SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 2129

Pronalaženje epistatskih interakcija pomoću algoritma kolonije mrava

Dino Blažeka

Zagreb, srpanj 2011.

Zahvaljujem se Miletu na stručnom vodstvu, velikom strpljenju i ćevapima.

Zahvaljujem se i Marku Čupiću bez čije volje i predanosti radu sa studentima izrada ovog rada ne bi bila moguća.

Hvala i Igoru Čanadiju zbog preporuke mentora.

SADRŽAJ

1. Uvod

Kroz zadnje desetljeće povećava se broj studija koje se bave pronalaženjem korelacija između jednog ili više genetskih polimorfizama i fenotipskih obilježja. Najčešće promatrano fenotipsko obilježje su bolesti ljudi iz razloga što otkrivanje genetskih polimorfizama koji uzrokuju određenu bolest omogućava vrlo dobro predviđanje rizika od nekih bolesti, pa čak i izliječenje. Korelacije se pronalaze statističkim metodama koje su glavno oruđe kvantitativne genetike – grane genetike koja proučava efekte gena na nivou svojstava fenotipa i zakonitosti koje se u tome pokazuju.

Genetika temeljena na Mendeljevim principima je pružila mnoge odgovore, no geni koji stoje iza kompleksnijih bolesti nisu pronađeni. Mnogi znanstvenici su mišljenja da takve bolesti ne uzrokuje jedan gen, već međudjelovanje više gena. Njihova pretpostavka potvrđena je u nekim istraživanjima koja su utvrdila da su bolesti kao što su rak dojke (Ritchie et al., 2001) ili dijabetes tipa II (Cho et al., 2004) uzrokovane upravo međudjelovanjem više različitih gena na različitim lokusima. Takva pojava naziva se epistatska interakcija ili, kraće, epistaza (engl. *epistasis*). (Cordell, 2002)

Od tog trenutka računarska znanost teži k osmišljavanju učinkovitih algoritama za pronalazak epistatskih interakcija. Neki od njih su metoda kombinatornog dijeljenja (engl. *CPM - combinatorial partitioning method*) (Nelson et al., 2001), metoda ograničenog dijeljenja (engl. *RPM - restricted partitioning method*) (Culverhouse et al., 2004), metoda smanjenja dimenzionalnosti (engl. *MDR - multifactor-dimensionality reduction*) (Greene et al., 2009). Te metode pokazale su se dobrima na relativno malim skupovima podataka, ali su zbog svoje visoke računalne složenosti poprilično neupotrebljive na velikim skupovima podataka. To je iznimno važno iz razloga što je uobičajeni ulazni skup podataka velik te se sastoji do nekoliko tisuća do nekoliko stotina tisuća genetskih polimorfizama prikupljenih nad nekoliko tisuća uzoraka — ljudi za koje se zna da li boluju od bolesti koja je predmet istraživanja.

Ovom NP-teškom problemu pokušalo se doskočiti algoritmom mravlje kolonije (engl. *ACO - ant colony optimization*), koji se prethodno pokazao dobrim u rješavanju NP-teških konstrukcijskih problema. Pokazalo se da on sam nije dovoljan za pronala-

ženje epistatskih interakcija te je kao heuristička spoznaja, iskorištena da vodi mravlji algoritam, upotrijebljena metoda smanjenja dimenzionalnosti, no to je povećalo složenost algoritma i ponovno dovelo do početnog problema — nemogućnosti rada sa velikim ulaznim skupovima podataka.

AntEpiSeeker je algoritam predložen 2009. godine te njegovi autori pokazuju da je efektivniji i efikasniji od gore navedenih metoda. (Wang et al., 2010) Cilj ovog rada je implementirati AntEpiSeeker algoritam, pokušati ga unaprijediti te replicirati rezultate autora.

2. Opis problema

Skup ulaznih podataka sadrži podatke o genomu različitih ljudi od kojih neki imaju svojstvo koje je predmet istraživanja, a neki ne (engl. *test-control study*). To svojstvo je najčešće bolovanje od neke bolesti. Cilj je pronaći dva ili više lokusa, ovisno o zadanim parametrima, za koje se statističkom analizom utvrdi da geni na njima imaju najveći utjecaj na prisustvo ili odsustvo promatranog svojstva, tj. takve lokuse za koje je vjerojatnost da kombinacija gena na njima nema veze sa prisustvom promatranog svojstva, statistički gledano, zanemariva. U svrhu statističke analize koristi se χ^2-test iz razloga što, promatrajući vremensku složenost, nije pretjerano skup, a ujedno nudi pouzdanu informaciju o kvaliteti rješenja. Kao algoritam koji će tražiti takve lokuse koristi se AntEpiSeeker. (Wang et al., 2010)

3. Algoritam kolonije mrava

Mravi su bića koja oduvijek fasciniraju znanstvenike. Njihovo iznimno zanimljivo ponašanje primijećeno je, zasigurno, prije mnogo godina. Fascinantan dio mravljeg ponašanja je to da mravi gotovo uvijek pronalaze najkraći put, a to je fascinantno iz razloga što se zna da mravi nisu sposobni računati niti predviđati udaljenosti, a još su povrh svega i slijepi. Dolazi do zanimljive pojave — svaki mrav sam po sebi ima zanemarivu razinu inteligencije, ali mravinjak, tj. kolonija mrava, se, kao sustav, ponaša iznimno inteligentno. Takva pojava naziva se izranjajuća inteligencija.

Mravlje ponašanje je vrlo zanimljivo znanstvenicima koji se bave računarskom znanosti iz razloga što mravi, tj. njihove kolonije, rješavaju probleme za koje ni računarska znanost ni matematika nisu uspjele osmisliti algoritme koji bi ih rješavali u polinomnoj ili manjoj vremenskoj složenosti. Takvi problemi pripadaju razredu NP-teških problema. Jedan od takvih problema je poznati matematički problem trgovačkog putnika.

