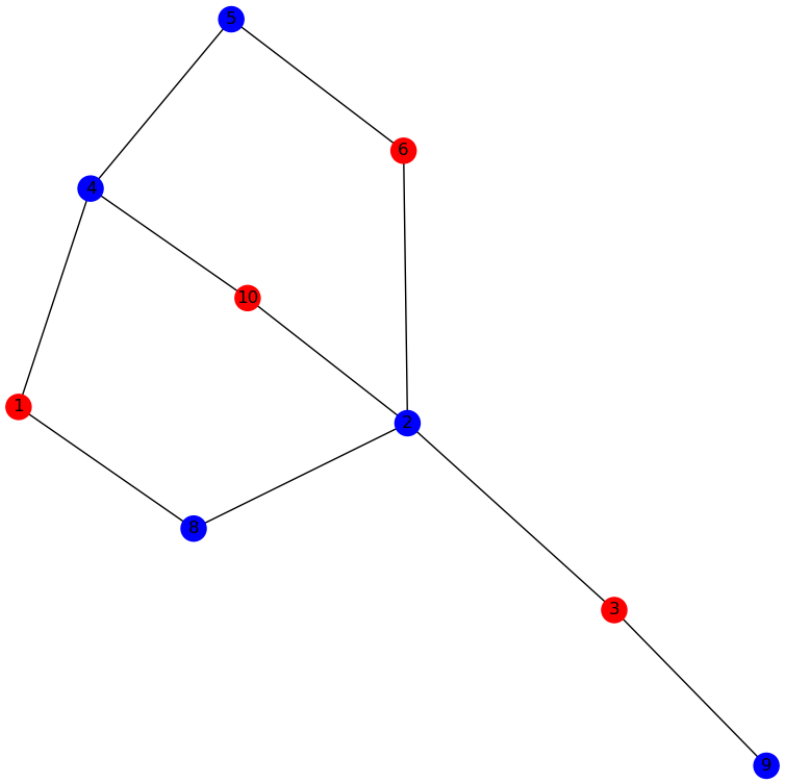
Rešavanje problema maksimalne nezavisne sekvence

Stefan Mirić
Matematički Fakultet, Univerzitet u Beogradu

Januar 2020

1 Uvod

Nezavisna sekvenca za graf G=(V,E) je niz čvorova $v_1, v_2, ..., v_m$ takvih da za svaki v_{i+1} postoji neki susedan čvor u koji nije susedan ni sa jednim čvorom v_j, $j \leq i$.
Takav niz, koji sadrži najveći broj elemenata naziva se **maksimalna nezavisna sekvenca**.
Problem maksimalne nezavisne sekvence podrazumeva traženje nezavisnog niza čvorova u grafu sa najvećom kardinalnošću, odnosno traženje nezavisne sekvence koja je najveća.



Primer grafa u kome je jedno od rešenja niz [10, 3, 6, 1]

Primetimo da su rešenja ovog problema nasledna: naime ako je $v_1, v_2, ..., v_m$ nezavisna sekvenca, onda će i svaka podsekvenca $v_{a_1}, v_{a2}, ..., v_{ax}$ biti nezavisna.

2 Predložena rešenja

Svi pristupi su implementirani u programskom jeziku Python 3 uz biblioteku *Networkx* za kreiranje i internu reprezentaciju grafa, i Matplotlib za prikazivanje grafova, kao i grafikona za testiranje i poredjenje

2.1 Gruba sila - Naivni pristup

Da bismo pronasli rešenje grubom silom, moraćemo da proverimo svaku mogucu permutaciju čvorova. Za graf G=(V,E), jasno je da ce slozenost ovakvog pristupa biti O(|V|!) pa ovakav pristup nije pogodan u realnim okolnostima.Ipak ovakav pristup sigurno daje optimalan rezultat.

Broj grana i ivica	**Prosečno vreme**
8	3s
9	35s
10	7min
11	oko 3 sata

Tabela vremena izvršavanja algoritma grube sile

2.2 Metaheuristički pristup - Simulirano kaljenje

Pristup iterativne optimizacije rešenja simuliranim kaljenjem izgleda primenljivo pri rešavanju ovakvog problema. Kako je čvoru potreban barem jedan susedan čvor koji nije susedan sa dosadašnjim članovima sekvence da bi se razmatrao kao član rešenja, čvorovi manjeg stepena imaju male šanse da uđu kao kasniji članovi sekvence, pa je razmatrano sortiranje u rastućem poretku pre početka optimizacije, međutim, nije bilo prevelikog odstupanja od običnog pristupa. Bolji rezultati su ipak primećeni pri kaljenju koje brže konvergira što govori da je možda i lokalna pretraga daje dovoljno dobro rešenje. Razmatrano je i drugačije biranje okoline, ali ne i testirano.

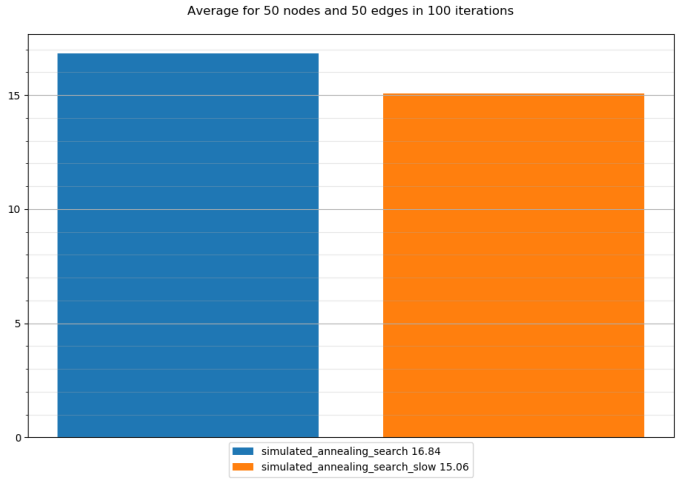
2.3 Metaheuristički pristup - Genetski algoritam

Genetski algoritam znatno sporije izvršava izračunavanje zato sto, za razliku od simuliranog kaljenja, vrednost se izračunava u svakoj iteraciji za svaku jedinku u svojoj populaciji. Vreme se moze ubrzati smanjivanjem populacije kao i smanjivanjem iteracija. Za prilike testiranja, korišćeno je ukrštanje prvog reda, verovatnoća mutacije 1% kao i turnirska selekcija.

3 Testiranje rešenja

Sva testiranja rađena su na računaru sa Windows 7 operativnim sistemom i sa Intel-ovim dvojezgarnim procesorom G2030 sa 3.0 GHz

3.1 Analiza pojedinačnih rešenja



Kaljenje sa funkcijama prihvatanja lošijeg rešenja : a)$\frac{1}{i}$ b)$\frac{1}{\sqrt{i}}$

3.2 t