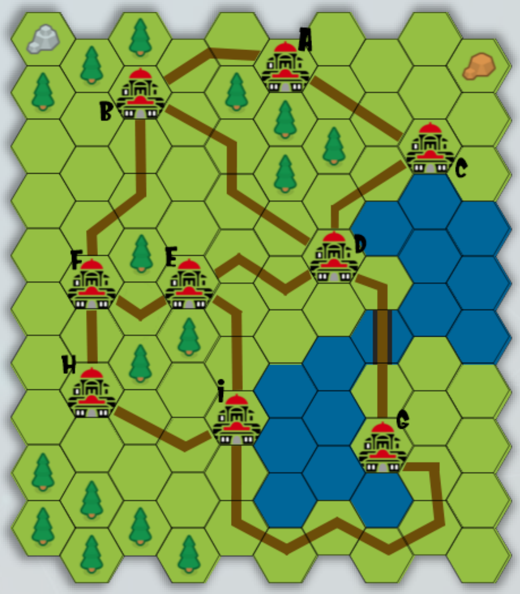


Zadatak za samostalnu vežbu - GoT

★★★★☆

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

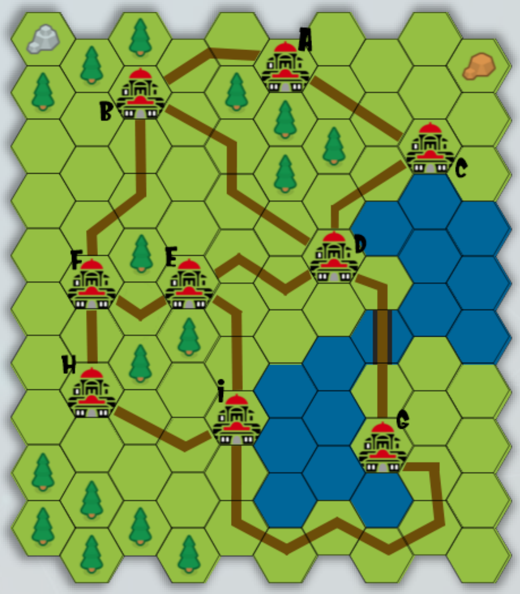
Lord Varis i lord Bejliš, deo Kraljevog malog veća, žele da saznaju što više o Tajvinu i vojsci koju dovodi u prestonicu, te šalju svoje špijune u bitne delove Vesterosa kako bi sakupili što više informacija. Tajvin i njegova vojska se nalaze u selu na poziciji čvora **A** u trenutku **t = 0** i kreću se putanjom **A – B – D – G – I**. Bejliš i Varis ne znaju putanju kojom će se kretati Tajvin i njegova vojska, te osmišljavaju 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 lord čiji je to špijun saznaje jednu novu informaciju od pijanih vojnika. Za svaki od sledećih obilazaka navesti algoritam pretraživanja koji se koristi, prikazati stablo pretrage, navesti redosled obilaska čvorova, dati putanju kojom će se špijuni kretati i odrediti koliko različitih informacija će saznati lordovi.

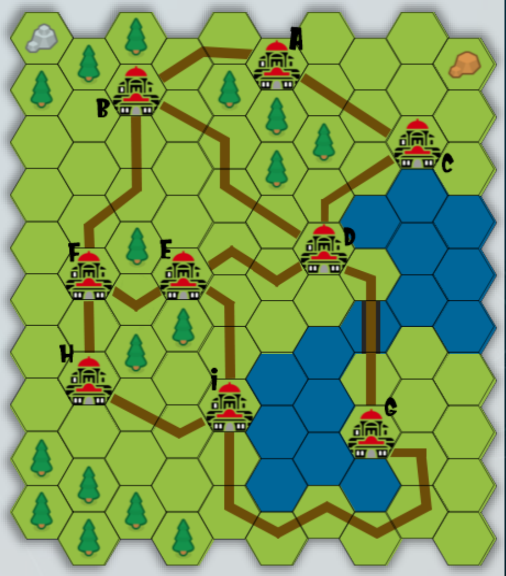


Zadatak za samostalnu vežbu - GoT

★★★★☆

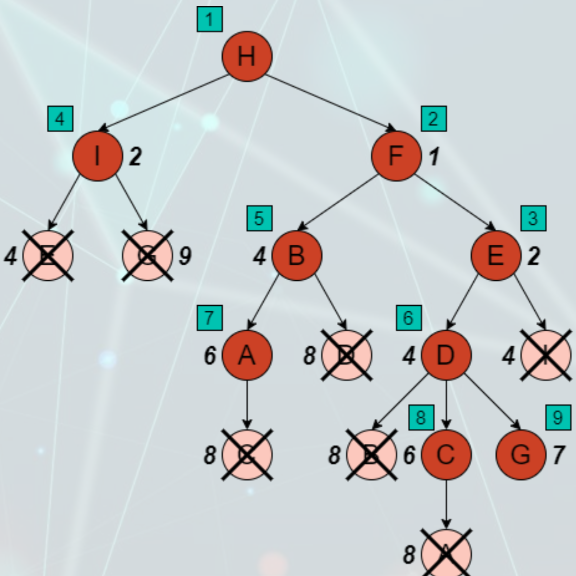
Smatrati da se po putevima svi kreću brzinom od jednog polja po danu, ne računajući polja sa mestima (npr. ukoliko je neko u mestu A bio u trenutku t = 0, onda će u mestu B biti u trenutku t = 2).

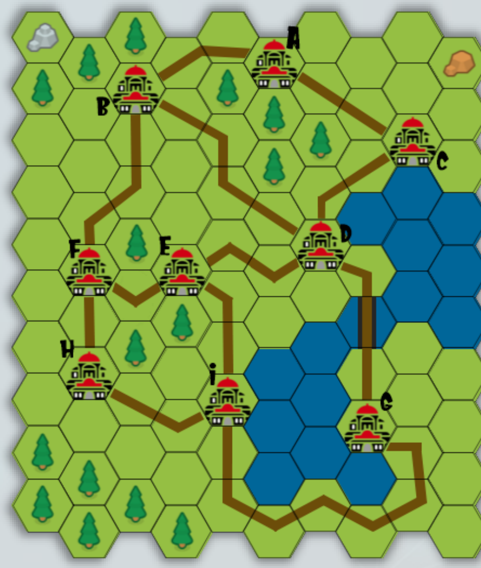
Varis šalje svoje špijune iz tačke H tako da u što kraćem vremenskom periodu zaposednu sva mesta na mapi deleći se u manje grupe po potrebi. Jednom kada zaposednu mesto, jedan od špijuna ostaje u tom mestu.