Znanstvenici su pokusom sa slike 3.2 utvrdili što to mravlju koloniju čini toliko inteligentnim sustavom. Na sličici A prikazani su mravi kako putuju od mravinjaka (*Nest* na slici) do hrane (*Food* na slici) putem na kojem nema prepreka te bez problema pronalaze optimalan put do hrane. Na sličici B prikazano je stanje netom nakon postavljanja prepreke na put. Prepreka je postavljena tako da obilazak s njezine gornje strane rezultira, u konačnici, kraćim ukupnim putem te, uz pretpostavku da mravi putuju približno jednakom prosječnom brzinom, većom količinom hrane u mravinjaku. Na početku približno jednak broj mrava obilazi prepreku i s gornje i s donje strane što je prikazano na sličici C. S vremenom se sustav stabilizira na način da svi mravi obilaze prepreku s gornje strane, prelazeći kraći put što je i prikazano na sličici D. Još je primijećeno i to da ako se prepreka skrati tako da se duljina obilaska sa donje strane prepreke smanji toliko da put bude kraći nego kod obilaska s gornje strane, a obilazak s gornje strane ostane iste duljine, mravi će i dalje nastaviti obilaziti prepreku gornjim, u tom slučaju duljim, putem. Ovi nalazi naveli su znanstvenike na nekoliko zaključaka:

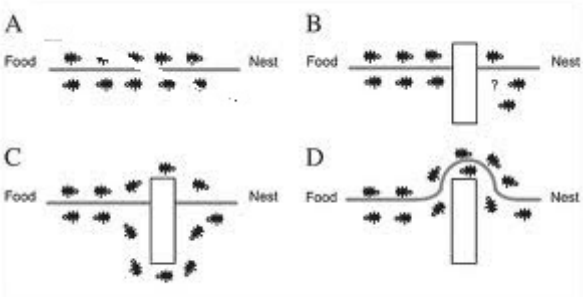
1. mravi pri svom hodu ostavljaju feromonske tragove, te

Slika 3.1: Mravlja "cesta"

2. mravi biraju svoj put nasumično, ali dajući veću šansu onom koji na sebi ima deponiranu veću količinu feromona.

Kasnije je pokazano i to da su feromoni poprilično nestabilni spojevi te relativno brzo isparavaju.

Prvi pokušaji da se ova saznanja pretoče u algoritam i nisu bila previše uspješna. Bili su to iznimno kompliciranisustavi koji su se trudili simulirati mravlju koloniju do najsitnijih detalja što je dovelo i do loše kvalitete rezultata, ali i do složenosti mnogo veće od zadovoljavajuće. To se promijenilo kada je Dorigo u svojem doktorskom radu 1991. predložio pojednostavljenu verziju algoritma. U nju su, po principu Occamove britve, ukomponirana samo ona saznanja koja su kritična za rad algoritma. Taj se algoritam naziva algoritmom mravlje kolonije (engl. *ACO - ant colony optimization*),a njegov pseudokod prikazan je kao algoritam 1.



Slika 3.2: Prikaz pokusa kojim se istražilo ponašanje mrava

Algorithm 1 Pseudokod algoritma kolonije mrava

```
while iteracija < nIteracija do
  for mrav in mravi do
    mrav = stvoriRješenje() {prošeći mrava}
    evaluirajRješenje(mrav)
  end for
  odabraniMravi = odaberiMrave(mravi)
  for mrav in odabraniMravi do
    ažurirajTragove()
  end for
  ispariTragove()
end while
```

Mravlji algoritmi se primarno koriste kod optimizacije problema koji se mogu prikazati grafom te je na njemu potrebno konstruirati stazu. Pri samoj inicijalizaciji algoritma na sve se bridove grafa koji se želi obići nanese ista pretpostavljena vrijednost feromona τ_0. Nakon toga se u svakoj iteraciji svih m mrava prošeće po grafu prema formuli 3.1.

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in N_i^k} \tau_{il}^\alpha}, & \text{ako } j \in N_i^k \\ 0, & \text{ako } j \notin N_i^k \end{cases} \tag{3.1}$$

U formuli p_{ij}^k označava vjerojatnost da će mrav prijeći iz čvora i u čvor j u k-toj iteraciji algoritma. Skup N_i^k ima sljedeću semantiku. U njemu su svi čvorovi, povezani sa čvorom i, koje mrav dotad nije posjetio. Time se izbjegavaju ponavljanja čvorova u stazi mrava (ciklusi) te se osigurava da mrav ne ode u čvor koji nije povezan sa onim u kojem je trenutno. To je osigurano tako da je vjerojatnost da mrav prijeđe iz čvora

i u čvor j koji nije u skupu N_i^k jednaka 0. Ako je prelazak u čvor j legalan, tada se vjerojatnost da mrav ode u taj čvor računa tako da se podijeli iznos deponiranih feromona između čvora i i čvora j sa sumom feromona deponiranih na bridu između čvora i te svih čvorova iz skupa N_i^k – u koji je prijelaz legalan. Brojnik i svaki član sume iz nazivnika se potenciraju koeficijentom α. Njime se može podešavati sam rad algoritma. Važno je, također, primijetiti da je suma vjerojatnosti prema svim legalnim čvorovima iz čvora i jednaka 1, tj. mrav mora prijeći u jedan od tih čvorova, što je i logično. Dodatno, na početku algoritma, nakon što je na sve čvorove deponirana ista količina feromona τ_0, mrav ima jednaku šansu da ode u bilo koji od legalnih čvorova, što je također očekivano s obzirom na prethodni opis ponašanja mrava.

Nakon što je mrav stvorio novi put taj put se evaluira. Mravu se pridružuje određena dobrota (engl. *fitness*) s obzirom na vrijednost funkcije koja se želi optimirati za stvoreni put. Primjerice, ako se rješava problem trgovačkog putnika, tada će dobrota pridružena mravu tipično biti duljina prijeđenog puta.

Sada, kada su svi mravi stvorili svoje puteve koji su i ocijenjeni nekom dobrotom, slijedi postupak biranja podskupa mrava koji će na bridove grafa deponirati feromone. Taj podskup može biti bilo koji podskup skupa svih mrava — može se odabrati n mrava koji su izgradili najbolji put, može se odabrati i stohastičkom metodom na način da se n mrava odabere tako da mravi sa boljim putem imaju veću šansu biti odabrani u taj skup, a u krajnjem slučaju da podskup može biti i čitav skup mrava iz algoritma.

Kad su mravi odabrani vrši se ažuriranje feromona prema formuli 3.2. Ona govori da se vrijednost u feromona na bridu koji povezuje čvorove i i j u $k+1$ iteraciji povećava za $\Delta\tau$. Vrijednost $\Delta\tau$ ovisi o problemu koji se rješava, ali se najčešće pri njegovom računanju koristi dobrota rješenja tako da mrav koji je generirao bolji put deponira više feromona na brodove kojima je prošao od onog čiji je put lošiji. Za problem trgovačkog putnika vrijednost $\Delta\tau$ bi mogla biti primjerice $\frac{1}{L}$ gdje je L duljina puta kojega je mrav generirao.

$$\tau_{ij}^{k+1} = \tau_{ij}^{k} + \Delta\tau \tag{3.2}$$

Nakon provedbe ovih koraka feromoni deponirani na bridovima se isparavaju prema formuli 3.3 u kojoj ρ predstavlja faktor isparavanja. Što je ρ veći to se feromoni deponirani na bridovima više isparavaju.

$$\tau_{ij} = \tau_{ij} \cdot (1 - \rho) \tag{3.3}$$

Ovdje je prikazana samo generička verzija algoritma kolonije mrava. Postoji još

mnogo različitih modifikacija tog algoritma pa su tako, primjerice, nastali:

Ant System: mravi u obzir uzimaju i heurističku informaciju, (korišteno u (Greene et al., 2009)),

Min-Max Ant System (MMAS) samo najbolji mrav ažurira feromone, čija je maksimalna i minimalna vrijednost ograničena, itd.

4. AntEpiSeeker algoritam

AntEpiSeeker je algoritam predložen 2009. godine (Wang et al., 2010) s namjerom da se poveća moć detekcije generičkog algoritma mravlje kolonije. Sam algoritam provodi se u dvije faze:

1. algoritam mravlje kolonije (engl. *Ant Colony Optimization*), te
2. algoritam iscrpne pretrage (engl. *Extensive search algorithm*).

Algoritam mravlje kolonije pretražuje lokuse iz ulaznih podataka dovoljne veličine (broj lokusa mora biti veći ili jednak veličini epistatske interakcije koja se traži). To rezultira skupovima lokusa (haplotipima) za čije članove postoji povećana sumnja da sudjeluju u epistatskim interakcijama. Ti skupovi se pronalaze na dva načina nakon što je algoritam mravlje kolonije završio s radom. Prvi je taj da se kao skupovi uzme određen broj najboljih rješenja (puteva) koje su mravi izgenerirali tokom čitavog vremena rada algoritma, a drugi je taj da se kao skup uzme podskup svih lokusa takav da se u njega umetnu oni lokusi prema kojima su mravi deponirali najviše feromonskih tragova tokom rada algoritma. Svi mogući podskupovi, koji su duljine tražene epistatske interakcije, dobivenih skupova se zatim pretražuju algoritmom iscrpne pretrage.

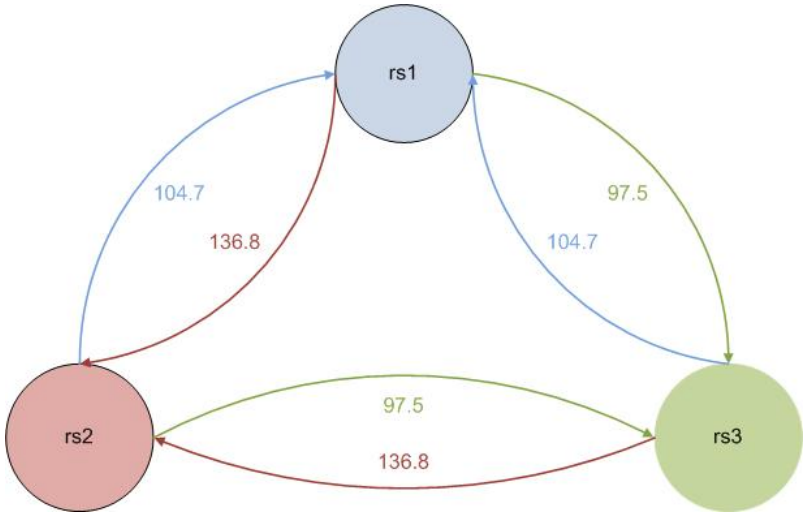
Algoritam se provodi u dva kruga na način da se u prvom krugu algoritmom mravlje kolonije traže veći haplotipovi (skupovi lokusa), tj. putevi kojima prolaze mravi sadrže više čvorova te se nakon toga skupovi za koje postoji povećana sumnja da sudjeluju u epistatskim interakcijama analiziraju algoritmom iscrpne pretrage. U drugom krugu ponavlja se prethodno opisani postupak uz razliku da se traže manji haplotipovi. Iz gore navedenih razloga, prvi krug pretrage osjetljiv je na jače signale, a drugi na slabije. Nakon izvođenja oba kruga minizira se prisustvo lažnih pozitiva u rezultatima.

Algorithm 2 Pseudokod AntEpiSeeker algoritma

```
for veličinaHaplotipa in (velikiHaplotip, maliHaplotip) do
  if veličinaHaplotipa = velikiHaplotip then
    nIteracija = nIteracijaZaVelikiHaplotip
  else
    nIteracija = nIteracijaZaMaliHaplotip
  end if
  postaviFeromonskeTragove(τ0)
  iteracija = 0
  while iteracija < nIteracija do
    for mrav in mravi do
      mrav = stvoriRješenje() {prošeći mrava}
      evaluirajRješenje(mrav)
      provjeriSpadaLiMravMeduNajbolje()
    end for
    for mrav in mravi do
      ažurirajTragove()
    end for
    ispariTragove()
    iteracija++
  end while
  rezultati.dodaj(provediIscrpnuPretragu(najboljiMravi, najviseDeponiranihFero-
  mona))
end for
minimizirajLažnePozitive(rezultati)
vratiRezultate()
```

4.1. Algoritam mravlje kolonije u algoritmu AntEpiSeeker

U algoritmu kolonije mrava AntEpiSeeker-a mravi se kreću usmjerenim, potpuno povezanim grafom u kojem čvorovi predstavljaju lokuse. Iz razloga što je graf potpuno povezan mravi mogu iz bilo kojeg čvora prijeći u bilo koji drugi pod uvjetom da on već nije u stazi. Vrijednosti feromona na svim bridovima prema čvoru i su jednake što znači da vjerojatnost da mrav prijeđe iz čvora i u čvor j ne ovisi o čvoru i. Graf je prikazan na slici 4.1. Čvorovi grafa predstavljaju lokuse, a težina usmjerenog brida između A i B predstavlja količinu feromona deponiranu na put od A do B. Mravi po grafu prelaze put koji je jednak duljini haplotipa – skupa genetskih polimorfizama.



Slika 4.1: Razine feromona između lokusa

Za evaluaciju haplotipa izgeneriranog od strane mrava korisi se χ^2-test. Što je veća vrijednost χ^2-testa, to je pronađeni haplotip bolji. Algoritmu se zadaje parametar $nBestKept$ koji govori koliko najboljih dobivenih haplotipova algoritam kroz svoj rad mora pamtiti.

Feromonske tragove osvježavaju svi mravi sa vrijednošću $\Delta\tau$ koja iznosi desetinu vrijednosti χ^2-testa za haplotip koji odgovara stazi mrava.

4.2. Algoritam iscrpne pretrage u algoritmu AntEpiSeeker

Algoritam iscrpne pretrage pretražuje najbolje haplotipove pronađene algoritmom kolonije mrava te lokuse za koje vrijedi da je na bridovima koji vode do njih deponirana najveća količina feromona. Haplotipovi se analiziraju tako da se provede χ^2-test nad svim podskupovima lokusa haplotipa koji imaju veličinu jednaku veličini epistatske interakciju koju želimo pronaći, a u rezultate sprema sve haplotipove veličine željene epistatske interakcije, za koje je vrijednost dobivena χ^2-testom veća od granične, zadane od strane korisnika.

Primjerice, ako se čuvaju 3 najbolja haplotipa duljine 3 i oni su:

1. $\langle 1, 3, 10 \rangle$,
2. $\langle 8, 2, 14 \rangle$, te
3. $\langle 12, 3, 7 \rangle$,

a duljina tražene epistatske interakcije je 2, tada će algoritam pretražiti sve moguće parove genetskih polimorfizama u svakom haplotipu. Tako će se za prvi haplotip evaluirati parovi:

1. $\langle 1, 3 \rangle$,
2. $\langle 1, 10 \rangle$, te
3. $\langle 3, 10 \rangle$.

Pretpostavimo da se pretražuju 4 lokusa prema kojima je deponiran najjači feromonski trag, a oni su: 1, 2, 3 i 4. Tada će algoritam evaluirati i sve parove iz tog skupa. Općenitije se može reći da će algoritam evaluirati sve podskupove duljine tražene epistatske interakcije. Nakon evaluacije, u slučaju da je rezultat evaluacije ($\chi^2 - testa$) veći od graničnog, podskup se sprema u rezultate.

4.3. Minimizacija lažnih pozitiva

Minimizacija lažnih pozitiva odvija se na samom kraju algoritma na rezultatima prikupljenim iscrpnim pretragama. Algoritam se može opisati u dva jednostavna koraka:

1. Skup $rezultati$ sadrži sve dobivene rezultate, a skup $minimizirano$ je prazan i u njega će se spremati minimizirani rezultati

2. Svaki haplotip h iz seta $rezultati$ se pokuša dodati u skup $minimizirano$. Ako haplotip h ne sadrži nijedan lokus koji se već nalazi u nekom haplotipu iz skupa $minimizirano$, haplotip h se dodaje u skup. Ako u $minimizirano$ već postoji haplotip takav da se preklapa sa h onda se u $minimizirano$ dodaje (ili ostavlja) onaj haplotip za kojeg je vrijednost χ^2-test veća.

5. Implementacija AntEpiSeeker algoritma

Gore opisani algoritam implementiran je u programskom jeziku *Java*. Dijagram razreda prikazan je na slici 5.1. Na njemu nisu prikazane serijske inačice algoritma kolonije mrava te algoritma iscrpne pretrage koje su također implementirane, a služile su za usporedbu vremena izvođenja serijske i paralelne inačice algoritma, tj. mjerenje ubrzanja. Zadaci razreda i sučelja su sljedeći:

AntEpiSeeker: upravlja čitavim algoritmom,

Data Parser: parsira ulaznu datoteku,

Data: modelira ulazne podatke,

Sample: modelira jedan uzorak,

ChiSquaredTester: vrši χ^2-test nad rješenjima,

Solution: modelira jedno rješenje,

IACOAlg: sučelje kojim je određen mravlji algoritam,

IESAAlg: sučelje kojim je određen algoritam iscrpne pretrage,

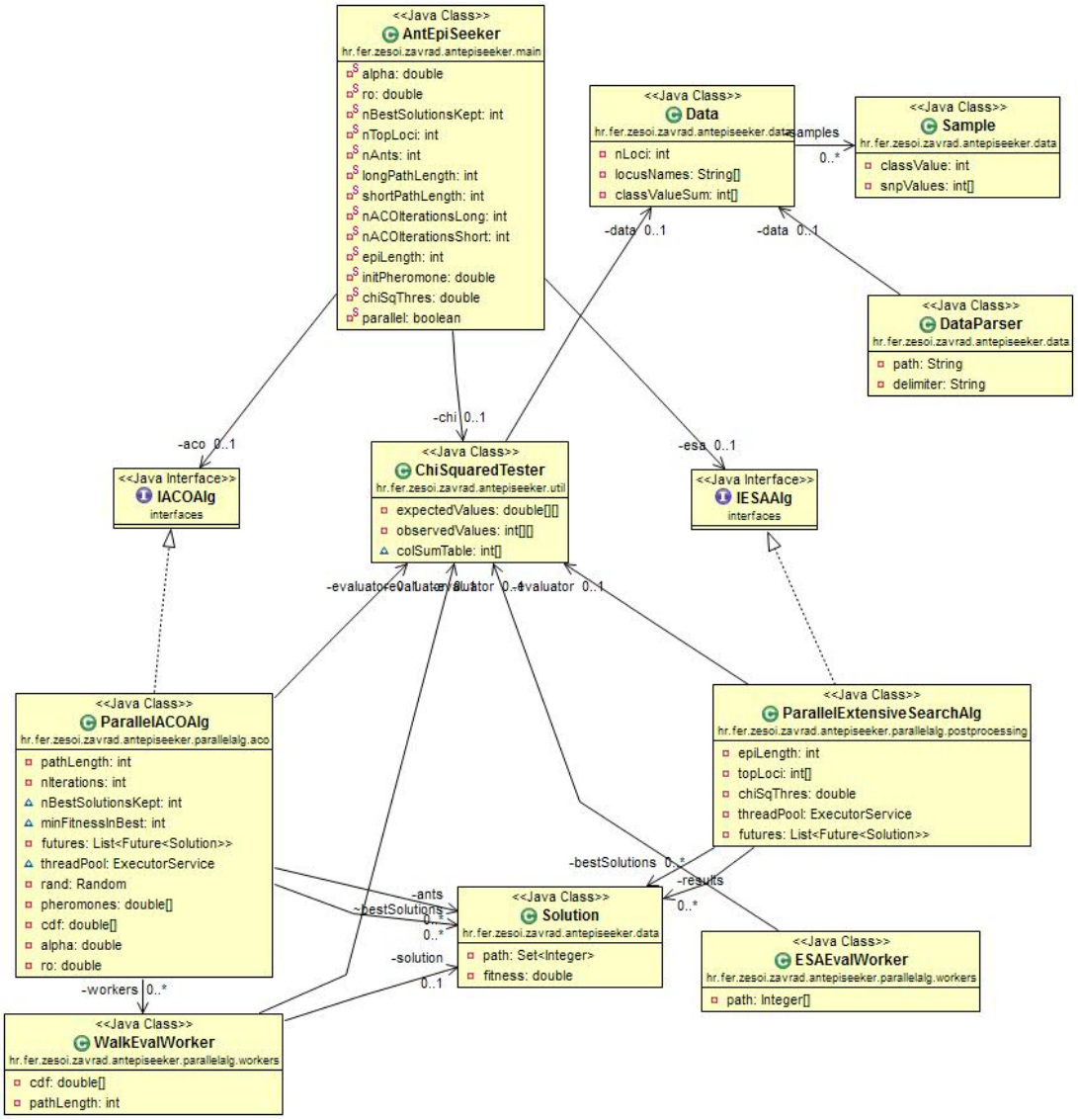
ParallelACOAlg: paralelna inačica algoritma kolonije mrava,

ParallelExtensiveSearchAlg: paralelna inačica algoritma iscrpne pretrage,

WalkEvalWorker: dio algoritma kolonije mrava koji je paraleliziran,

ESAEvalWorker: dio algoritma iscrpne pretrage koji je paraleliziran.

Na slici 5.3 prikazan je dijagram toka podataka za algoritam. Na početku se ulazni podaci parsiraju pomoću *DataParser-a* a potom se, obrađeni, proslijeđuju *ChiSquaredTester-u*. Algoritmi kojima je potreba evaluacija rješenja dobivaju referencu na *ChiSquaredTester* te ga tako mogu koristiti. Nakon svega rezultati se minimiziraju te se spremaju u datoteku.



Slika 5.1: Dijagram razreda

5.1. Ulazna datoteka

Ulazna datoteka u prvom retku sadrži identifikatore lokusa odvojene zarezima te na zadnjem mjestu identifikator *class* koji označava stupac u kojem je zapisan podatak

o prisustvu promatranog svojstva. U narednim retcima slijede podaci o genima na lokusima te podaci o prisustvu promatranog svojstva. Na slici 5.2 dan je primjer ulazne datoteke. Broj lokusa koji se analiziraju je 11, a njihovi identifikatori su u rasponu od *rs0* do *rs10*. Svaki redak, osim prvoga koji sadrži identifikatore, predstavlja jedan uzorak. Prvi uzorak je pozitivan na promatrano svojstvo (engl. *test*) što se vidi po tome što je vrijednost pridružena identifikatoru *class* 1, a drugi je negativan (engl. *control*) pa mu je vrijednost pridružena identifikatoru *class* 0. Geni na lokusima kodirani su brojevima 0, 1 i 2.

```
rs0 , rs1 , rs2 , rs3 , rs4 , rs5 , rs6 , rs7 , rs8 , rs9 , rs10 , c l a s s
2 ,2 ,2 ,2 ,1 ,0 ,1 ,2 ,2 ,0 ,2 ,1
0 ,1 ,1 ,2 ,1 ,2 ,1 ,2 ,1 ,2 ,1 ,0
```

