SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 217

EVOLUCIJA ALGORITAMA RASPOREĐIVANJA UPORABOM GENETSKOG PROGRAMIRANJA

Maja Legac

Zagreb, Lipanj, 2011.

1. Uvod

U ovom radu se obrađuje problem raspoređivanja uporabom genetskog programiranja. Kao vrsta genetskog programiranja odabrano je gramatičko genetsko programiranje (eng. *Gene expression programming*, GEP). GEP je u predložio Candida Ferreria u 1999. godini, kao prirodni nastavak genetskih algoritama i genetskog programiranja. GEP – om je postignut velik napredak koji leži u mogućnosti konstrukcije kromosoma sposobnog za prikazivanje bilo kojeg gramatičkog stabla (eng. Expression Tree, ET). Za čitanje i prikazivanje informacija GEP kromosoma stvoren je novi jezik, Karva jezik.

Nadalje, kromosomi su napravljeni tako da njihova struktura dozvoljava stvaranje višestrukih gena, gdje je svaki kodiran u posebno gramatičko stablo. Svaki gen se sastoji od glave i repa. Upravo ta strukturalna i funkcijska organizacija gena jamči produkciju ispravnih jedinki, bez obzira koje i koliko velike promjene se dogode na kromosomu.

GEP je baš poput genetskih algoritama i genetskog programiranja, genetski algoritam zato jer koristi populaciju jedinki, odabire jedinke ovisno o njihovoj dobroti i uvodi genetske varijacije uporabom jednog ili više genetskih operatora. Temeljna razlika između ove tri metode nalazi se u načinu kodiranja jedinki. Kod genetskih algoritama jedinke su simbolički nizovi znakova (eng. *string*) fiksne duljine. Kod genetskog programiranja jedinke su nelinearni entiteti različitih veličina i oblika, tzv. stabla parsiranja (eng. *parse trees*). Jedinke kod GEP - a su kodirane kao znakovni nizovi fiksne duljine koji su onda izraženi kao nelinearni entiteti različitih duljina i oblika, tzv. gramatička stabla (eng. *expression trees*).

U radu se razmatra problem raspoređivanja *n* poslova koji tijekom vremena dolaze na stroj s ciljem optimizacije jednog ili više kriterija. Problemi koji se rješavaju su dinamički poslovi raspoređivanja na jednom stroju s poznatim trenutcima pripravnosti. U ovom slučaju GEP se koristi kao heuristička pretraga prostora rješenja pravila raspoređivanja. Pravilo raspoređivanja je matematička formula koja može biti kodirana u kromosom GEP - a koji se uobičajeno sastoji od jednog ili više gena.

Rad se sastoji od pet poglavlja. U sljedećem poglavlju, drugom, dan je opis GEP – a. U trećem poglavlju su opisani kriteriji minimizacije, te je dan strukturni pregled programskog rješenja. U četvrtom poglavlju dan je pregled rezultata mjerenja i posljednje, peto poglavlje je zaključak.

2. Genetsko gramatičko programiranje

2.1. Uvod

GEP je veoma jednostavan. Postoje dva glavna elementa : kromosom i gramatičko stablo. Gramatička stabla su interpretacija genetske informacije iz kromosoma. Kao i u prirodi, proces dekodiranja informacija se naziva prevođenje (eng. *translation*). Prevođenje podrazumijeva da postoji kôd i skup pravila za taj kôd. Genetski kôd je jednostavan. To je jedan - na - jedan veza između simbola kromosoma i funkcija ili završnih znakova koje predstavljaju. Pravila su također jednostavna. Ona određuju prostornu organizaciju funkcija i završnih znakova u gramatičkom stablu i vrstu interakcije između više gramatičkih podstabala (eng. *sub - ETs*).

U GEP - u stoga postoje dva jezika, jezik gena i jezik gramatičkih stabala. Poznavajući redoslijed ili strukturu jednog, ekvivalentno je poznavanju oba. U prirodi, bez obzira na to što je moguće zaključiti redoslijed proteina na temelju redoslijeda gena i obrnuto, još uvijek ne znamo mnogo o pravilima koja određuju trodimenzionalnu strukturu proteina. Ali u GEP - u, zahvaljujući jednostavnim pravilima koji određuju strukturu gramatičkih stabala i njihovog međudjelovanja, moguće je odmah zaključiti o fenotipu ako je poznat redoslijed gena i obrnuto. Taj dvojezičan i nedvosmislen sustav naziva se *Karva* jezik.

2.2. Kromosom

U GEP - u kromosom (genom) se sastoji od znakovnog niza fiksne duljine koji se sastoji od jednog ili više gena. Unatoč fiksnoj duljini kromosoma vidjeti će se da od GEP kromosoma nastaju gramatička stabala različitih duljina i oblika.

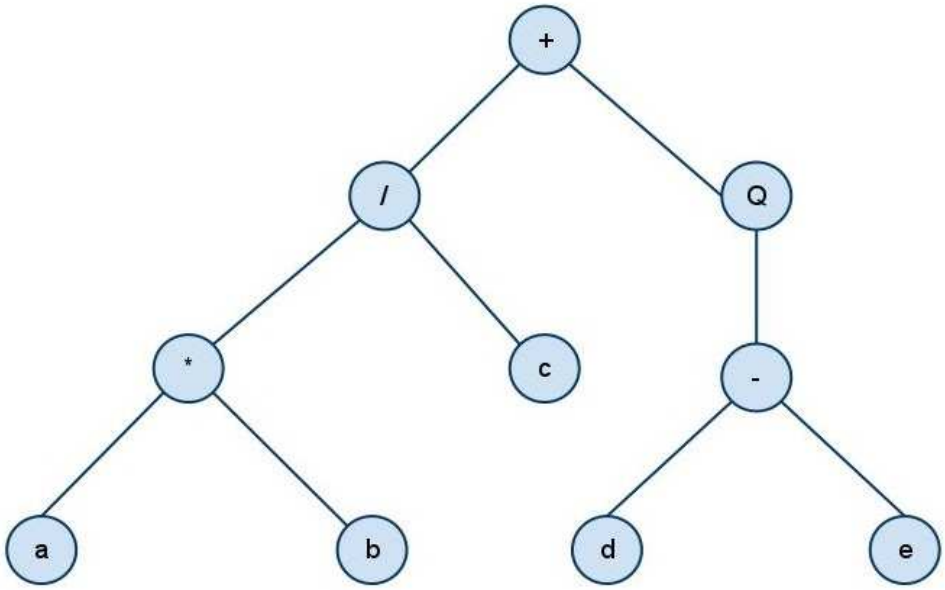
2.2.1. Otvoreni okviri za čitanje i geni

Strukturnu organizaciju gena je lakše razumjeti u terminima otvorenih okvira za čitanje (eng. *open reading frames, ORFs*). U biologiji ORF ili kodirajući niz gena započinje sa početnim kodonom, nastavlja s aminokiselinskim kodonima i završava sa završnim kodonom. Međutim, gen je više od reprezentacije ORF - a. Iako je kod GEP - a početni kodon uvijek na prvom mjestu u genu, završni kodon ne mora uvijek biti na posljednjem mjestu. Posve je uobičajeno za GEP - gene da postoji ne kodirajući niz, jer se ne podudaraju s izrazom. Prikazati ćemo to na sljedećem primjeru.

$$\frac{a \times b}{c} + \sqrt{d - e}$$

Slika 1.Matematička formula

Izraz sa slike 1. također možemo predstaviti kao stablo, koje je prikazano na slici 2.



Slika 2. Gramatičko genetsko stablo

«Q» predstavlja drugi korijen. Ovo gramatičko stablo predstavlja fenotip GEP kromosoma. Genotip se dobije iščitavanjem iz stabla red po red s lijeva na desno.

```
0123456789
+/Q*c-abde
```

Genotip predstavlja ORF koji započinje sa «+», na poziciji nula i završava na poziciji devet sa znakom «e». ORF - ovi se nazivaju K - izrazi od *Karva* jezika.

Da bismo točno izrazili ORF, moramo slijediti pravila koja određuju prostorni raspored funkcija i završnih znakova. Pravila su sljedeća :

- Početak gena odgovara korijenu gramatičkog stabla. Time se dobiva čvor u prvom redu.
- Broj čvorova u drugom retku jednak je zbroju argumenata funkcije svakog čvora iz prethodnog retka. Različite funkcije mogu imati različiti broj argumenata, dok završni znakovi uvijek imaju nula argumenata,
- S lijeva na desno, čvorovi se popunjavaju znakovima u istom redoslijedu kako se nalaze u kromosomu.

- Proces se ponavlja sve dok nije formiran redak koji se sastoji samo od završnih znakova.

Promotrimo sljedeći K – izraz.