Zadatak za samostalnu vežbu - Rešenje

Grananje i ograničavanje:





broj sakupljenih informacija: 3

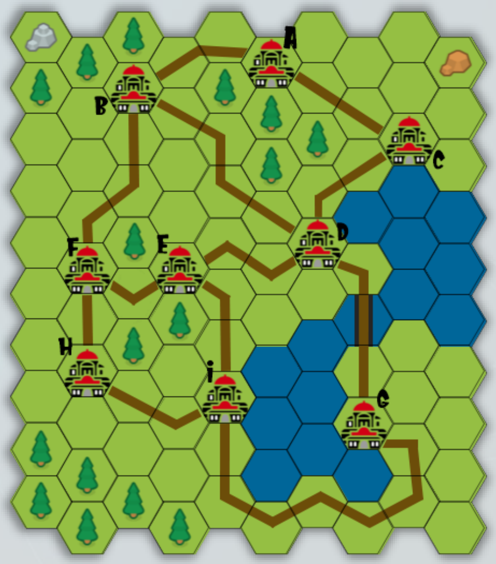
t	A	B	C	D	E	F	G	H	I
Tajvin	0	2	-	6	-	-	9	-	16
Varis	6	4	6	4	2	1	7	0	2

Zadatak za samostalnu vežbu - GoT

★★★★☆

Bejlišev najbolji špijun kreće pretragu iz mesta I. Špijun u svakom mestu po gavranima šalje pisma sa porukom „Da li se Tajvin nalazi u vašem selu?“ svim ostalim mestima na mapi.

Mesto u kojem se nalazi Tajvin u istom trenutku šalje potvrdan odgovor po gavranu. Na osnovu dobijenog odgovora špijun bira sledeće mesto za obilazak kao najbliže mestu u kojem se nalazi Tajvin (najbliže mesto je ono koje je udaljeno najmanji broj polja). Pretpostaviti da odgovor ne može da stigne dok je Tajvin na putu i da odgovor stiže istog dana kada je Tajvin u mestu. Špijun čeka u mestu u kom se nalazi dok ne dobije odgovor. U slučaju da se špijun i Tajvin nađu u istom mestu, špijin se infiltrira u vojsku i nastavlja da se kreće istim putem kao i Tajvin.

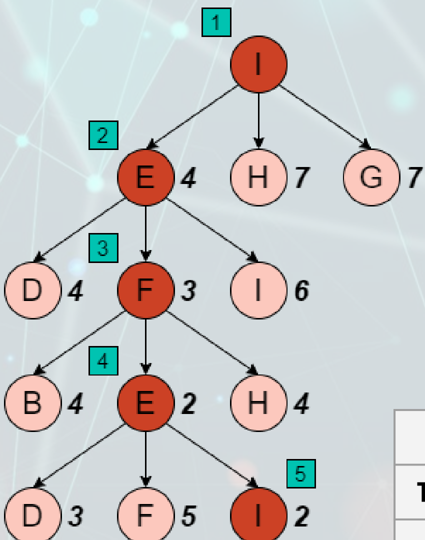


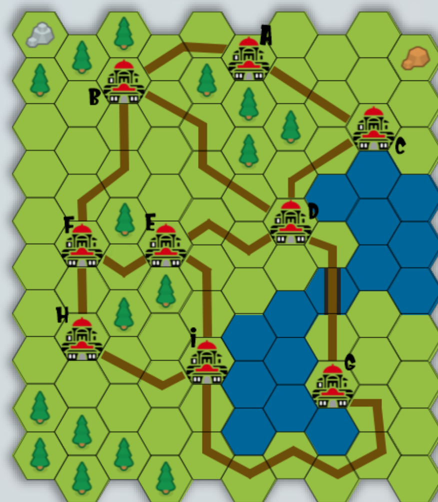
Zadatak za samostalnu vežbu - Rešenje

Pretraga po dubini sa heuristikom:

(heuristika je udaljenost čvora od čvora u kojem je Tajvin)

Dozvoljena su vraćanja nazad u gradove koji su obiđeni!





broj sakupljenih informacija: 1

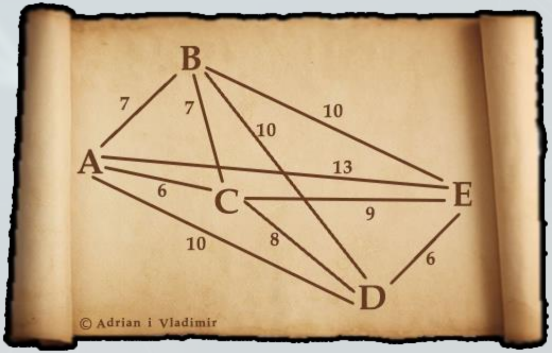
t	A	B	C	D	E	F	G	H	I
Tajvin	0	2	-	6	-	-	9	-	16
Bejliš	-	-	-	-	2-2 7-9	3-6	-	-	0-0 11-

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

<u>Problem trgovačkog putnika – NP problem (*NP hard*)</u>

- Jako teški za rešavanje (podproblemi su jednako kompleksni kao originalni problem)
- Pojedinačno rešenje se lako može verifikovati u polinomijalnom vremenu
- *Bruteforce* (permutacija broja gradova, *n!*), *Branch and bound*

Heuristička rešenja nisu uvek optimalna.

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

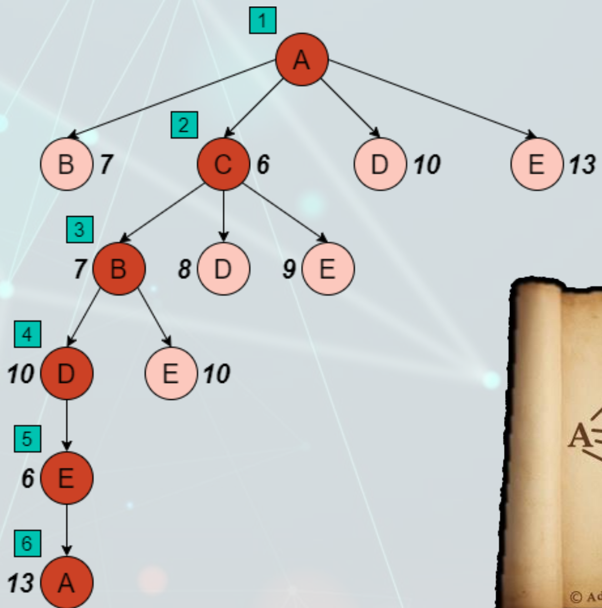
Zadatak 4 – Rešenje

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

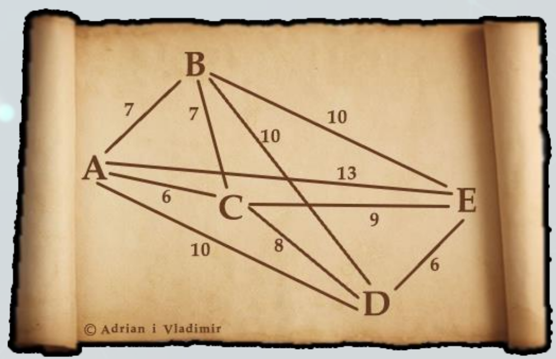
- Lista čvorova sadrži startni čvor
- Dokle god je lista čvorova neprazna:
    - Uklanja se čvor sa početka liste i ispituje da li je ciljni
        - Ako je u pitanju ciljni čvor, pretraga je završena
        - Ako nije u pitanju ciljni čvor, dodati njegove sledbenike (ukoliko postoje) u **sortiranom 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*** parcijalne putanje mogla bi se primeniti dužina ***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.

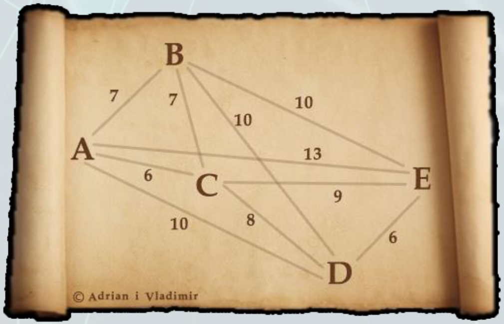
Kako se kreira MST?

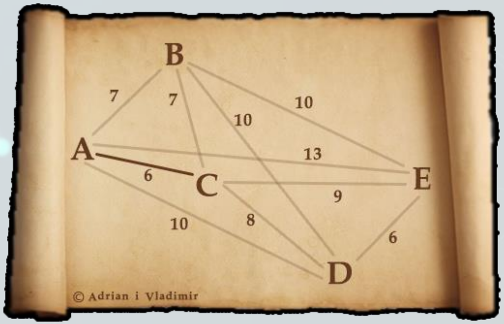
Npr. koristeći **Kruskalov** algoritam. Stablo inicijalno čine svi čvorovi i nijedna grana. Zatim se bira grana najmanje težine tako da se ne kreira petlja u stablu. Ovaj postupak se ponavlja sve dok se ne povežu svi čvorovi u jedinstveno stablo.

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.

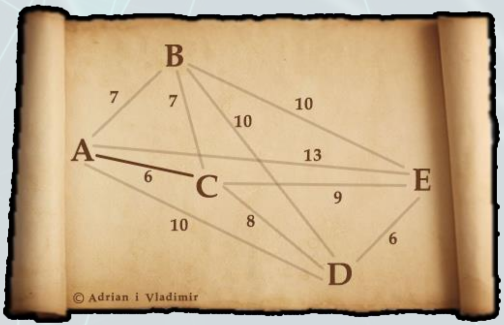


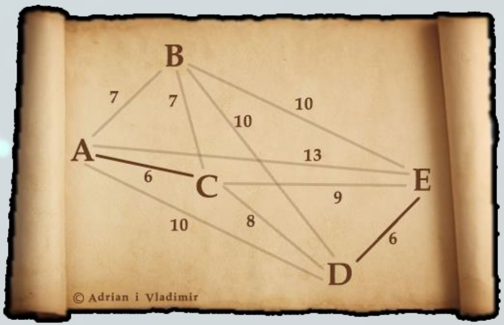


Zadatak 4 – Rešenje

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.

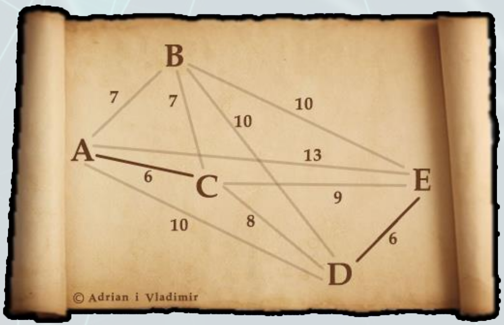


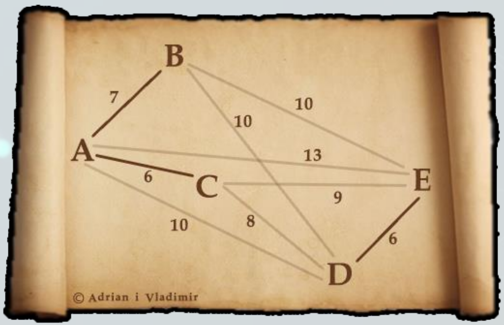


Zadatak 4 - Rešenje

<u>*Kako se kreira MST?*</u>

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.

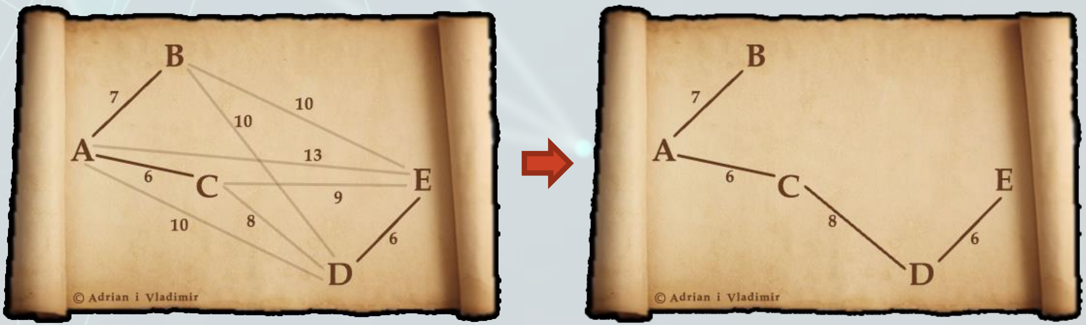




Zadatak 4 – Rešenje

<u>Kako se kreira MST?</u>

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.



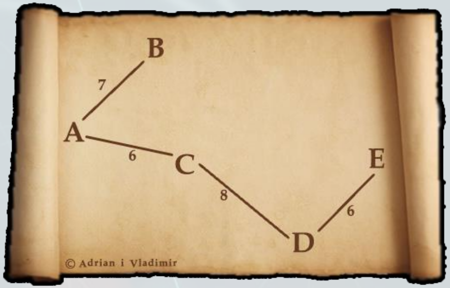
Zadatak 4 - Rešenje

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 parcijalne putanje putem čija je cena manja od cene MST-a.

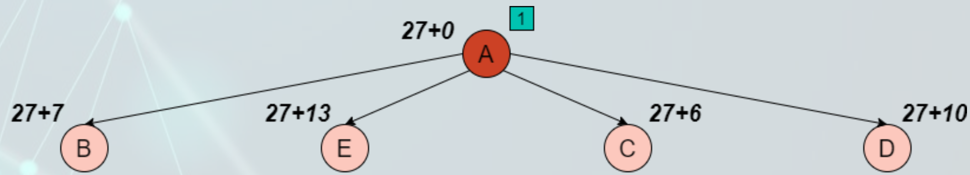
Zašto i startni i ciljni čvor parcijalne putanje čine MST?

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

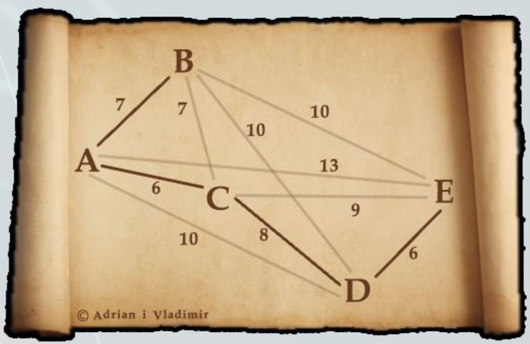


Zadatak 4 – Rešenje

A*:

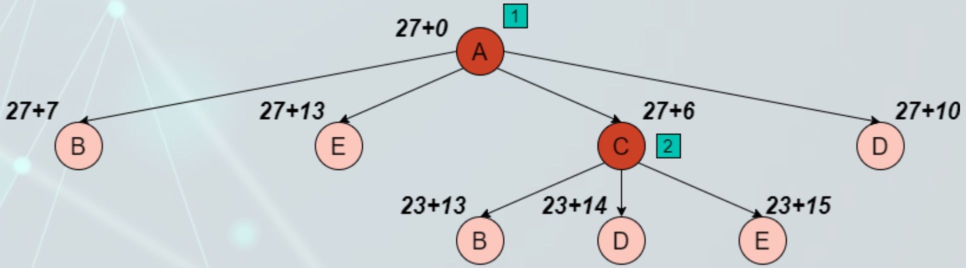


MST(BCDEA):
h = 27

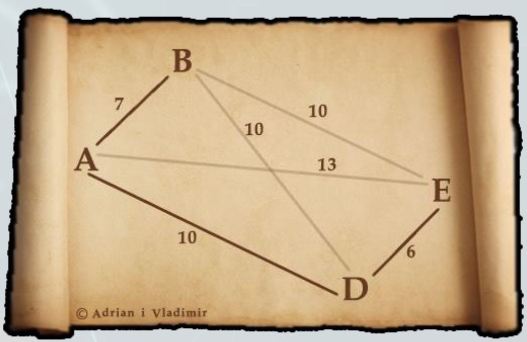


Zadatak 4 – Rešenje

A*:

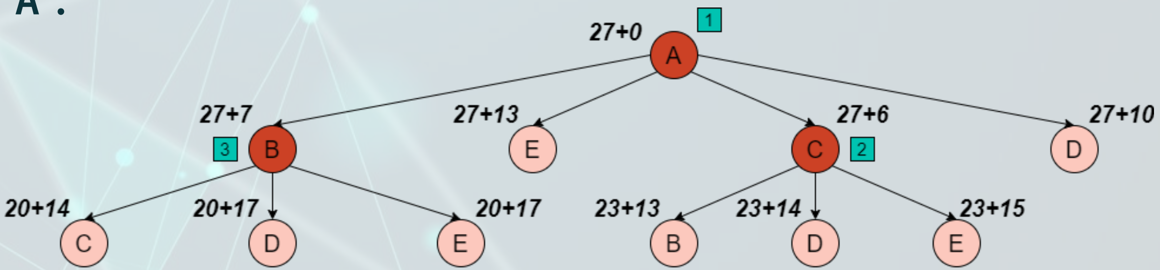


MST(BDEA):
h = 23

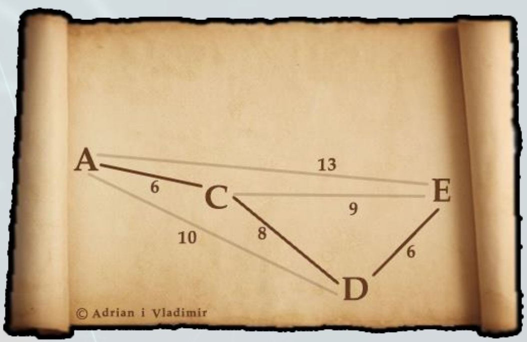


Zadatak 4 – Rešenje

A*:

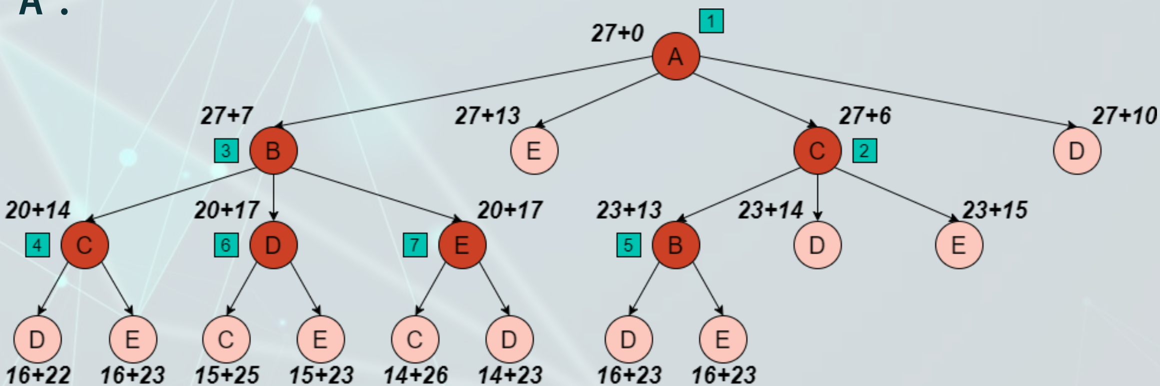


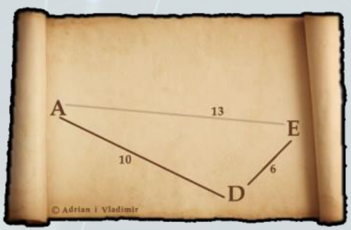
MST(CDEA):
h = 20

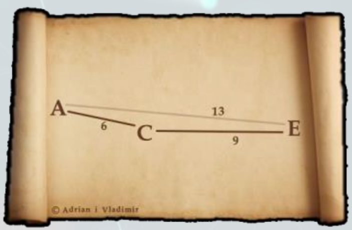


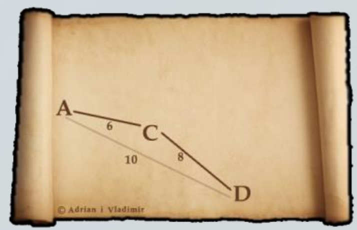
Zadatak 4 – Rešenje

A*:



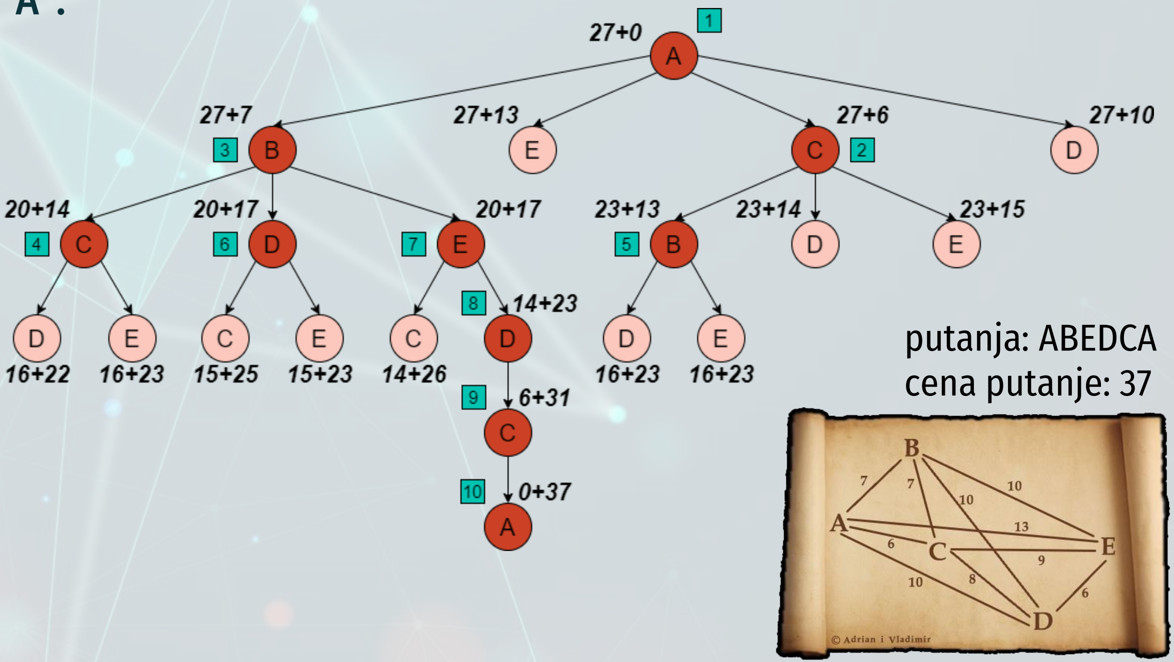






Zadatak 4 - Rešenje

A*:



putanja: ABEDCA

cena putanje: 37

Zadatak 5 - Množenje brojeva

★★★★★

Mali dečak Pera želi da pomnoži dva (za njega velika) pozitivna broja i u samom tom procesu donosi sledeće odluke (u zavisnosti od ispunjenosti određenih uslova):

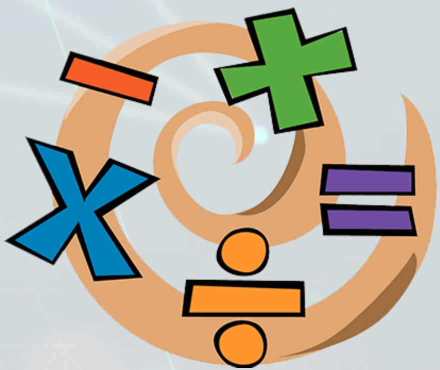
- Množenje dva broja, od kojih je barem jedan 10^x ili 0, može izvršiti direktno i cena takvog množenja je 0.
- Množenje dva broja od kojih svaki ima tačno jednu nenultu cifru (> 1) može izvršiti direktno i cena takvog množenja je 1.
- Množenje dva broja koji zajedno imaju tačno 3 nenulte cifre rešava primenjujući distributivnost tako da se broj sa 2 nenulte cifre prikaže u obliku A + B (A > B), pri čemu se A dobija postavljanjem nenulte cifre najmanje težine na vrednost 0. Inicijalna procena takvog množenja je 2.
- Množenje dva broja koji zajedno imaju više od 3 nenulte cifre rešava na jedan od dva načina prikazanih na sledećem slajdu (inicijalna procena takvog množenja je proizvod broja nenultih cifara u oba broja).

Zadatak 5 - Množenje brojeva

★★★★★

način 1: Primenjujući distributivnost na način opisan u prethodnom koraku razlaganjem broja sa više nenultih cifara ili broja većeg po apsolutnoj vrednosti ukoliko oba broja imaju jednak broj nenultih cifara.

način 2: Primenjujući Karacubin algoritam množenja.

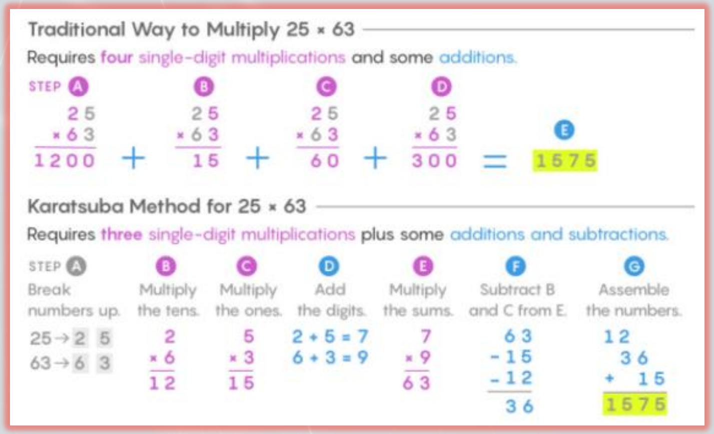


Zadatak 5 - Množenje brojeva

★★★★★

Karacubin algoritam množenja

Tradicionalni način množenja dva *n*-tocifrena broja zahteva *n^2* množenja pojedinačnih cifara. Ruski matematičar *Anatoly Karatsuba* je 1960. predložio bolji način, koji zahteva *n^1.58* takvih množenja.



Zadatak 5 - Množenje brojeva

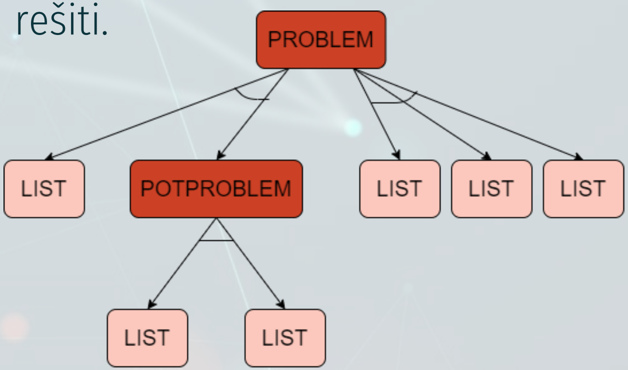
★★★★★

Napomena: Primena *k*-konektora koji povezuje problem sa podproblemima na koje je razložen (množenje dva broja sa parcijalnim množenjima) iznosi *k* jedinica.

Koristeći AO* algoritam prikazati rešenje ovog problema na AND/OR stablu pretrage.

Zadatak 5 - Rešenje

U **AND/OR** stablu pretrage čvorovi predstavljaju množenje dve vrednosti, gde koren stabla predstavlja množenje dve inicijalne vrednosti. Od svakog unutrašnjeg čvora (osim listova) vode grane ka čvorovima naslednicima, koji predstavljaju podprobleme na koje se čvor-problem može razložiti. Listovi stabla predstavljaju terminalne situacije, koje se ne razlažu na podprobleme i koje se direktno mogu rešiti.



Zadatak 5 – Rešenje

Rešenje problema možemo prikazati AND/OR stablom.

- Stanja su predstavljena čvorovima stabla.
- Grane stabla (**konektori**) povezuju čvorove-probleme sa svojim naslednicima preko kojih su definisani.
- Moguće je dodeliti cene konektorima (što odgovara ceni primene akcije/pravila) kao i samim čvorovima (cene rešavanja problema).
- Čvor povezan sa svojim naslednicima jednim k-konektorom naziva se **AND čvor**. Čvor povezan sa dva ili više k-konektora sa svojim naslednicima naziva se **OR čvor**.

Zadatak 5 - Rešenje

Kako vršiti pretragu AO stabla?

Rešenje problema predstavlja podstablo **T’** stabla **T**, koji definiše problem koji se rešava.

Polazi se od startnog čvora i bira se jedan od konektora. **Konektori** povezuju čvor-problem sa njegovim naslednicima (potproblemima preko kojih je čvor problem definisan). Ukoliko su svi čvorovi po tom konektoru rešeni onda je rešenje pronađeno. U suprotnom, za svaki od čvorova-naslednika bira se jedan konektor koji se, zajedno sa naslednicima tog čvora, uključuje u trenutno rešenje problema.

Zadatak 5 - Rešenje

Kako vršiti pretragu AO stabla?

Algoritam se sastoji iz dve faze: **ekspanzije izabranog čvora** i **revizije funkcije procene čvorova grafa**.

Polazi se od startnog čvora, čija je inicijalna procena jednaka heurističkoj vrednosti. Ekspanzijom čvora u rešenje se uvode i konektori i čvorovi preko kojih je čvor definisan i čije su inicijalne procene jednake njihovim heurističkim vrednostima. Zatim se vrši računanje funkcije procene za razvijani čvor po svim konektorima i vrši se izbor konektora po kome je funkcija procene najmanja. Taj konektor se označava strelicom kao aktivan i ažurira se funkcija procene razvijanog čvora.

Zadatak 5 - Rešenje

Kako vršiti pretragu AO stabla?

Zatim se za ekspanziju bira jedan od čvorova-naslednika koji se nalazi u trenutnom rešenju (najboljoj putanji) i algoritam se izvršava rekurzivno.

Ukoliko se u nekom trenutku funkcija procene čvora promeni onda je neophodno ažurirati funkcije procene svih čvorova na koje ova izmena utiče (čvorovi koji se nalaze u trenutnom rešenju, a definisani su preko čvorova čija se funkcija procene promenila). Ta propagacije izmene se vrši ka startnom čvoru grafa, dok god ima promene u funkciji procene čvorova.

Zadatak 5 – Rešenje

Kako vršiti pretragu AO stabla?

Izmena funkcije procene čvorova može dovesti do izbora nekog drugog konektora, preko koga je funkcija procene čvora možda postala manja od ažurirane funkcije procene čvora preko konektora koji čini trenutno rešenje. Tada taj drugi konektor postaje aktivan i vrednost funkcije procene čvora preko tog konektora se propagira unazad.

Redosled izbora konektora je proizvoljan ukoliko je funkcija procene čvora preko više konektora identična.

Zadatak 5 - Rešenje

Kako vršiti pretragu AO stabla?

Ciljni čvorovi su uvek označeni kao rešeni.

Da bi AND čvor bio rešen moraju se pronaći rešenja svih njegovih naslednika, a da bi OR čvor bio rešen dovoljno je pronaći rešenje jednog od njegovih naslednika.

Ukoliko je čvor označen kao rešen ta informacija se propagira unazad čvorovima preko kojih je taj čvor definisan.

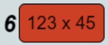
Zadatak 5 - Rešenje

AO* (*AO star search*):

- Stablo T’ inicijalno sadrži samo startni čvor stabla T
- Dokle god ima neekspandovanih čvorova u stablu T’:
  - Uzima se neekspandovani čvor stabla T’
    - Ako je u pitanju ciljni čvor, označiti ga kao rešen i propagirati ovu informaciju svim čvorovima preko kojih je čvor definisan
    - Ukoliko je startni čvor označen kao rešen prekinuti pretragu
    - Izračunati funkcije procene čvora preko svih konektora. U stablo T’ ubaciti onaj konektor i čvorove naslednike preko koga je funkcija procene čvora najmanja, a izbaciti prethodno aktivni konektor (ukoliko postoji) sa čvorovima. Ukoliko je došlo do izmene funkcije procene propagirati tu informaciju unazad.
- Ako je startni čvor grafa T’ označen kao rešen, pretraga je uspešna.

Zadatak 5 - Rešenje

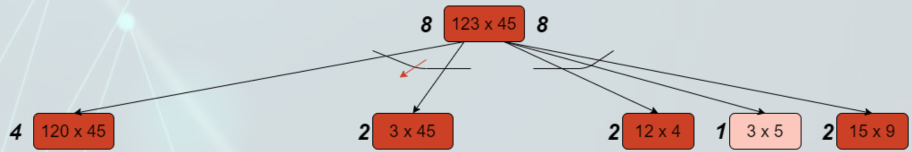
Rešenje:



Čvor **123x45** ima inicijalnu funkciju procene 6 koja je dobijena na osnovu heuristike iz teksta zadatka. Kako čvor ima više od 4 nenulte cifre, funkcija procene se dobija kao proizvod broja nenultih cifara u levom i desnom operandu (3 x 2 = 6).

