# Rešenje zadataka iz Algoritamskih heuristika

## Pokretanje programa

Na Linuksu se programi pokreću tako što se otvori terminal u odgovarajućem folderu (task1, task2 ili task3) i ukuca se komanda *make* (da bi se kompajlirao C++ kod). Nakon toga će se generisati objektni fajl. Programe pokrećemo pomoću komande *./main*

Na Windowsu se dati programi mogu pokrenuti iz nekog IDE-a, na primer CodeBlocks-a. Neophodno je skinuti CodeBlocks sa MinGW kompajlerom, napraviti nove projekte (konzolne aplikacije), ubaciti odgovarajuće main.cpp i matrix.in fajlove i potom pokrenuti svaki projekat.

## Prvi zadatak – A\* algoritam

**Ulazni podaci** se nalaze u datoteci matrix.in Sledećeg su oblika:

broj redova, broj kolona [8 8]

početna x pozicija, početna y pozicija [2 5]

krajnja x pozicija, krajnja y pozicija [4 1]

Oblik matrice (1 ako nema prepreke, 0 ako ima)

**Izlazni podaci** se ispisuju u konzolu. Ispisuju se dve matrice:

matrica f – predstavlja cene obiđenih elemenata matrice. Vrednosti funkcije cene *f(x)* za svako polje u matrici su prikazane u tabeli 1 (slučaj pod a – *h(i) = Manh\_distxy(T, i)*) i tabeli 2 (slučaj pod b – *h(i) = max(Manh\_distx(T, i), Manh\_disty(T, i))* ). Plavom bojom je označen izvorišni čvor, crvenom uvorišni čvor. Sivom bojom su označene prepreke, a svetlo zelenom **pronađena putanja**.

matrica prev - govori ko je “roditelj” obiđenih elementa matrice(u obliku *y\_pozicija\*10 + x\_pozicija*, recimo vrednost na poziciji [2 3] će biti 24 ako je njegov roditelj element na poziciji [2 4]). Roditelj ima analognu definiciju kao u teoriji grafova, element koji je prethodnik nekom drugom elementu.

Članovi matrice koji mogu da budu posećeni se pamte u prioritetnom redu (engl. *priority queue*). Razlika u odnosu na običan red je u tome što se na vrhu tog reda uvek nalazi element sa najvećom vrednošću, što će u ovom slučaju biti funkcija cene, *f(x)*. Već obiđeni članovi matrice se skladište u strukturi podataka zvanoj skup (engl. *set*) – data struktura nije numerisana (odnosno sortirana) i omogućava povrat vrednosti u O(logn).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 12 | 10 | 8 | 8 | 8 | 10 | 12 | 14 |
| 1 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 2 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 3 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 4 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 5 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 6 | 12 | 10 | 8 | 8 | 8 | 10 | 12 | 14 |
| 7 | 14 | 12 | 10 | 10 | 10 | 12 | 14 | 16 |

Tabela 1 Rešenje pod a)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 11 | 9 | 7 | 7 | 8 | 9 | 11 | 13 |
| 1 | 10 | 8 | 6 | 6 | 6 | 8 | 10 | 12 |
| 2 | 9 | 7 | 5 | 5 | 6 | 7 | 9 | 11 |
| 3 | 8 | 6 | 4 | 5 | 6 | 7 | 8 | 10 |
| 4 | 7 | 5 | 4 | 5 | 6 | 7 | 8 | 9 |
| 5 | 6 | 5 | 4 | 5 | 6 | 7 | 8 | 9 |
| 6 | 8 | 7 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 9 | 8 | 9 | 10 | 11 | 12 | 13 |

Tabela 2 Rešenje pod b)

## Drugi zadatak – Keringan-Linov algoritam

**Ulazni podaci** se nalaze u datoteci matrix.in. Sledećeg su oblika:  
Širina, visina [10 8]  
Broj iteracija [5]  
Matrica incidencije netova i komponenti [tabela 3].

