Statička i dinamička particija grafova

Marko Jovanović - RA124-2016
Milan Kresović - RA137-2016

● Uvod

Većina današnjih problema iz kompjuterskih nauka, kao i iz ostalih nauka, može se rešiti primenom teorije grafova. Međutim, da bi se ovakvi problemi rešili, potrebno je poznavati određene predstave grafova i operacije nad istim. U matematičkom smislu, grafovi predstavljaju uređeni par čvorova i ivica. Svaki čvor može, a i ne mora da bude povezan sa drugim čvorom pomoću ivica, koje mogu da budu orjentisane ili neorjentisane. Čvorovi i ivice mogu da imaju određene težine, koje predstavljaju numerčke vrednosti, a graf tada predstavlja težinski graf.

Pored same predstave, jedna od bitnih operacija nad grafovima je i particija grafova. Ona može biti statička ili dinamička, u odnosu na način vršenja podele grafa. Ukoliko se u graf ne dodaju novi elementi ili se ne menja početno stanje, takva particija je statička, dok se graf kod koga se menja početno stanje i vrši reparticija, partitioniše dinamički. Korišćenje particije grafova ima veliku primenu u paralelizaciji procesa u procesoru.

Cilj ove vežbe bio je upoznavanje sa vrstama statičke i dinamičke particije, kao i pisanje jednostavnog algoritma za statičku particiju grafa.

● Metod

Za rešavanje problema statičke particije, koristili su se neorjentisani grafovi sa jediničnim težinama u ivicama. Grafovi su bili predstavljeni pomoću matrica **nxn**, gde je n broj čvorova u grafu. Ivice između dva čvora su bile prikazane jedinicama u presečnom članu vrste i kolone ta dva čvora.

Iako postoji ogroman spektar kompleksnih algoritama za rešavanje ove vrste problema, za potrebe ove vežbe korišćen je KL (**Kernighan-Lin**) algoritam zbog svoje jednostavnosti. U svom najopštijem opisu, ovaj algoritam uzima graf **G(V, E)** sa otežanim ivicama i parnim brojem čvorova i inicijalnu biparticiju **(V1, V2)** gde je kardinalnost particije V1 i V2 jednaka. Jedna od bitnijih osobina ovog algoritma je i funcija cene **(cost function)**. Ova funkcija određuje koja dva čvora iz inicijalne biparticije će zameniti mesta, na osnovu unutrašnje i spoljašnje cene, koje se mogu opisati kao suma svih ivica određenog čvora sa ostalim čvorovima iz svoje particije i suma svih ivica određenog čvora sa ostalim čvorovima iz suprotne particije, respektivno.

Sam algoritam vrši nekoliko restrikcija nad sobom tokom izvršavanja:

1. Broj čvorova u biparticijama treba da ostane približno jednak
2. Funkcija cene treba da bude minimizovana

Restrikcija broj 2 direktno utiče na to da broj grana između dve particije bude što manji, što je jedna od važnijih osobina problema statičke particije. Algoritam je potrebno izvršavati dok više ne bude promena u obe particije.

● Rezultati

U tabeli 1 prikazane su brzine KL algoritma za određen broj čvorova i matrica.

Broj čvorova	Broj ivica	Vreme izvršavanja [s]
10	16	0
50	27	0.001
100	42	0.002
500	112	0.004

Tabela 1. Prikaz odnosa broja čvorova, ivica i vremena izvršavanja

● Diskusija

Iz rezultata se može videti da je ovaj algoritam za današnje standarde spor algoritam. Takođe, zbog nedovoljne optimizacije ograničen je na rad sa do 1000 čvorova.

U cilju poboljšanja ovog algoritma razvijen je FM algoritam (**Fiduccia-Mattheyeses**), koji se oslanja na KL algoritam i može da radi na hipergrafovima. Kako je cilj vežbe bio da se steknu znanja iz teorije grafova i particije grafova, direktno je opisan samo KL algoritam koji radi biparticiju i može biti optimizovan za pravljenje 2^n particija rekurzijom.

Pored statičke particije nad grafovima moguće je raditi i dinamičku particiju. Jedna od glavnih razlika ova dva problema je ta što kod dinamičke particije postoji dodatna restrikcija koja optimizuje algoritam da izvrši što manje promena na već postojećim particijama. Kod ovakvih vrsta algoritama, dva koja su izdvojena u ovoj vežbi su algoritam zasnovan na difuziji i Scratch-Remap algoritam.

Algoritam zasnovan na difuziji, početne, neizbalansirane particije pokušava da repartitioniše tako što minimizuje težinu svake particije pomeranjem čvorova iz teže particije u lakšu, pri tome gledajući da napravi što manji i optimalniji broj pomeraja.

Scratch-Remap algoritam izvršava zasebnu reparticiju nad celokupnim grafom tako što koristi algoritam za statičku particiju, gde tu novu reparticiju koristi kao model od kog inicijalna particija treba što manje da se razlikuje.