Zadatak 5 - Rešenje

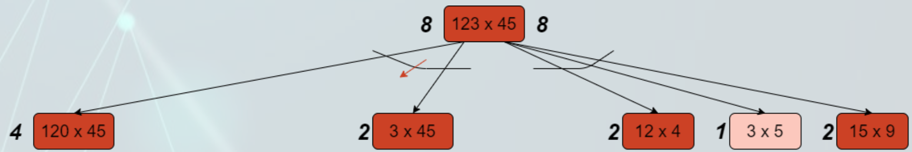
Rešenje:



Čvor **123x45** se može rešiti na jedan od dva načina (primenom distributivnosti ili Karacubinim metodom). Primenom distributivnosti čvor **123x45** može da se razloži na potprobleme rešavanja proizvoda brojeva **120x45** i **3x45**. Za svaki od čvorova se računaju funkcije procene po heuristici iz postavke zadatka (čvor **120x45** ima četiri nenulte cifre pa je njegova funkcija procene proizvod nenultih cifara levog i desnog operanda; čvor **3x45** ima tri nenulte cifre pa je njegova funkcija procene 2). Primenom Karacubinog metoda čvor **123x45** može da se razloži na potprobleme rešavanja proizvoda brojeva **12x4**, **3x5** i **15x9**. Čvor **3x5** može direktno da se reši te je on na slici označen kao rešen.

Zadatak 5 - Rešenje

Rešenje:



Funkcije procene korenog čvora po levom i desnom konektoru se računaju kao zbir funkcija procene svih čvorova-dece i cene konektora.

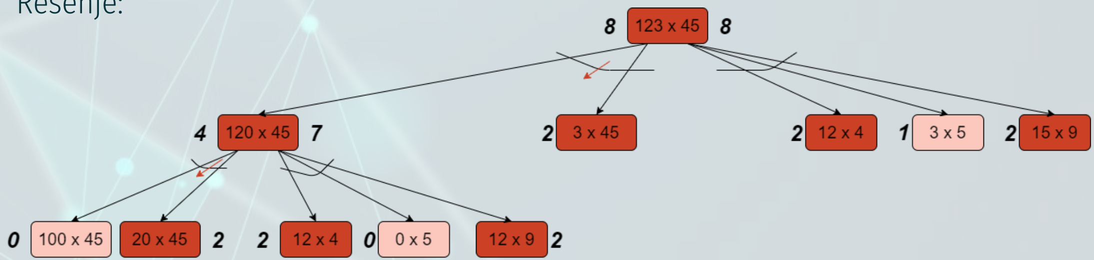
Funkcija procene po levom konektoru je 4 + 2 + k(2) = 4 + 2 + 2 = 8.

Funkcija procene po desnom konektoru je 2 + 1 + 2 + k(3) = 2 + 1 + 2 + 3 = 8.

Kako su funkcije procene po oba konektora jednake, proizvoljno označavamo levi konektor aktivnim i prelazimo na rešavanje njegovih potproblema.

Zadatak 5 - Rešenje

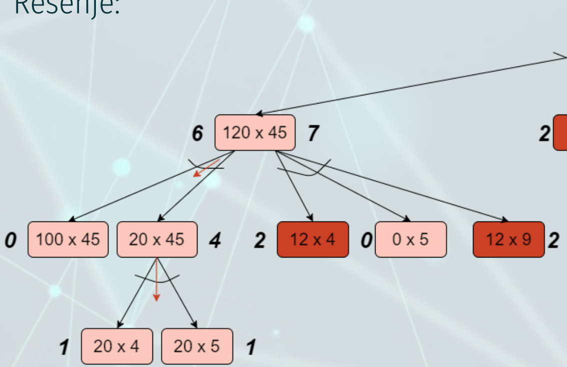
Rešenje:



Čvor **120x45** se može rešiti na jedan od dva već pomenuta načina. Uz svaki čvor date su funkcije procene. Kako je po levom konektoru funkcija procene bolja, levi konektor se označava kao aktivan. Nije došlo do izmene funkcije procene čvora **120x45** pa se ne menja funkcija procene korenog čvora.

Zadatak 5 - Rešenje

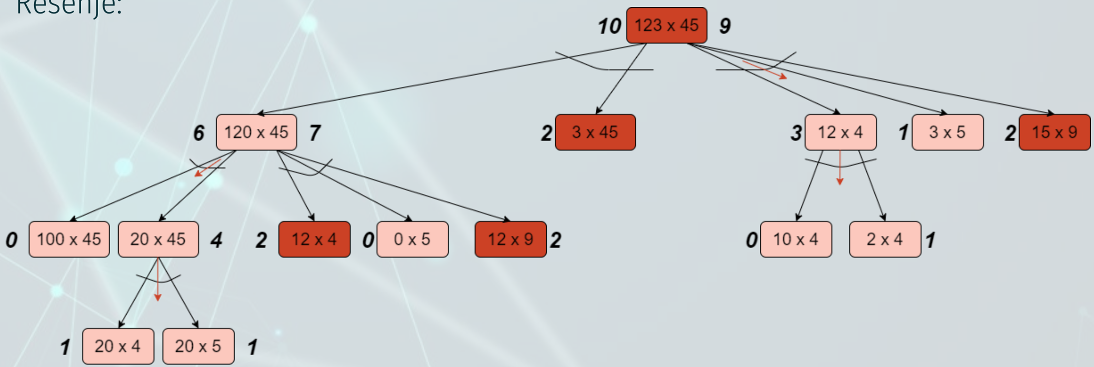
Rešenje:



Čvorovi **20x4** и **20x5** su označeni kao rešeni pa je njihov čvor-otac takođe označen kao rešen. Čvor **120x45** je takođe označen kao rešen jer su po jednom njegovom konektoru sada rešeni svi čvorovi-deca. Dodatno, funkcija procene čvora **20x45** se ažurira na 1 + 1 + k(2) = 4, a po istom principu se funkcija procene čvora **120x45** ažurira na 0 + 4 + k(2) = 6. Funkcija procene korenog čvora je sada 6 + 2 + k(2) = 10, što znači da se desni konektor sada označava kao aktivni jer ima bolju funkciju procene.

Zadatak 5 - Rešenje

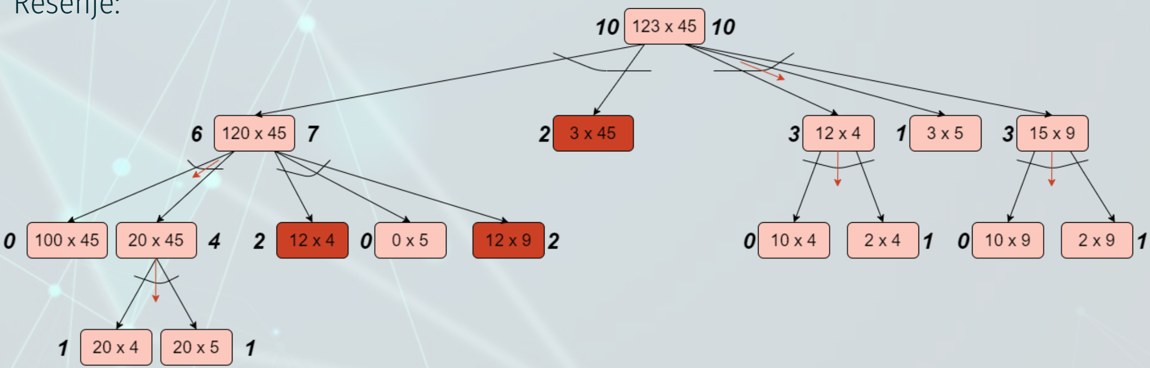
Rešenje:



Čvorovi **10x4** i **2x4** su rešeni pa je njihov čvor-otac takođe označen kao rešen. Dodatno, funkcija procene čvora **12x4** se ažurira na 0 + 1 + k(2) = 3, a po istom principu se funkcija procene korenog čvora ažurira na 3 + 1 + 2 + κ(3) = 9

Zadatak 5 - Rešenje

Rešenje:



Kako su po desnom konektoru svi čvorovi naslednici rešeni, tada je i koreni čvor rešen i rešenje problema je dato aktivnim konektorima.

Zadatak za samostalnu vežbu - Zamena

★★★★☆

Mala devojčica Maša želi da prikaže broj 6 kao niz jedinica i u samom tom koristi sledeća pravila za zamenu brojeva na levoj strani nizom brojeva na desnoj strani:

- Broj 6 može da zameni nizom brojeva 3, 3 ili nizom brojeva 4, 2.
- Broj 4 može da zameni nizom brojeva 2, 2 ili nizom brojeva 3, 1.
- Broj 3 može da zameni nizom brojeva 2, 1.
- Broj 2 može da zameni nizom brojeva 1, 1.

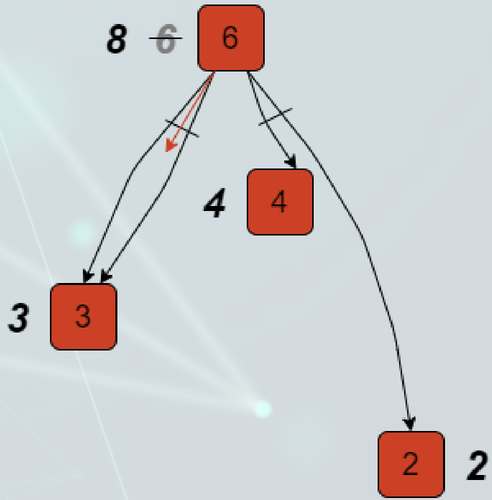
Problem rešiti primenom AO* algoritma na AND/OR grafu pretrage. Usvojiti da je cena *k*-konektora jednaka *k*. Vrednost heurističke funkcije *h* čvora označenom brojem 1 je 0. Vrednost heurističke funkcije *h* čvora označenog brojem *n* je *n*.

Zadatak za samostalnu vežbu - Rešenje

Rešenje:

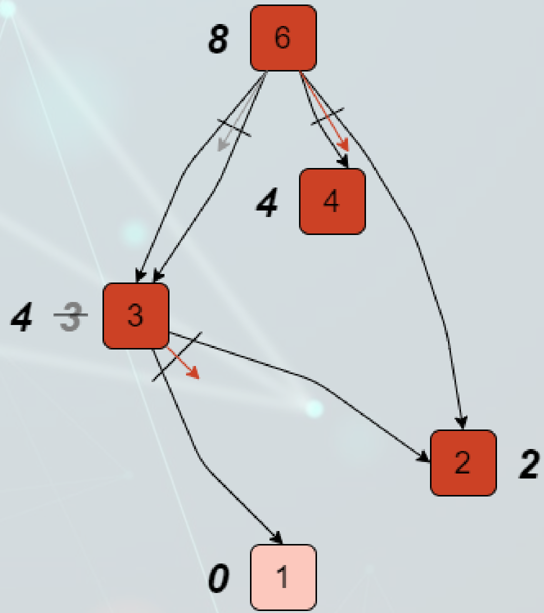
Zadatak za samostalnu vežbu - Rešenje

Rešenje:



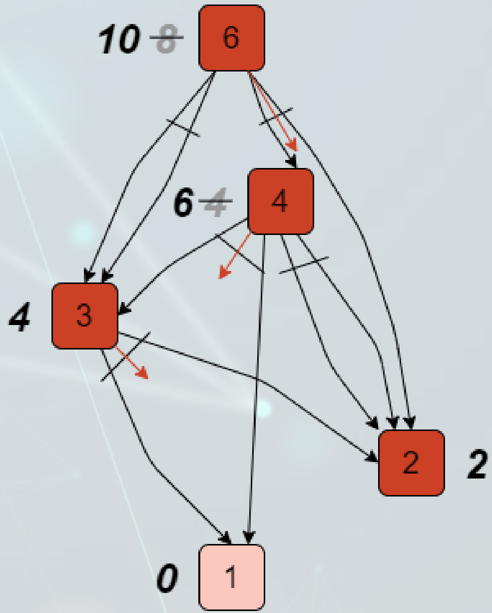
Zadatak za samostalnu vežbu - Rešenje

Rešenje:



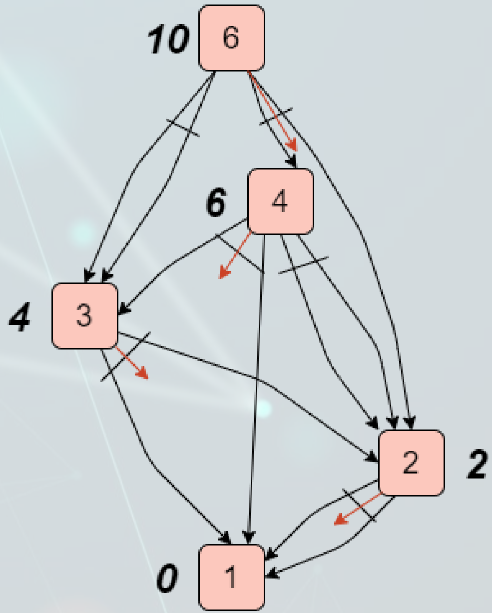
Zadatak za samostalnu vežbu - Rešenje

Rešenje:



Zadatak za samostalnu vežbu - Rešenje

Rešenje:



PITANJA?

http://ri4es.etf.rs/