**Izlazni podaci** se ispisuju u konzolu. Izlazni podaci su u obliku:  
oznaka kapije (slovo od a do j – slika 2): broj 1 ili 2 (da li pripada prvoj ili drugoj particiji). Ova linija se ponavlja 10 puta, za svaku kapiju.  
Broj netova koji se moraju preseći za takav raspored.

Pronađeno **rešenje** nakon 5 iteracija je: (a, b, d, f, i) je jedna particija, (c, e, g, h, j) druga particija. Moraju se preseći 3 neta.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h | i | j |
| n1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| n2 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| n3 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| n4 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| n5 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| n6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| n7 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| n8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

Tabela 3 Matrica incidencije netova i komponenti

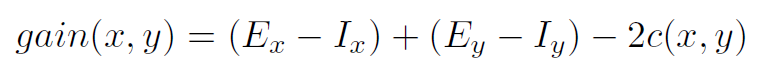
Od matrice incidencije netova i komponenti je razvijen graf u kojem su čvorovi komponente, a težine grana se određuju po formuli opisanom u datom algoritmu (tabela 4, slika 1):

*We typically use so called the kclique model, where a net that contains k gates forms a k-clique in G, and each edge in the clique gets a weight of 1/(k − 1). In case an edge (x, y) already exists from a prior net conversion, we just add the new weights instead of adding a parallel edge.*

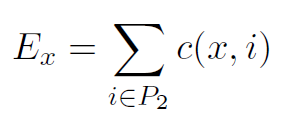
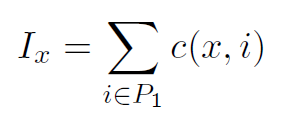
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h | i | j |
| a | 0 | 0 | 0 | 0.5 | 0 | 0.5 | 0 | 0 | 0 | 0 |
| b | 0 | 0 | 0 | 0.25 | 0.25 | 0 | 0.25 | 0.25 | 0 | 0 |
| c | 0 | 0 | 0 | 0 | 0.5 | 0 | 0 | 0.5 | 0 | 0 |
| d | 0.5 | 0.25 | 0 | 0 | 0.25 | 0.5+0.5 | 0.5+0.25 | 0.25 | 0 | 0 |
| e | 0 | 0.25 | 0.5 | 0.25 | 0 | 0 | 0.33+0.25 | 0.33+0.5+0.25 | 0 | 0.33 |
| f | 0.5 | 0 | 0 | 0.5+0.5 | 0 | 0 | 0.5 | 0 | 1 | 0 |
| g | 0 | 0.25 | 0 | 0.5+0.25 | 0.33+0.25 | 0.5 | 0 | 033+0.25 | 0.5 | 0.5+0.33 |
| h | 0 | 0.25 | 0.5 | 0.25 | 0.25+0.5+0.33 | 0 | 0.25+0.33 | 0 | 0 | 1+0.33 |
| i | 0 | 0 | 0 | 0 | 0 | 1 | 0.5 | 0 | 0 | 0.5 |
| j | 0 | 0 | 0 | 0 | 0.33 | 0 | 0.5+0.33 | 1+0.33 | 0.5 | 0 |