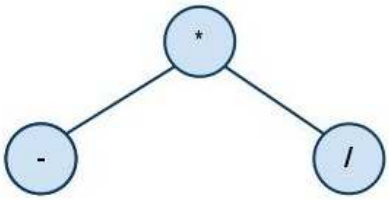
```
012345678901
*-/Qb+b+aaab
```

Prođimo proces stvaranja ET - ija za gornji genotip.

U korijen ET - ija se smješta prvi gen iz K – izraza, kao što je prikazano na slici 3 a).

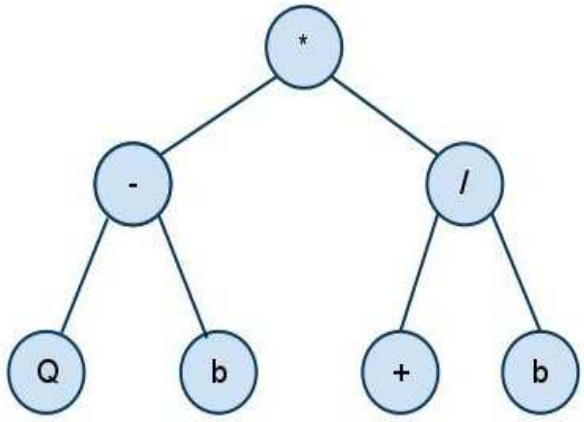
Slika 3 a). Korijen ET - ija

Funkcija množenja prima dva argumenta, stoga će se u sljedećem retku nalaziti dva čvora. U ovom slučaju znakovi s pozicije jedan i dva, kao što je prikazano na slici 3 b).



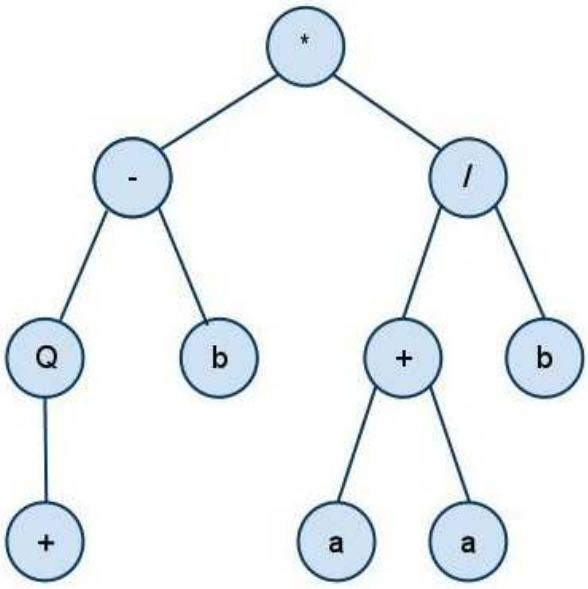
Slika 3 b). ET nakon koraka 2

Oduzimanje i dijeljenje također su funkcije koje imaju dva argumenta. Dakle, u sljedećem retku nalaziti će se četiri čvora, u ovom slučaju znakovi s pozicija tri, četiri, pet i šest. Stablo nakon dodavanja tih čvorova je prikazano na slici 3 c).



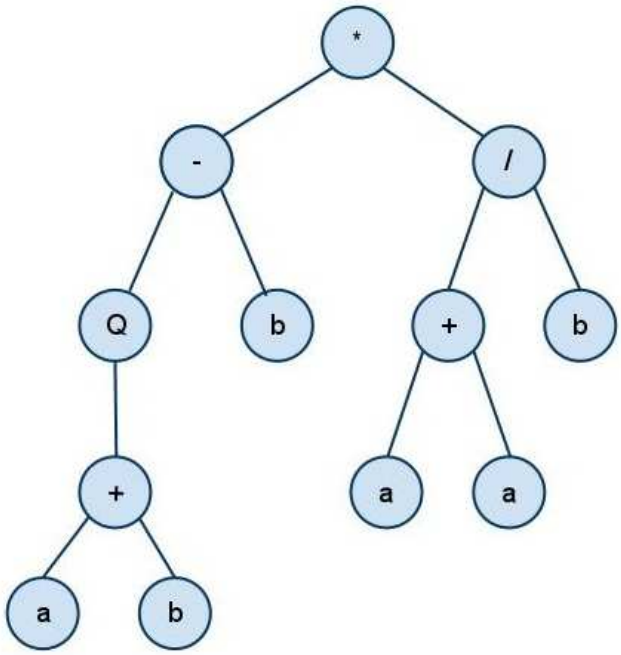
Slika 3 c). ET nakon dodavanja čvorova u trećem koraku

Kao što se može primijetiti u trećem retku nalaze se dvije različite funkcije; prva funkcija, drugi korijen, je funkcija koja prima samo jedan argument. Druga funkcija, zbrajanje, prima dva argumenta. U sljedećem, četvrtom retku nalaziti će se tri nova čvora. Popunjavaju se elementima s pozicija sedam, osam i devet. Izgled gramatičkog stabla je prikazan na slici 3 d).



Slika 3 d). ET nakon četvrtog koraka

U ovom retku iako postoje tri čvora u samo jednom je funkcija. Ponovno, sljedeća dva čvora iz K - izraza smještaju se ispod funkcijskog čvora, čime dobivamo sljedeće gramatičko stablo, prikazano na slici 3 e).



Slika 3 e). ET nakon petog koraka

S ovim korakom gramatičko stablo je u potpunosti formirano zato jer posljednji redak sadrži samo završne znakove. Vidjeti ćemo da zahvaljujući strukturnoj organizaciji GEP gena, posljednji redak gramatičkih stabala uvijek sadržava samo završne znakove. To je ekvivalentno izjavi da su sva gramatička stabla uvijek sintaksno ispravna.

Gledajući samo strukturu GEP ORF - ova, teško je ili čak nemoguće vidjeti prednosti ovakvog prikaza. Međutim, kada se ORF - ovi analiziraju u kontekstu gena, prednosti ovakvog predstavljanja postaju očite. Kao što je rečeno, GEP kromosomi su fiksne duljine, i sastoje se od od jednog ili više gena iste duljine. Stoga, duljina gena je također fiksna. Dakle, ono što varira kod GEP - a nije duljina gena, nego duljina ORF - ova. Duljina ORF - a može biti manja ili jednaka od duljine gena. Iz toga vidimo da se ne moraju uvijek iskoristiti svi znakovi iz kromosoma. U slučaju kada se ne iskoriste svi

znakovi iz kromosoma imamo ne - kodirajuće područje (eng. *non – coding region*).

Koja je uloga tih nekodirajućih područja? Oni su zapravo osnova GEP - ove fleksibilnosti i snage, jer dozvoljavaju modifikacije kromosoma korištenjem bilo kojeg genetskog operatora bez ograničenja, proizvodeći uvijek sintaksno ispavane jedinke bez potrebe za kompliciranim uređivanjem ili vrlo ograničenim načinom implementacije genetskih operatora. Ovo je najveća razlika između GEP - a i prethodnih implementacija GP - a, sa ili bez linearnih kromosoma.

Analizirajmo strukturnu organizaciju GEP gena da bismo razumjeli kako uvijek prikazuju sintaksno ispravne jedinke i zašto dozvoljavaju slobodnu primjenu bilo kojeg genetskog operatora.

2.2.2. GEP geni

GEP geni se sastoje od glave i repa. Glava se može sastojati od funkcijskih i završnih znakova, dok se u repu mogu nalaziti samo završni znakovi (eng. *terminals*). Za svaki problem duljina glave *h* se bira, dok je duljina repa *t* funkcija duljine glave *h* i broja argumenata funkcije iz skupa koja može primiti najviše argumenata *n.* Formula za izračunavanje duljine repa prikazana je na slici 4.

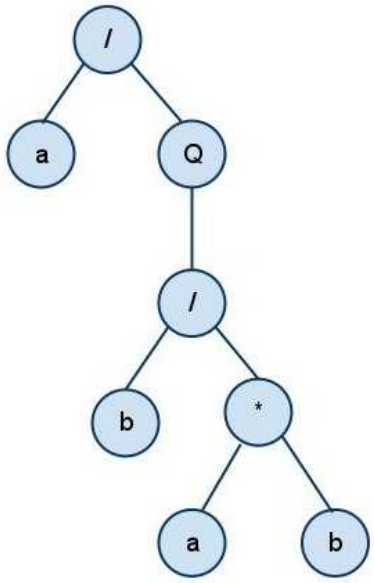
$$t = h(n-1)+1$$

Slika 4. Formula za izračunavanje duljine repa

Uzmimo za primjer gen sa skupom funkcija F = {Q, *, /, -, +} i skupom završnih znakova T = {a, b}. U ovom slučaju, *n = 2,* a ako odaberemo da je *h = 15,* onda je *t = 16,* a ukupna duljina gena je 15 + 16 = 31. Jedan takav gen je prikazan odmah ispod.

```
012345678901234567890123456789
/aQ/b*ab/Qa*b*-abababaababbabbbba
```

Gramatičko stablo ovog K - izraza prikazano je na slici 5.

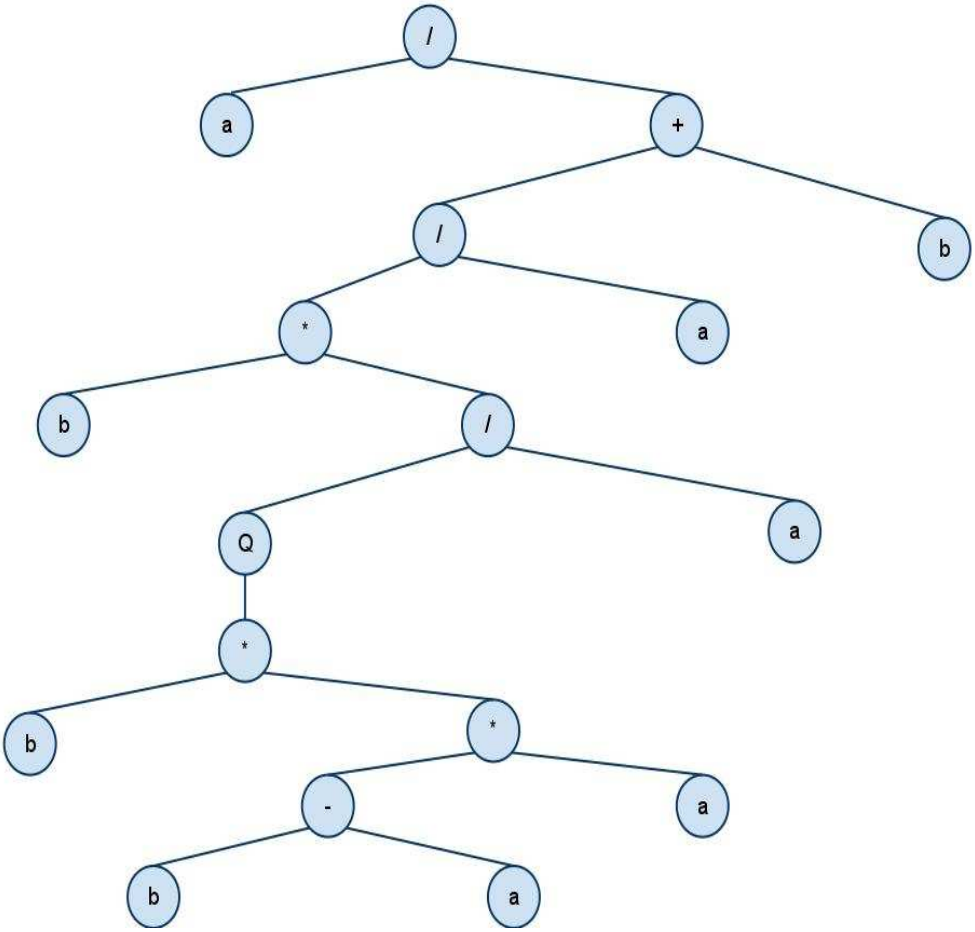


Slika 5. Gramatičko stablo

U ovom slučaju, ORF završava na poziciji sedam, dok gen završava na poziciji trideset. Pretpostavimo da se dogodila mutacija na poziciji dva, mijenjajući «Q» u «+». Tada imamo sljedeći gen.

```
01234567890123456789012345678 90
/a+/b*ab/Qa*b*-abababaababbabbbba
```

Iz tog K - izraza nastaje sljedeće stablo, prikazano na slici 6.



Slika 6. ET nakon promjene znaka na poziciji 2

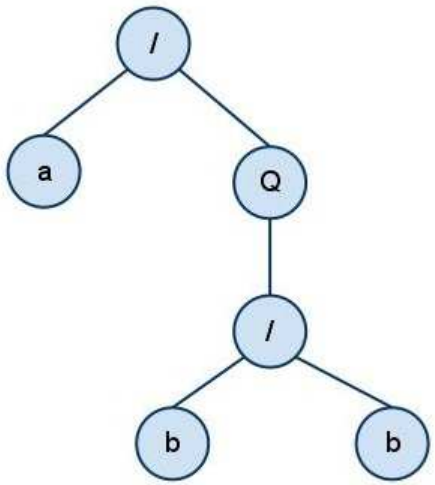
U ovom slučaju pozicija završetka pomaknula se za deset mjesta u desno, na poziciju sedamnaest. Očito, direktno suprotno se također moglo dogoditi. Promotrimo ponovno originalni gen.

```
01234567890123456789012345678 90
/aQ/b*ab/Qa*b*-abababaababbabbbba
```

Pretpostavimo da se dogodila mutacija na poziciji pet, mijenjajući znak «*» u «b». Gen sada izgleda :

```
01234567890123456789012345678 90
/aQ/bbab/Qa*b*-abababaababbabbbba
```

Stablo prikazano na slici 7. dobiva se od gore navedenog K – izraza.



Slika 7. ET nakon mutacije na poziciji 5

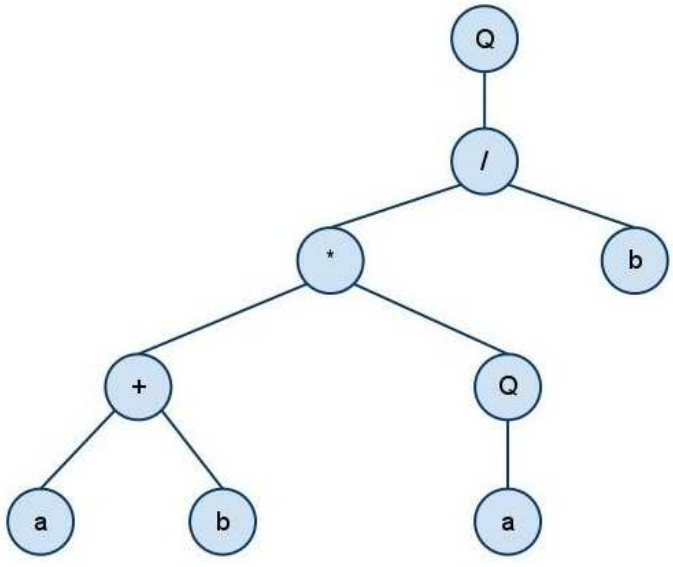
U ovom slučaju ORF završava na poziciji pet, skraćujući stablo za dva čvora. Dakle, unatoč fiksnoj duljini, svaki gen imapotencijal kodirati gramatička stabla različite veličine i oblika, gdje je najjednostavnije gramatičko stablo ono koje se sastoji od samo jednog čvor, slučaj kada je prvi element gena završni znak. Najkompleksnije je pak ono koje se sastoji od svih elemenata gena, kada su svi elementi glave funkcije s maksimalnim brojem argumenata. Iz gornjeg primjera jasno je da bilo koja modifikacija nad genomom uvijek rezultira strukturalno ispravnim gramatičkim stablom. Jedino je važno održavati strukturnu organizaciju gena, zadržavajući uvijek granice između glave i repa i ne dozvoljavajući funkcijskim elementima da se pojave u repu.

2.2.3. Višegenski kromosomi

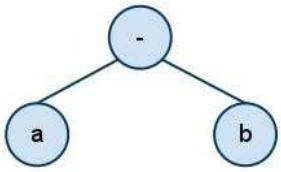
GEP kromosomi se uglavnom sastoje od jednog ili više gena jednake duljine. Za svaki problem, broj gena, kao i duljina glave su *a priori* odabrani. Svaki gen kodira jedno gramatičko podstablo (eng. *sub - ET*), a gramatička podstabla uzajamno djeluju tvoreći složenija gramatička podstabla (eng. *multi - subunit ETs*). Razmotrimo, na primjer, sljedeći kromosom duljine 45, koji se sastoji od tri gena, svaki duljine 15.

```
012345678901234
Q/*b+Qababaabaa
-abQ/*+bababbab
**-*bb/babaaaab
```

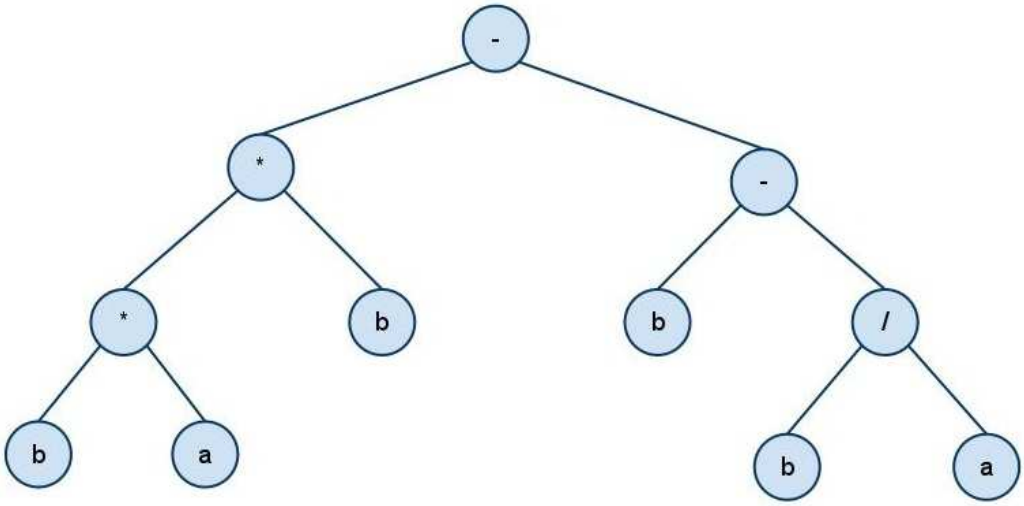
Ovaj kromosom sastoji se od tri ORF – a. Svaki ORF kodira jedno gramatičko podstablo. Pozicija nula označuje početak svakog gena. Kraj svakog ORF - a je poznat tek nakon konstrukcije odgovarajućeg podstabla. Kao što je pokazano na slici 8., prvi ORF završava na poziciji osam, drugi završava na poziciji dva, a treći završava na poziciji deset. Dakle, GEP kromosomi sadrže više ORF - ova, gdje svaki ORF kodira strukturno i funkcionalno jedinstveno gramatičko podstablo. Ovisno o trenutnom problemu, ta gramatička podstabla mogu se pojedinačno i neovisno odabrati ovisno o dobroti ili mogu stvoriti kompleksnije gramatičko podstablo i biti odabrani ovisno o dobroti tog kompleksnijeg gramatičkog podstabla. Važno je imati na umu da je svako gramatičko podstablo jedinstven entitet, ali ujedno i dio veće složenije hijerarhijske strukture, te kao u svim kompleksnim sustavima «*the whole is more than the sum of parts*».



Slika 8. a) Prvi ORF



Slika 8. b) Drugi ORF



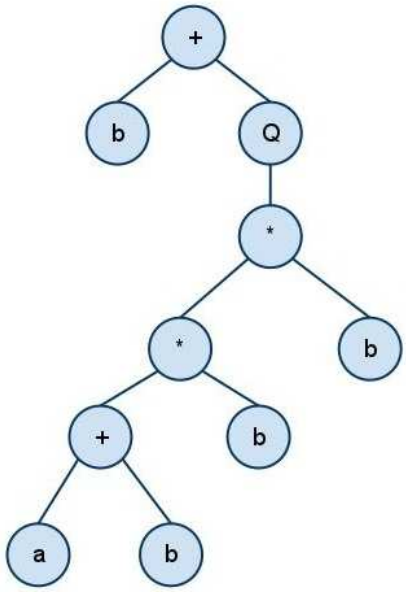
Slika 8. c) Treći ORF

2.2.4. Povezivanje gramatičkih stabala

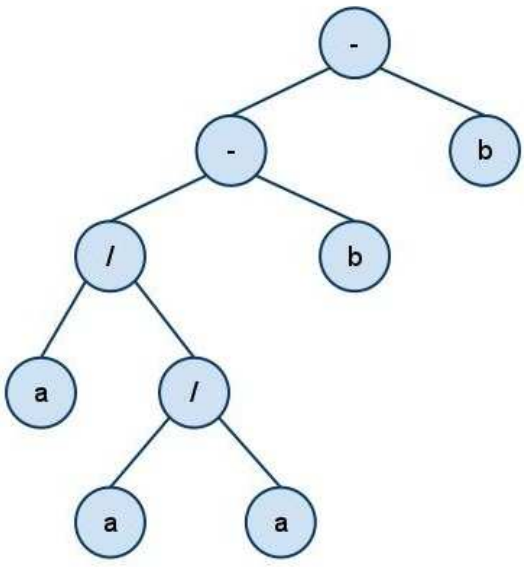
Kod GEP – a izražavanje genetske informacije započinje prevođenjem (eng. *translation*). To je proces prevođenja jedinke od gena do gramatičkog stabla. Već smo vidjeli da prevođenje rezultira formiranjem gramatičkih podstabala (eng. *sub - ETs*) različitih veličina i oblika. U većini slučajeva, za potpuno izražavanje genetske informacije potrebna je interakcija različitih gramatičkih podstabala. Jedna od najjednostavnijih tipova interakcije je povezivanje podstabla nekom funkcijom. Ovaj proces je sličan procesu povezivanja proteina u proteinske lance. Kada su u funkcijskim čvorovima nalaze algebarske ili logičke funkcije tada se bilo koja algebarska ili logička funkcija može se iskoristiti za povezivanje tih gramatičkih podstabla u jedno stablo. Algebarske funkcije koje se najčešće odabiru za povezivanje su zbrajanje i množenje, a od logičkih to je logičko zbrajanje. Na slikama 9. a), 9. b) i 9.c),

pokazana su gramatička stabla niže navedenog kromosoma, a na slici 10. ta gramatička podstabla povezana su funkcijom zbrajanja.

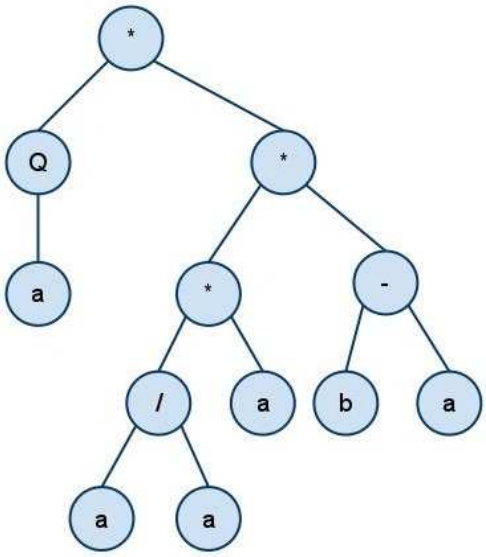
```
012345678901234012345678901234012345678901234
+bQ**b+babababbb–b/ba/aaababab*Q*a*-/abaaaaab
```



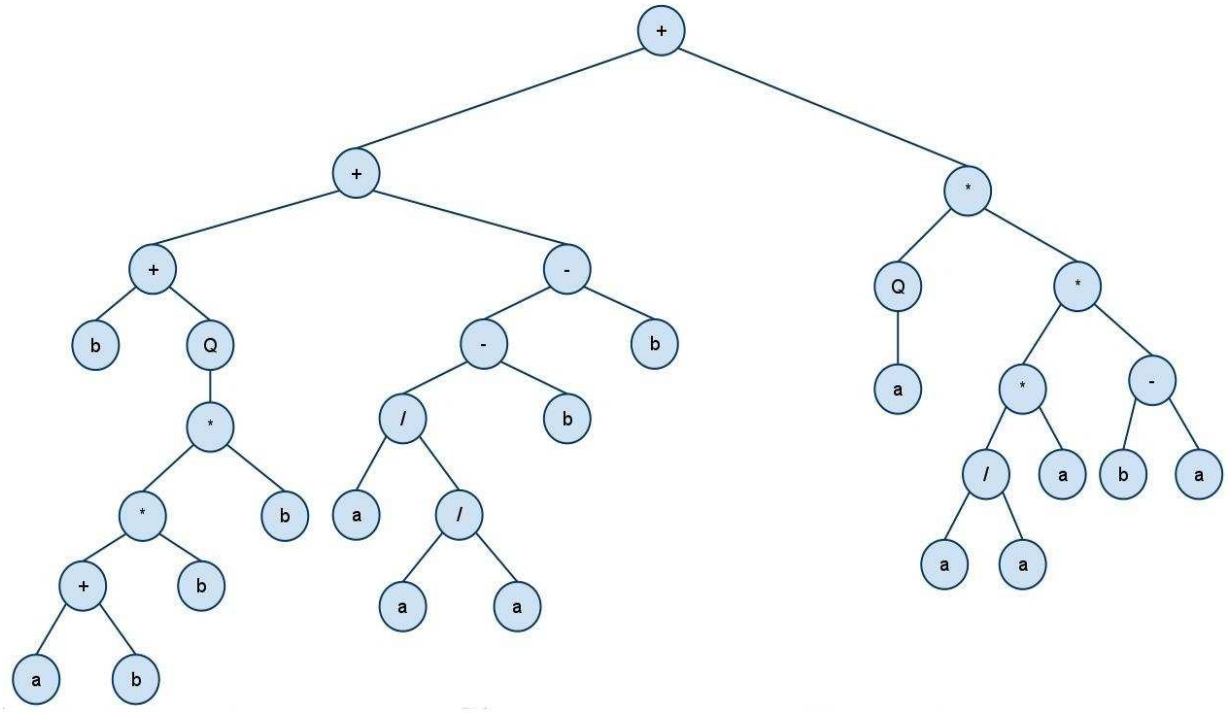
Slika 9. a) Prvo gramatičko pod – stablo



Slika 9. b) Drugo gramatičko pod – stablo



Slika 9. c) Treće gramatičko pod – stablo



Slika 10. Novi ET nastao povezivanjem gramatičkih podstabala funkcijom zbrajanja

Za evoluciju složenih problema, bolje je koristiti višegenske kromosome, zato jer je s njima moguća modularna izgradnja složenih, hijerarhijskih struktura, gdje svaki gen gradi manji blok[1]. Oni su odvojeni, stoga mogu evoluirati individualno.

3. Genetski operatori i evolucija

Genetski operatori čine srž svih genetskih algoritama. Dva su zajednička svim svim evolucijskim sustavima, selekcija i replikacija. Iako su to dva najvažnija operatora, oni sami ne mogu unijeti varijacije u genetski materijal populacije. Uporabom isključivo ta dva operatora kroz svaku generaciju razlika između jedinki bi se smanjivala, dok se na kraju cijela populacija ne bi sastojala od identičnih jedinki. Stoga su potrebni operatori koji unose varijacije u populaciju. Kod GEP - a to su operatori mutacije, rekombinacije i transpozicije. U nastavku ovog poglavlja će biti pokazano kako genetski operatori funkcioniraju i kako se mogu implementirati.

3.1. Selekcija i replikacija

U svim evolucijskim sustavima jedinke se odabiru prema dobroti. Metoda odabira jedinki koja se koristila ovdje naziva se *fitness proportional rulette - wheel*[3] metoda odabira. Najbolja jedinka se klonira u sljedeću generaciju. S obzirom na dobrotu i «sreću» biraju se jedinke za replikaciju. Tijekom replikacije kromosomi se kopiraju u sljedeću generaciju. Što jedinka ima veću dobrotu to ima veće šanse za stvaranje više potomaka. Tijekom replikacije kromosom se kopira toliko puta u sljedeću generaciju koliko je puta bio odabran. Kolo se zavrti toliko puta koliko ima jedinki u populaciji. Na taj način broj jedinki u populaciji je uvijek isti.

3.2. Mutacija

Mutacija se može dogoditi bilo gdje u kromosomu. Međutim, strukturna organizacija kromosoma mora ostati netaknuta. U glavi, bilo koji simbol se može promijeniti u bilo koji drugi simbol. U repu, završni znakovi se mogu promijenit samo u završne znakove. Na taj se način čuva strukturna organizacija i sve mutirane jedinke su sintaksno ispravne. Tipično, koristi se stopa mutacije p_m jednaka *two point mutation* po kromosomu. Stopa mutacije p_m je vjerojatnost da se nad određenim genom iz kromosoma dogodi mutacija. Promotrimo sljedeći kromosom koji se sastoji od tri gena.

```
012345678900123456789001234567890
Q+bb*bbbaba-**--abbbaaQ*a*Qbbbaab
```

Pretpostavimo da je mutacija promijenila znak «*» na poziciji četiri u genu jedan u znak «/», znak «-» na poziciji nula u genu dva u znak «Q» i znak «a» na poziciji dva u genu tri u znak «+». Tada se dobiva sljedeći kromosom :

```
012345678900123456789001234567890
Q+bb/bbbabaQ**--abbbaaQ*+*bbbaab
```

Ako je funkcijski znak mutiran u završni i obrnuto, ili ako se funkcijski znak od dva argumenta promjeni u funkcijski znak jednog argumenta i obrnuto to drastično mijenja izgled gramatičkog stabla. Primijetimo također da mutacija koja se dogodila na genu jedan je neturalna mutacija zato jer se dogodila u nekodirajućem dijelu gena jedan. Važno je za istaknuti da su nekodirajući dijelovi kromosoma savršena mjesta za akumulaciju neutralnih mutacija. Zaključno, u GEP - u nema ograničenja na mutaciju ni prema vrsti mutacije, ni prema broju mutacije. U svim slučajevima novo nastale jedinke su sintaksno ispravne[1].

3.3. Transpozicija

Dijelovi koji se mogu trasponirati su dijelovi koji se mogu uzeti i prebaciti u drugi dio kromosoma. U GEP - u postoje tri vrste elemenata koji se mogu transponirati :

- kratki fragmenti s funkcijom ili završnim znakom na prvoj poziciji koji se transponira u glavu gena, ne u korijen;
- kratki fragmenti s funkcijiskim znakom na prvoj pozicji koji se transponiraju u korijen kromosoma;
- cijeli gen se transponira na početak kromosoma.

3.3.1. Transponiranje u glavu gena

Bilo koji podniz u kromosomu se može transponirati u glavu, to je tzv. podniz za umetanje (eng. *insertion sequence, IS*). Ti se elementi metodom slučajnog odabira biraju unutar kromosoma. Kopija elementa za transponiranje se umeće na bilo koju poziciju u glavi, osim na nultu poziciju. Uobičajeno je stopa transpozicije p_{is} 0.1 i koristi se skup od tri IS elementa različitih duljina. Stopa transpozicije p_{is} je vjerojatnost da se izvrši zamjena genetskog materijala između gena slučajno odabranog kromosoma. Operator transpozicije metodom slučajnog odabira odabire kromosome, početak IS elementa, ciljano mjesto u kromosomu i duljinu fragmenta za transponiranje. Promotrimo sljedeći kromosom od dva gena.

```
012345678901234560123456789012345 6
-aba+Q-baabaabaabQ*+*+-/aababbaaaa
```

Pretpostavimo da je odabran podniz «a+Q» u genu jedan metodom slučajnog odabira da postane IS element i transponira se između pozicija 2 - 3 u genu dva. Time se dobiva :

```
012345678901234560123456789012345 6
-aba+Q-baabaabaabQ*+a+Q+ababbaaaa
```

Primijetimo da se u jednu ruku podniz za transponiranje pojavljuje dva puta u kromosomu, ali u drugu ruku, niz s jednakim brojem znakova kao i IS element je bio izbrisan na kraju glave u ciljnom genu. U ovom slučaju to je bio niz «-/a». Strukturna organizacija kromosoma je očuvana i stoga su sve novostvorene jedinke sintaksno ispravne.

3.3.2. Transponiranje u korijen kromosoma

Svi elementi za transponiranje u korijen kromosoma započinju funkcijskim znakom i stoga se uvijek biraju u glavi. Ti elementi se još nazivaju nizovi za umetanje u korijen (eng. *root insertion sequence, RIS*). Metodom slučajnog odabira odabere se neka pozicija i ako element na toj poziciji nije funkcija pomičemo se udesno prema repu dok ne naiđemo na funkcijski znak. Ta funkcija postaje početna pozicija RIS elementa. Ako se niti jedna funkcija ne pronađe operator ne čini ništa. Tipično, koristi se stopa transponiranja od 0.1 i i skup od tri RIS elementa različitih veličina. Taj operator metodom slučajnog odabira odabire kromosom, gen koji će biti modificiran, početak RIS elementa i njegovu duljinu. Pogledajmo kromosom od dva gena ispod :

```
012345678901234560123456789012345 6
*-bQ/++/babbabbba//Q*baa+bbbabbbbb
```

Pretpostavimo da je odabran podniz «Q/+» u genu jedan. Transponiramo taj podniz u korijen kromosoma, čime dobivamo sljedeći kromosom :

```
012345678901234560123456789012345 6
Q/+*’bQ/babbabbba//Q*baa+bbbabbbbb
```

Primijetimo da se tijekom transpozicije cijela glava posmiče u desno, tako da se RIS element može ubaciti, gubeći na taj način posljednje elemente glave. Broj elemenata glave koji će biti zamijenjeni je jednak broju znakova RIS elementa. U ovom slučaju podniz «++/» je obrisan. Primijetimo opet da je novostvorena jedinka na ovaj način sintaksno ispravna.

3.3.3. Transponiranje gena

Kod transponiranja gena cijeli gen se uzima i transponira na početak kromosoma. Za razliku od drugih metoda transponiranja, gen koji se transponira briše se na svojoj originalnoj poziciji. Transponiranjem gena se samo premještaju geni i za gramatička stabla vezane komutativnom funkcijom u kratkom roku to ne znači mnogo. Međutim, transpozicija gena je veoma važna u kombinaciji s ostalim operatorima zato jer omogućava ne samo dupliciranje gena nego i generaliziranu rekombinaciju gena i manje gradivne blokove.

Kromosom iz kojeg će se odabrati gen za transponiranje je odabran metodom slučajnog odabira. Iz tog kromosoma metodom slučajnog odabira bira se jedan gen, osim prvog, za transponiranje. Promotrimo sljedeći kromosom koji se sastoji od tri gena :

```
012345678901201234567890120123456789012
/+Qa*bbaaabaa*a*/Qbbbbbabb/Q-aabbaaabbb
```

Gen tri je odabran za transponiranje. Dobiva se sljedeći kromosom :

```
012345678901201234567890120123456789012
/Q-aabbaaabbb/+Qa*bbaaabaa*a*/Qbbbbbabb
```

Ako je funkcija koja se koristi za povezivanje algebarska i komutativna, onda ova promjena nema veliki značaj, u suprotnom ima veliku ulogu.

3.4. Rekombinacija

Kod GEP - a postoje tri vrste rekombinacije :

- rekombinacija s jednom točkom prekida;
- rekombinacija s dvije točke prekida;
- rekombinacija gena.

Kod bilo koje vrste rekombinacije, dvije jedinke se odabiru metodom slučajnog odabira i zatim se sparuju sa ciljem razmijene genetskog materijala stvarajući dva nova kromosoma kćeri. Kromosomi kćeri se razlikuju jedna od druge isto koliko se razlikuju i od roditelja.

3.4.1. Rekombinacija s jednom točkom prekida

Kod rekombinacije s jednom točkom prekida kromosomi se sparuju i dijele u istoj točki. Materijal desno od točke prekida se izmjenjuje između dva kromosoma. Pogledajmo na primjeru :

```
        01234567890123456012345678901234 56
        +*-b-Qa*aabbbbaaa-Q-//b/*aabbabbab
        ++//b//-bbbbbbbbb-*-ab/b+bbbaabbaa
```

Pretpostavimo da je točka šest u genu jedan odabrana za točku prekida. Kromosomi se prekidaju u točki prekida i genetski materijal se razmjenjuje. Nove jedinke izgledaju :

```
        01234567890123456012345678901234 56
        +*-b-Q/-bbbbbbbbb-*-ab/b+bbbaabbaa
        ++//b/a*aabbbbaaa-Q-//b/*aabbabbab
```

Ovaj operator je veoma važan operator za unošenje varijacija u populaciju. Stopa rekombinacije s jednom točkom prekida koja se najčešće koristi je između 0.3 i 0.7.

3.4.2. Rekombinacija s dvije točke prekida

Kod rekombinacije s dvije točke prekida dva kromosoma se sparuju i dvije točke prekida se odabiru metodom slučajnog odabira. Materijal između točaka prekida razmjenjuje se na sljedeći način :

```
01234567890123456012345678901234 56
*-+Q/Q*Qaaabbbbabb QQab*++-aabbabaab
Q/-b-+/abaabbbaab/*-aQa*babbabbabb
```

Pretpostavimo da je pozicija pet u genu jedan i pozicija sedam u genu dva odabrana kao točka prekida metodom slučajnog odabira. Rekombinacijom dobivamo sljedeća dva kromosoma.

```
01234567890123456012345678901234 56
*-+Q/+/abaabbbaab/*-aQa*-aabbabaab
Q/-b-Q*Qaaabbbbabb QQab*++babbabbabb
```

Vrijedi primijetiti da je nekodirajući dio kromosoma idealno mjesto za točku prekida.

3.4.3. Rekombinacija gena

U trećoj vrsti rekombinacije, rekombinacija gena, cijela dva gena se izmjenjuju između dva kromosoma roditelja i na taj način formiraju dva nova kromosoma kćeri koje imaju gene od oba roditelja. Pogledajmo na sljedećem primjeru :

```
012345678901201234567890120123456789012
/+/ab-aabbbbb-aa**+aaabaaa-+--babbbbaab
+baQaaaabaaba*-+a-aabbabbb/ab/+bbbabaaa
```

Pretpostavimo da je gen dva bio odabran za razmjenu. Dobivamo sljedeća dva kromosoma kćeri.

```
012345678901201234567890120123456789012
/+/ab-aabbbbb*-+a-aabbabbb-+--babbbbaab
+baQaaaabaaba-aa**+aaabaaa/ab/+bbbabaaa
```

Kromosomi kćeri sadrže cijele gene kromosoma roditelja. Primijetimo, s ovakvom vrstom rekombinacije, slični geni se mogu razmijeniti, no uglavnom se razmjenjuju različiti geni i na taj način se novi materijal unosi u populaciju[1].

4. Pravila raspoređivanja

Raspoređivanje igra važnu ulogu u kontroli sustava okoline raspoređivanja (eng. *shop floor control system*) koji ima značajan utjecaj na performanse okoline raspoređivanja (eng. *shop floor*). Raspoređivanjem se alociraju ograničeni resursi (uglavnom strojevi) za aktivnosti (uglavnom poslovi) s ciljem optimiziranja jednog ili više kriterija (npr. minimiziranje ukupnog trajanja (eng. *makespan*), trajanja protjecanja (eng. *flow time*), kašnjenja (eng. *lateness*) ili zakašnjelosti (eng. *tardiness*). Tokom posljednjih godina sve je više efikasnih metoda raspoređivanja za sustave kontrole raspoređivanja (eng. *shop floor control*) otkriveno s razvojem metodologija raspoređivanja, kako u istraživanjima i praksi tako i tehnološkim napretkom u računarstvu. Probleme raspoređivanja možemo podijeliti u dvije kategorije, statičke probleme raspoređivanja i dinamičke probleme raspoređivanja. Kod statičkih problema raspoređivanja, pretpostavlja se da su svi atributi poznati na početku i da ostaju nepromijenjeni do završetka. No rijetko je moguće da su svi atributi poslova za raspoređivanje poznati unaprijed i da ostanu nepromijenjeni tijekom trajanja stvaranja rasporeda zato jer postoje razne vrste nepredvidljivih događaja. Na primjer, poslovi dolaze kontinuirano tijekom vremena, strojevi se kvare i popravljaju, i rokovi (eng. *due dates*) poslova se mijenjaju tijekom obrade. Takvi poslovi se nazivaju dinamički poslovi raspoređivanja. U ovom radu se razmatra problem raspoređivanja *n* poslova koji se pojavljuju tijekom vremena na stroju s ciljem optimizacije jednog ili više kriterija. To je definicija dinamičkih poslova raspoređivanja na jednom stroju s vremenima pripravnosti. Takvi poslovi se razmatraju iz sljedećih razloga :

- široko su rasprostoranjeni u praksi i moraju hitno biti riješeni;
- problemi raspoređivanja na jednom stroju često formiraju komponente rješenja za kompleksnije probleme raspoređivanja.

Tijekom posljednja dva desetljeća učinjeni su značajniji napori za predlaganje novih tehnika raspoređivanja dinamičkih poslova. Postojeće strategije podijeljene su u tri kategorije :

- potpuno reaktivni pristup (eng. *completely reactive approaches*)
- stvaranje robusnih rasporeda (eng. *robust scheduling approaches*)
- prediktivno – reaktivne pristupe (eng. *predictive - reactive approaches*)

Mnoge heuristike, tzv. pravila raspoređivanja (eng. *dispatching rules*) koriste se za izračunavanje funkcije prioriteta za svaki posao. Na temelju izračunatih prioriteta poslovi se raspoređuju. Prioritetna funkcija koja je enkapsulirana u heuristici i dodjeljuje vrijednosti poslovima naziva se pravilo raspoređivanja (eng. *scheduling rule, SR*).

4.1. Dinamički problemi raspoređivanja na jednom stroju

Dinamički problemi raspoređivanja na jednom stroju (eng. *Dynamic single machine scheduling problem,* DMSP) s vremenima pripravnosti su opisani na sljedeći način. Okolina raspoređivanja sastoji se jednog stroja i *n* poslova, koji pristižu tokom vremena i obrađuju se samo jednom. Svaki posao je određen s više atributa, kao trajanje obrade (eng. *processing time*) p_i, vrijeme pripravnosti (eng. *release date*) r_i, rok (eng. *due date*) d_i i težina (eng. *weight*) w_i, kojom je određena važnost posla *i, i = 1, 2, …, n.* Atributi poslova su poznati samo za poslove koji su trenutno dostupni na stroju. Također, pretpostavlja se da stroj u određenom trenutku može obrađivati samo jedan posao. Cilj je odrediti redoslijed poslova na stroju takav da je zadovoljen jedan ili više optimizacijskih kriterija. Postoje četiri optimizacijska kriterija. Uvodi se dodatna oznaka c_i koja predstavlja vrijeme završetka posla *i.*

Ukupno trajanje

$$C_{max} = max(c_i, i = 1, ..., n)$$

Ukupno trajanje protjecanja

$$F - \sum_{i-1}^{n} (c_i \quad r_i)$$

Maksimalno kašnjenje

$$L_{max} = max(c_i - d_i, i = 1, ..., n)$$

Ukupna zakašnjelost

$$T - \sum_{i-1}^{n} max(c_i \quad d_i, 0)$$

4.2. Heuristika za problem DSMSP s vremenima pripravnosti

U statičkom okruženju, svi atributi poslova su poznati unaprijed (u trenutku *t* = 0) i ne mijenjaju se tijekom vremena. Stoga cijeli raspored uglavnom može bit napravljen na početku. Ovdje to nije slučaj. U ovom poglavlju opisuje se heuristika za raspoređivanje poslova s vremenima pripravnosti na jednom stroju, čiji se pseudokod nalazi niže u tekstu. Vremena pripravnosti nisu poznata unaprijed.

```
Heuristika za DSMSP za poslove s vremenima pripravnosti :
t = 0, početno vrijeme;
dok(postoje neraspoređeni poslovi) {
  JSs(t) = {svi poslovi koji zadovoljavaju wti < Pmin(t)};
  Računanje prioritenih vrijednosti za sve poslove iz JSs(t);
  Rasporedi posao s najvećim prioritetom na stroj;
  Obriši posao iz J*;
  Ažuriraj t = vrijeme završetka J*;
}
```

$JS_s(t)$ predstavlja skup poslova koji mogu biti raspoređeni u trenutku *t.* wt_j predstavlja vrijeme čekanja posla *j*, dakle $wt_j = max\{r_j - t, 0\}$. $P_{min}(t)$ predstavlja najkraće vrijeme obrade poslova koji su već pristigli ali nisu raspoređeni u trenutku *t*.

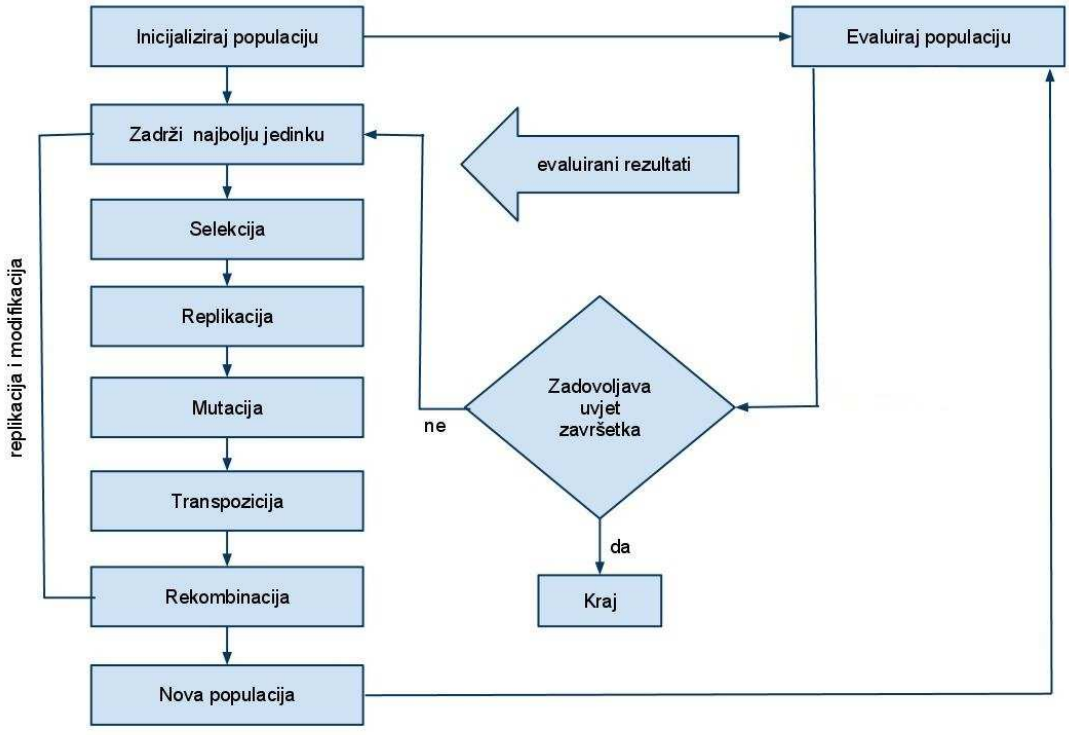
Može se primijetiti da se najbolji prioritet može shvatiti dvojako. Najbolji prioritet može imati ili posao s najvećom vrijednošću ili onaj s najmanjom. U radu se definira da posao s manjom vrijednošću ima veći prioritet.

U heuristici, pravilo raspoređivanja je važna komponenta i njegovo ponašanje ima značajan učinak na performanse[2].

4.3. Programsko rješenje evolucije i evaluacije

GEP funkcionira kao heuristička tehnika pretraživanja koja će pretražiti prostor pravila raspoređivanja za dani problem.

Programski dio se sastoji od dva dijela, modula za učenje i simulacijskog modula. Simulacijski modul radi kao evaluator performansi, a modul za učenje koristi GEP za evoluciju pravila raspoređivanja na temelju rezultata vraćenih od simulacijskog modula. Skica modula za učenje i simulacijskog modula prikazana je na slici 1.



Slika 1. Programski dio

Dio za učenje započinje s inicijalnom populacijom koja se sastoji od slučajno generiranih pravila raspoređivanja. Ta pravila se prenose simulacijskom modulu i procjenjuju se korištenjem jednog ili više kvantitativnih mjera učinka. Rezultati mjerenja svakog kandidata se vraćaju natrag modulu za učenje, gdje će se sljedeća populacija pravila biti stvorena i modificirana prema trenutnom najboljem rješenju koristeći evolucijske operatore poput replikacije, mutacije, selekcije i transpozicije. Zatim se sljedeći skup pravila prosljeđuje modulu za simulaciju. To se ponavlja sve dok uvjet zaustavljanja nije zadovoljen.

4.3.1. Funkcijski i završni znakovi

Svaki kromosom GEP - a generira se na početku pretrage i modificira tijekom evolucije elementima iz skupa završnih i funkcijskih znakova. GEP koristi predefiniran skup elemenata za otkrivanje potencijalnih rješenja za dani problem. Stoga, izbor odgovarajućih elemenata za funkcijski skup i završni skup je ključni korak u provedbi procesa optimizacije. Funkcijski i završni skup znakova je definiran na sljedeći način.

Funkcijski skup završnih znakova uključuje funkcije «+», «-» i «*» koje predstavljaju odgovarajuće aritmetičke funkcije i «/» koji predstavlja funkciju dijeljenja koja vraća jedan ako je djelitelj jednak nuli.

Skup završnih znakova: uključuje elemente koji označavaju trenutno stanje i osobine poslova za raspoređivanje, kao što su:

- *p* - trajanje obrade (eng. *processing time*)
- *r* - vrijeme pripravnosti (eng. *release date*)
- *d* - rok (eng. *due date*)
- *sl* - dopušteno vrijeme kašnjenja (eng. *job's positive slack*), $max\{d - p - max\{t, r\}, 0\}$, gdje *t* predstavlja trenutno vrijeme.
- *st* - trajanje zadržavanja, $max\{t - r, 0\}$, gdje *t* predstavlja trenutno vrijeme.

- *wt* - trajanje čekanja, $max(r - t, 0)$, gdje *t* predstavlja trenutno vrijeme.

5. Rezultati mjerenja

Razumne postavke za parametre GEP – a otkrivaju se kroz opsežne eksperimente uključujući veličinu populacije, uvjet zaustavljanja, broj gena u kromosomu, duljina glave gena, različite stope rekombinacije, mutacije, transpozicije. Parametri prikazani u tablici 1. su odabrani na temelju prijašnih rezultata dobivenih u [2].

Tablica 1. Parametri GEP - a

Parametri	**Vrijednosti**
Veličina populacije	100
Uvjet zaustavljanja	500 iteracija
Veličina glave kromosoma	15
Broj gena u kromosomu	3
Stopa rekombinacije	0.9
Stopa transpozicije	0.3, 0.1, 0.1
Stopa mutacije	0.2

Da bi se ocijenila učinkovitost ove metode korištena su dva algoritma raspoređivanja za usporedbu rezultata. Odabrani algoritmi su «trošak prividnog kašnjenja» (eng. *Apparent tardiness cost rule*, ATC) i «najraniji datum dospijeća»(eng. *earliest due date*, EDD). Oni su odabrani zato jer su primjenom tih algoritama dobiveni najbolji rezultati u [2].

5.1. Trošak prividnog kašnjenja

Ovaj algoritam procjenjuje kaznu za kašnjenje korištenjem eksponencijalne funkcije, tj. prioriteti poslova koji će biti raspoređeni računa se sljedećom funkcijom.

$$\frac{1}{p_i} e^{-\frac{(d_i - t - p_i)}{k \bar{p}}}$$

Slika 1. Funkcija određivanja prioriteta

Ako posao kasni onda se ATC svodi na $1/p_i$. U suprotnom slučaju, svodi se na «minimum slack time» pravilo[2]. Također, ako je procijenjeno čekanje veliko ATC se opet svodi na $1/p_i$. Posao s najvećim prioritetom se sljedeći raspoređuje na stroju. Faktor k (eng. *look – ahead factor*) ima velik utjecaj na raspoređivanje. Može varirati od 0.5. do 4.5. Vrijednost korištena u ovom radu je 1.0.

5.2. Najraniji rok

Svi poslovi koji čekaju da budu raspoređeni sortiraju se u uzlaznom redoslijedu prema svojem roku d_i (eng. due dates). Prvi posao se rasporedi na stroju. Ovo pravilo je najpopularnije pravilo od svih pravila baziranih na rokovima.

5.3. Ispitni primjeri

Programski dio je implementiran u programskom jeziku Java. Svi skupovi primjera su izvršavani na računalu «Lenovo Thinkpad T400» sa sljedećim karakteristikama:

- Procesor: Intel Core 2 Duo T9600 (2.83GHz, 1066MHz FSB, 6MB Cache)
- Memorija: 2GB DDR3 RAM

5.3.1. Skupovi primjera s deset poslova

Instance problema se generiraju slučajno. Svaki posao se generira sa svim svojim parametrima čije s vrijednostima kao u [2]. Broj poslova za koje se ispitivala učinkovitost GEP – a je 10. Za svaki optimizacijski kriteriji stvara se 5 različitih ispitnih primjera, a svaki ispitni primjer se raspoređivao 10 puta. Na temelju dobivenih podataka rezultati su prikazani u odstupanju od najboljeg rješenja determinističkog algoritma, tj. ili od ATC – a ili od EDD – a.

Niže u tekstu prikazani su rezultati u tablicama za skupove primjera s 10 poslova.

Rezultati dobiveni za minimiziranje ukupnog trajanja prikazni su u tablici 2.

Tablica 2. Minimiziranje ukupnog trajanja, usporedba s EDD pravilom

Skup testova	ATC (% greška)	EDD (% greška)	GEP (% greška)
1	20.819	0.0	-4.521
2	20.610	0.0	-5.325
3	20.011	0.0	-5.120
4	20.389	0.0	-5.291
5	21.128	0.0	-4.893

Najbolja generirano pravilo, tj. funkcija prioriteta u ovom slučaju je *r*. Iz toga se zaključuje da je GEP sposoban naučiti koji je najvažniji parametar za minimizaciju ukupnog trajanja. Također, zaključuje se da minimizacija ukupnog trajanja ne ovisi o ostalim parametrima, nego samo o vremenu pripravnosti *r*.

Rezultati dobiveni za minimiziranje trajanja protjecanja prikazani su u tablici 3.

Tablica 3. Minimiziranje trajanje protjecanja, usporedba s ATC pravilom

Skup testova	ATC (% greška)	EDD (% greška)	GEP (% greška)
1	0.0	9.348	-1.258
2	0.0	5.383	-1.104
3	0.0	6.340	-1.334
4	0.0	7.477	-1.202
5	0.0	5.667	-1.289

Najbolje generirano pravilo u ovom slučaju je $p + 2wt + r/wt$. Iz ovog pravila primjećuje se da minimiziranje trajanja protjecanje ne ovisi o roku *d* i o zalihi vremena *sl*.

Rezultati dobiveni za minimiziranje maksimalnog kašnjenja prikazani su u tablici 4.

Tablica 4. Minimiziranje maksimalnog kašnjenja, usporedba s EDD pravilom

Skup testova	ATC (% greška)	EDD (% greška)	GEP (% greška)
1	0.550	0.0	-16.164
2	0.230	0.0	-16.583
3	0.254	0.0	-16.900
4	0.619	0.0	-16.446
5	0.626	0.0	-16.199

Najbolje generirano pravilo u ovom slučaju je *2d + sl + p + r + wt*. Kao što se primjećuje minimiziranje maksimalnog kašnjenja ovisi o svim parametrima.

Rezultati dobiveni za minimiziranje ukupne zakašnjelosti prikazani su u tablici 5.

Tablica 5. Minimiziranje ukupne zakašnjelosti, usporedba s EDD pravilom

Skup testova	ATC (% greška)	EDD (% greška)	GEP (% greška)
1	4.262	0.0	-11.033
2	4.748	0.0	-11.942
3	4.762	0.0	-11.420
4	4.938	0.0	-11.089
5	4.370	0.0	-11.063

Najbolje generirano pravilo u ovom slučaju je *sl + 2p + r wt*^2.

Kao što se primjećuje iz rezultata mjerenja GEP u svim slučajevima daje najbolje rješenje.

5.3.2. Skupovi primjera s pedeset poslova

Skupovi primjera se generiraju kao i u slučaju s deset poslova. Za svaki optimizacijski kriterij stvorena su tri različita testna primjera. U ovim primjerima korištena su pravila koja su dobivena evolucijom na skupovima primjera od deset poslova. Dobiveni rezultati prikazani su u tablici 6. do tablica 9. Prikaz uspješnosti pojedinog algoritma za svaki od navedenih optimizacijskih kriterija je dan u tablici 11.

Tablica 6.Minimiziranje ukupnog trajanja

Skup testova	ATC (% greške)	EDD (% greške)	GEP (% greške)
1	0.0	1.728	-1.758
2	0.0	1.239	-1.730
3	0.0	1.872	0.231

Tablica7. Minimiziranje trajanja protjecanja

Skup testova	ATC (% greške)	EDD (% greške)	GEP (% greške)
1	0.0	-3.772	-1.606
2	0.0	2.171	-1.783
3	0.0	2.219	-1.501

Tablica8. Minimiziranje maksimalnog kašnjenja

Skup testova	ATC (% greške)	EDD (% greške)	GEP (% greške)
1	0.0	2.395	-2.686
2	0.0	1.726	-0.291
3	0.0	2.219	-1.178

Tablica 9. Minimiziranje ukupne zakašnjelosti

Skup testova	ATC (% greške)	EDD (% greške)	GEP (% greške)
1	0.0	5.076	-25.749
2	0.0	1.846	-25.312
3	0.0	-3.165	-25.626

Tablica 10. Prikaz uspješnosti pojedinog algoritma na skupovima s 50 poslova

Kriterij optimizacije	ATC	EDD	GEP
Minimiziranje ukupnog trajanja	3	1	2
Minimiziranje trajanja protjecanja	2	1	3
Minimiziranje maksimalnog kašnjenja	3	0	3
Minimiziranje ukupne zakašnjelosti	2	1	3

GEP je dao najbolje rješenje za 11 različitih ispitnih primjera, ATC za 10 različitih ispitnih primjera, a EDD za 3 različita ispitna primjera. GEP se ponovno pokazao kao najuspješniji, no za razliku od prijašnjeg slučaja ATC pravilo se pokazalo uspješnijim od EDD pravila, iako je bilo očekivano slično ponašanje kao u skupu s deset poslova gdje je EDD bio uspješniji.

6. Zaključak

U ovom radu razmatra se problem raspoređivanja dinamičkih poslova s poznatim vremenima pripravnosti na jednom stroju. Kao rješenje problema koristi se nova metoda genetskog programiranja koju je predložio C.Ferreria, tzv. gramatičko genetsko programiranje (GEP). GEP služi kao heuristička metoda pretrage prostora rješenja. GEP – om se automatski evoluiraju pravila raspoređivanja koja predstavljaju rješenje promatranog problema. Za minimizaciju ukupnog trajanja, ukupnog protjecanja, maksimalnog kašnjenja i ukupne zakašnjelosti učinkovitost GEP – a je bila ispitivana na brojnim primjerima. Dobiveni rezultati ispitivanja GEP – om pokazali su se boljima i uspješnijima u usporedbi s performansama ATC i EDD pravila što je potvrdilo rezultate dobivene u drugim istraživanjima[2].

7. Literatura

1. Ferreria, C. Gene expression programming in problem solving, WSC6 Tutorial, 2001.
2. Nie, Li. Shao, X. Gao, L. Li, W. Evolving scheduling rules with gene expression programming for dynamic single – machine scheduling problems, Springer – Verlang London, 2010.
3. Golub, M. Skripta 1.dio : Genetski algoritmi, 27. rujan 2004. *Genetski algoritmi, prvi dio,* http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf, 1. lipanj 2009.
4. Jakobović, D. Budin, L. Dynamic scheduling with genetic programming, http://www.zemris.fer.hr/~yeti/download/EuroGP_2006.pdf , 2006.

8. Sažetak

U ovom radu se razmatra problem raspoređivanja *n* poslova s vremenima pripravnosti na jednom stroju primjenom gramatičkog genetskog programiranja. Pritom postoje četiri različita optimizacijska kriterija, minimiziranje ukupnog trajanja, minimiziranje ukupne zakašnjelosti, minimiziranje trajanja protjecanja i minimiziranje maksimalne zakašnjelosti. Genetsko gramatičko programiranje koristi se kao heuristička metoda pretrage pravila raspoređivanja.

Ključne riječi : raspoređivanje na jednom stroju, dinamički poslovi, vremena pripravnosti, pravila raspoređivanja, genetsko gramatičko programiranje

9. Abstract

The paper considers the problems of scheduling *n* jobs that are released over time on a machine in order to optimize one or more objectives. The problems are dynamic single-machine scheduling problems with job release dates. Gene expression programming was proposed to construct effective scheduling rules. Gene expression programming worked as a heuristic search to search the space of SRs.

Key words : Single machine scheduling, dynamic scheduling, release dates, scheduling rules, gene expression programming