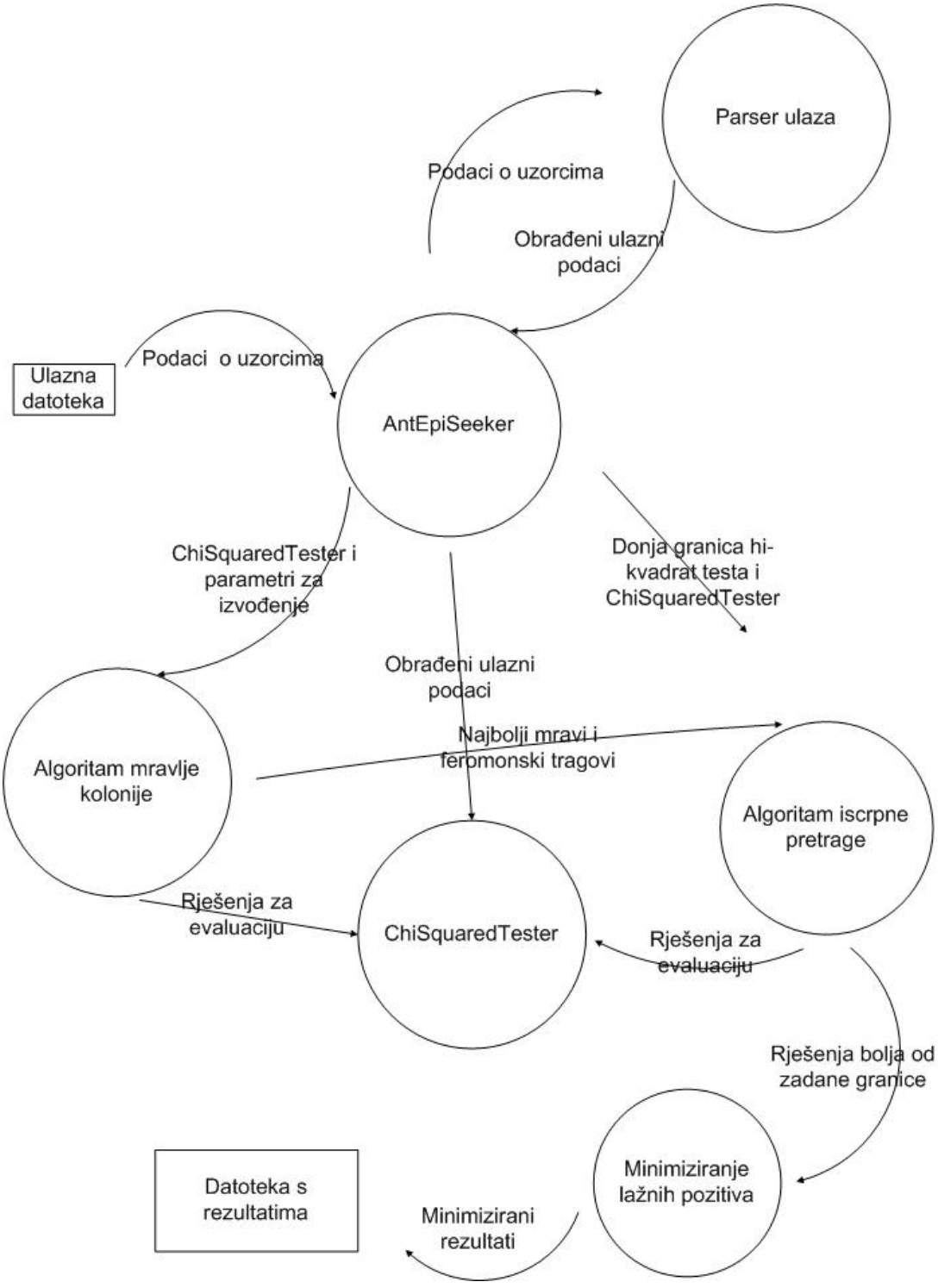
Slika 5.2: Primjer ulazne datoteke

5.2. Paralelizacija algoritma

Dijelovi algoritma AntEpiSeeker su paralelizirani, a to su:

1. šetnja mrava te evaluacija puta kod algoritma mravlje kolonije, i
2. evaluacija rješenja kod iscrpne pretrage.

U tu svrhu korišteni su mehanizmi ugrađeni u programski jezik *Javu*. Kao bazen s dretvama (engl. *thread pool*) koristi se *ExecutorsService* koji je u inicijalizaciji algoritma podešen tako da se automatski prilagođava broju dostupnih jezgri. To je iznimno zahvalno iz razloga što ne zahtijeva nikakve preinake ako se algoritam pokrene na više ili manje jezgri. Paralelizirani dijelovi algoritma oblikovani su radnicima (engl. *workers*) koji implementiraju sučelje *Callable* te se tako mogu proslijediti u red za izvršavanje.



Slika 5.3: Dijagram toka

6. Testiranja i rezultati

Implementacija je testirana podacima koje prilažu autori algoritma (Wang et al., 2010). Podaci sadrže generirane podatke o 2000 lokusa za 2000 uzoraka. Parametri za izvođenje algoritma dani su u tablici 6.1. Ako se parametri mijenjaju kod nekih mjerenja, to je posebno navedeno.

Tablica 6.1: Parametri za AntEpiSeeker

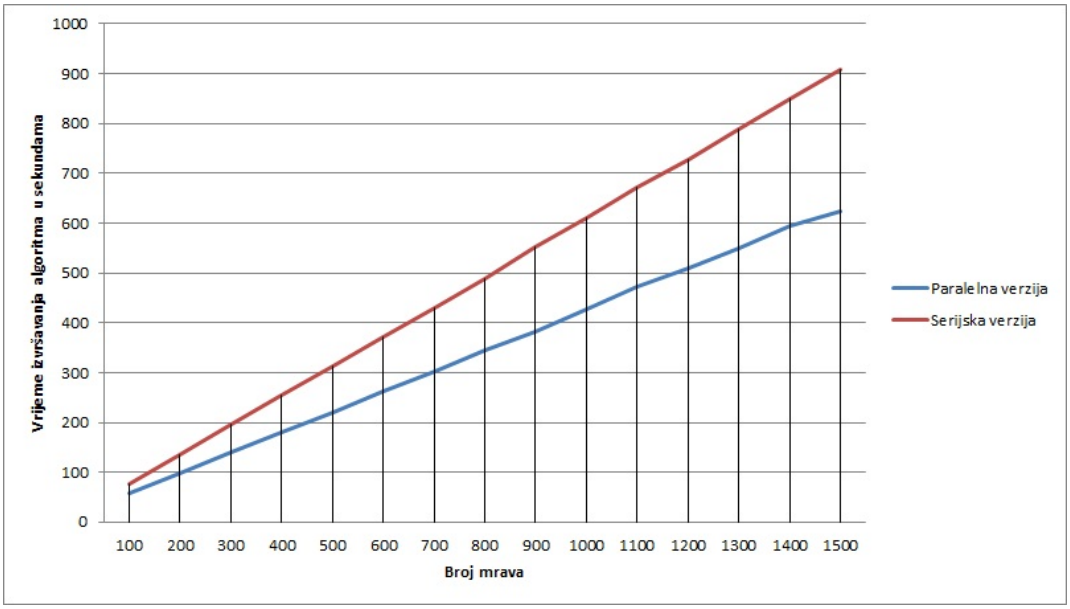
Parametar	Pojašnjenje	Vrijednost
nAnts	Broj mrava	300
ρ	Faktor isparavanja feromona	0.05
τ_0	Početna razina feromona	100
α	Parametar alfa mravljeg algoritma	1
nBestKept	Najbolji mravi koje ACO čuva	1000
nTopLoci	Najjači tragovi koji se pretražuju	200
longPathLength	Duljina dužeg haplotipa (1. ACO)	6
shortPathLength	Duljina kraćeg haplotipa (2. ACO)	3
nIterationsLong	Iteracije s dužim haplotipom	150
nIterationsShort	Iteracije s kraćim haplotipom	300
epiLength	Duljina epistatske interakcije koja se traži kao rezultat	2
chiSqThres	Granica vrijednosti χ^2-testa iznad koje su rješenja	20.09

6.1. Testovi brzine izvođenja

Svi testovi su izvedeni na sljedećem sklopovlju:

- $IntelCore^{TM}2DuoCPUP6400s2.13GHzx2$
- $4GBradnememorije$

Graf na slici 6.1 prikazuje ovisnost brzine izvođenja algoritma Antepiseeker o broju mrava koji sudjeluju unutar njega u algoritmu kolonije mrava. Na grafu se lijepo vidi



Slika 6.1: Prikaz brzine izvođenja u ovisnosti o broju mrava

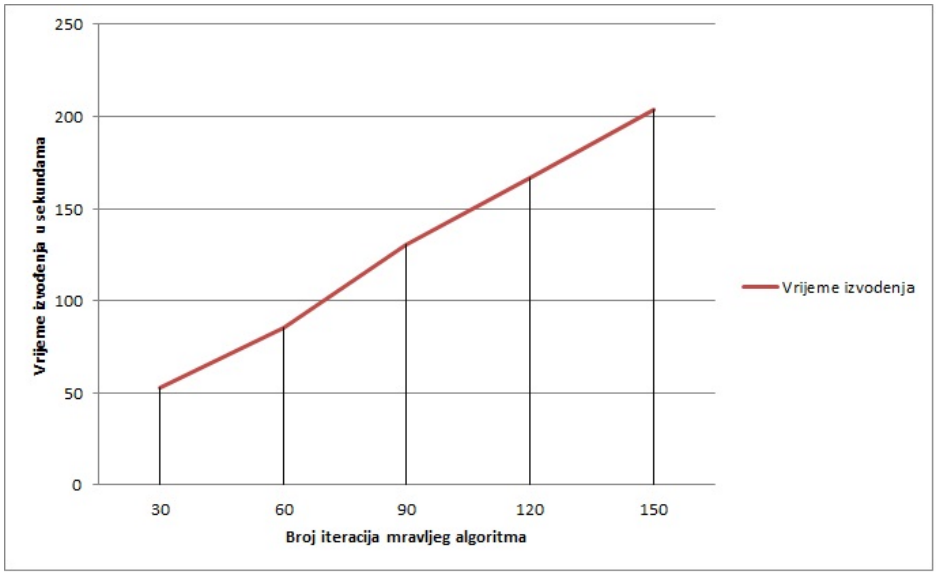
da vrijeme izvođenja algoritma linearno ovisi o broju mrava, ali i to da je ubrzanje paralelne verzije u odnosu na serijsku otprilike 30% što je zadovoljavajući rezultat s obzirom da je algoritam izvršavan na dvojezgrenom procesoru te je tako idealno ubrzanje 50%. Takvo ubrzanje je nemoguće postići iz razloga što su samo dijelovi algoritma pogodni za paralelizaciju te stoga što postoji određena količina *overhead-a*.

Graf na slici 6.2 prikazuje ovisnost brzine izvođenja algoritma o broju iteracija kroz koje se provodi mravlji algoritam. I ovdje se također može uočiti linearna ovisnost. Linearne ovisnosti su očekivane iz razloga što je asimptotska složenost mravljeg algoritma jednaka $nIteracija * nMrava$. Važno je napomenuti da broj iteracija na slici predstavlja broj iteracija mravljeg algoritma za veći haplotip. Za manji haplotip je broj iteracija dvostruko veći.

6.2. Kvaliteta rezultata s obzirom na parametre

6.2.1. Broj iteracija i rezultati

U tablici 6.2 prikazane su vrijednosti χ^2-testa za 5 najboljih rezultata ovisno o broju iteracija. Može se primijetiti da su, očekivano, s porastom broja iteracija i rezultati bolje. Zanimljiva stvar dogodila se kod izvođenja s 60 iteracija. Tokom tog izvođenja algoritam nije uspio pronaći najbolje rješenje (44.8233). To je posljedica stohastičke prirode mravljeg algoritma. Važno je primijetiti da su kod tog izvođenja 2. i 3. rezultat



Slika 6.2: Prikaz brzine izvođenja u ovisnosti o broju iteracija mravljeg algoritma

Tablica 6.2: Ovisnost rezultata o broju iteracija

R.br./Iteracije	30	60	90	120	150
1.	44.8233	40.6763	44.8233	44.8233	44.8233
2.	39.3469	39.5930	39.3469	40.7811	40.7881
3.	36.8151	39.3469	36.2463	39.3469	39.3469
4.	35.3253	35.3392	35.5611	38.4116	39.1061
5.	34.0919	35.3974	35.3974	36.2463	35.3391

bolji nego kod izvođenja s 90 iteracija. Također je važno napomenuti da broj iteracija na slici predstavlja broj iteracija mravljeg algoritma za veći haplotip. Za manji haplotip je broj iteracija dvostruko veći.

6.2.2. Parametar $nTopLoci$ i rezultati

U tablici 6.3 prikazane su vrijednosti χ^2-testa za 5 najboljih rezultata ovisno o broju lokusa s najjačim feromonskim tragom koji se dalje iscrpno pretražuju. Vidi se da su rezultati gotovo identični. Raslikuju se tek u 5. rezultatu prvog stupca kada se parametar postavljen na 50. Očekivano, veća vrijednost parametra znači bolje rezultate. Demonstrirana je i velika moć detekcije algoritma jer već kod niske vrijednosti parametra pronalazi rješenja zadovoljavajuće kvalitete.

Tablica 6.3: Ovisnost rezultata o parametru $nTopLoci$

R.br./nTopLoci	50	100	150	200
1.	44.8233	44.8233	44.8233	44.8233
2.	41.5869	41.5869	41.5869	41.5869
3.	40.7881	40.7881	40.7881	40.7881
4.	39.3469	39.3469	39.3469	39.3469
5.	36.2463	39.1061	39.1061	39.1061

6.2.3. Parametar $nBestKept$ i rezultati

Tablica 6.4: Ovisnost rezultata o parametru $nBestKept$

R.br./nBestKept	200	400	600	800
1.	44.8233	44.8233	44.8233	44.8233
2.	39.3469	39.5930	40.7881	41.5869
3.	35.5611	39.3469	39.3469	40.7881
4.	35.3392	39.1061	39.1061	39.3469
5.	34.4133	36.8151	36.8151	39.1061

U tablici 6.4 prikazane su vrijednosti χ^2-testa za 5 najboljih rezultata ovisno o broju najboljih haplotipova generiranih algoritmom mravlje kolonije koji se potom iscrpno pretražuju. Rezultati su očekivani. Sa povećanjem parametra, povećava se i kvaliteta rezultata. Treba primijetiti i to da je algoritam našao najbolje rješenje sa svim vrijednostima parametra što je još jedan dokaz robusnosti. Ta robusnost algoritma posljedica je toga što je na algoritam kolonije mrava, koji je stohastički po prirodi, nadoveza deterministički algoritam iscrpne pretrage.

7. Zaključak

U radu je opisan algoritam AntEpiSeeker, njegova implementacija u programskom jeziku *Java* te način na koji je implementirana njegova paralelna inačica. Testirana ovisnost vremenskih svojstava algoritma o broju iteracija i broju mrava. Također, testirana je i ovisnost kvalitete rezultata o broju iteracija, broju najboljih haplotipova koje se prenose u algoritam iscrpne pretrage te broj lokusa koji se iscrpno pretražuju zbog deponiranih feromona. Paralelizacijom je postignuto ubrzanje od otprilike 30% na dvojezgrenom stroju, što je zadovoljavajući rezultat. Algoritam se pokazao pouzdanim i robusnim. Uz to daje zadovoljavajuće rezultate nad NP-teškim problemom zadržavajući polinomnu vremensku složenost izvođenja, što mu je zapravo i najveća prednost nad sličnim algoritmima.

LITERATURA

YM Cho, MD Ritchie, JH Moore, JY Park, KU Lee, HD Shin, HK Lee, i KS Park. Multifactor-dimensionality reduction shows a two-locus interaction associated with type 2 diabetes mellitus. *Diabetologia*, 47:549–554, 2004. doi: 10.1007/s00125-003-1321-3.

HJ Cordell. Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Hum Mol Genet*, 11:2463–2468, 2002. doi: 10.1093/hmg/11.20.2463.

R Culverhouse, T Klein, i W Shannon. Detecting epistatic interactions contributing to quantitative traits. *Genet Epidemiol*, 27:141–152, 2004. doi: 10.1002/gepi.20006.

CS Greene, JM Gilmore, J Kiralis, PC Andrews, i JH Moore. Optimal use of expert knowledge in ant colony optimization for the analysis of epistasis in human disease. *Lect Notes Comput Sci*, 5483/2009:92–103, 2009. doi: full_text.

MR Nelson, SL Kardia, RE Ferrell, i CF Sing. A combinatorial partitioning method to identify multilocus genotypic partitions that predict quantitative trait variation. *Genome Res*, 11:458–470, 2001. doi: 10.1101/gr.172901.

MD Ritchie, LW Hahn, N Roodi, LR Bailey, WD Dupont, FF Parl, i JH Moore. Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. *Am J Hum Genet*, 69:138–147, 2001. doi: 10.1086/321276.

Yupeng Wang, Xinyu Liu, Kelly Robbins, i Romdhane Rekaya. Antepiseeker: detecting epistatic interactions for case-control studies using a two-stage ant colony optimization algorithm. *BMC Research Notes*, 3(1):117, 2010. ISSN 1756-0500. doi: 10.1186/1756-0500-3-117. URL `http://www.biomedcentral.com/1756-0500/3/117`.

Pronalaženje epistatskih interakcija pomoću algoritma kolonije mrava

Sažetak

Pronalaženje epistatskih interakcija u ljudskom genomu predstavlja izazov za računarsku znanost. AntEpiSeeker je algoritam koji pokušava riješiti taj problem. Kombinirajući stohastički algoritam kolonije mrava te deterministički algoritam iscrpne pretrage trudi se obuhvatiti najbolje od oba svijeta – robusnost determinističkih algoritama i polinomijalnu vremensku složenost metaheuristika. Algoritam je implementiran u programskom jeziku *Javi* te je i paraleliziranm, čime je na dvojezgrenom stroju dobiveno ubrzanje od 30%. Istražene su ovisnosti vremena izvođenja algoritma i kvalitete rezultata o parametrima.

Ključne riječi: algoritam mravlje kolonije, AntEpiSeeker, epistaza, Java, paralelizacija

Title

Abstract

Detection of epistatic interaction inside humane genome presents a great challenge to computer science. AntEpiSeeker is an algorithm that is trying to solve this problem. By combining stohastic Ant Colony Optimization and deterministic extensive search it tries to get the best of both worlds – robustness of deterministic algorithms and polynomial complexity of metaheuristics. Algorithm has been implemented in programming language *Java* and has been parallelized. This led to 30% faster execution time. Dependency between execution time and parameters has been tested.

Keywords: Ant Colony Optimization, AntEpiSeeker, epistasis, Java, parallelization