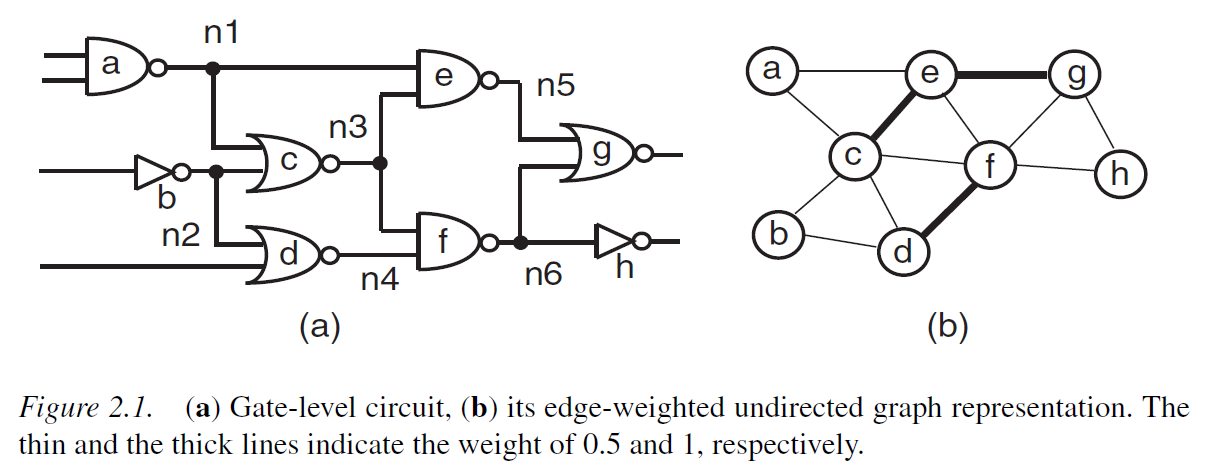
Tabela 4 Neusmeren graf u matričnom obliku

Težine datog grafa su korišćene u određivanju koje kapije će da se zamene particija, pomoću sledeće formule:

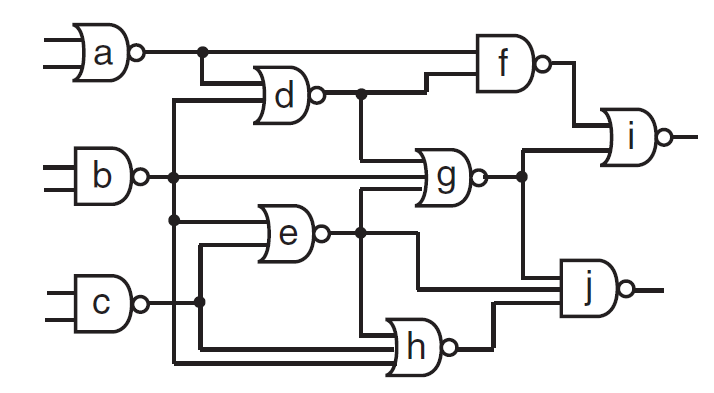


gde važi:



Slika 1 Primer el. kola i odgovarajućeg neusmerenog grafa



Slika 2 Traženo kolo

## Treći zadatak – simulirano kaljenje

**Ulazni podaci** se nalaze u datoteci matrix.in. Sledećeg su oblika:

Širina, visina [10 8]

Broj iteracija, alfa (koeficijent geometrijske progresije za temperaturu), inicijalna temperatura [10 0.9 100]

Matrica incidencije netova i komponenti [tabela 3].

Za ulazne vrednosti su odabrane najbolje vrednosti iz priloženog rada „Partitioning using SA“:

*If we ignore the MaxTime, the best results are obtained for initial temperature To= 100, cooling rate a = 0,9, and M=10.*

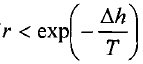
**Izlazni podaci** se ispisuju u konzolu. Oblik izlaznih podataka je identičan kao u drugom zadatku:

oznaka kapije (slovo od a do j – slika 2): broj 1 ili 2 (da li pripada prvoj ili drugoj particiji). Ova linija se ponavlja 10 puta, za svaku kapiju.

Broj netova koji se moraju preseći za takav raspored.

**Rezultati** se razlikuju pri svakom pokretanju. Algoritam uspeva da pronađe rešenje u kojem je presečeno ukupno 3 neta.

Algoritam nasumice odabere dve kapije iz različitih particija. Ukoliko je cena manja od postojeće, novo rešenje se prihvata. Ukoliko nije, rešenje će se prihvatiti ako važi:



gde je r proizvoljan broj između 0 i 1, Δh razlika između postojećeg i najboljeg rešenja, i T trenutna temperatura. Temperatura se menja u svakoj iteraciji geometrijskom progresijom:

