Playing Regex Golf with Genetic Programming

Projekat u okviru kursa Računarska inteligencija
Matematički fakultet
Univerzitet u Beogradu

Anđela Ilić
mi17105@alas.matf.bg.ac.rs
Mina Milošević
mi17081@alas.matf.bg.ac.rs

Februar 2021

Sadržaj

1 Opis problema

Data su dva skupa reči - M i U. Cilj *Regex Golf* igre je pronaći najkraći regularni izraz kojim se mogu zapisati sve reči iz skupa M, ali kojim se ne može zapisati nijedna reč skupa U. Za date skupove M i U ne možemo sa sigurnošću da tvrdimo da postoji rešenje koje zadovoljava prethodne uslove. Takođe, ako dobijemo regularni izraz koji zadovoljava navedene uslove, ne možemo za svaki primer znati da li postoji i bolje rešenje tj. kraći regularni izraz.

2 Implementacija

Svaka jedinka u Genetskom programiranju će biti predstavljena kao drvo. U listovima nalaze elementi koje ćemo jednim imenom zvati *Terminali* (terminal set), a u unutrašnjim čvorovima su elementi koje nazivamo *Funkcije* (function set).
Skup funkcija sadrži operatore koji se mogu javiti u regularnim izrazima. Primeri takvih operatora su: .∗, .+, .?, .{.,.}+, (.), [.], [^.], .., .|.. Tačka . je mesto na kome se nalaze deca u drvetu.
Skup terminala čine elementi koji zavise i koji ne zavise od ulaznih skupova M i U. Elementi koji su nezavisni - opsezi malih i velikih slova, opsezi brojeva u regularnim izrazima, karakteri ^ i $, wildcard karakter '%' (kasnije se transformiše u .). Elementi skupa terminala koji su zavisni - skup karaktera iz M, parcijalni opsezi karaktera iz M i n-grami.

2.1 Obrada ulaznih podataka

Bez obzira na veličinu ulaznih skupova M i U i njihov sadržaj, implementacija prethodno navedenih zavisnih terminala je ista.
Skup karaktera iz M sadrži sve karaktere koji se mogu naći u rečima iz M, bez ponavljanja i sortirani po engleskom alfabetu.
Parcijalni opsezi se prave na osnovu skupa karaktera iz M. Potrebno je naći maksimalne podskupove tog skupa karaktera tako da se svi karakteri iz intervala $[c_f, c_l]$ nalaze u skupu karaktera iz M. c_f je prvi karakter, a c_l je poslednji karakter iz podskupa. Kao rezultat se vraćaju parcijalni opsezi u formatu $c_f - c_l$.
n-grami. Pravimo skup svih n-grama dužine $2 \leq n \leq 4$ koji se mogu naći u M ili u U (ili oba). Svakom dobijenom n-gramu se dodelju vrednost koja predstavlja njegov *score*. Za svaku reč iz M koja sadrži dati n-gram, njegov *score* se uvećava za 1, a za svaku reč iz U koja ga sadrži, *score* se umanjuje za 1. Nakon formiranja svih n-grama i određivanja njihovih vrednosti, sortiraju se opadajuće po vrednosti. Potrebno je uzeti najmanji podskup n-grama tako da njihova ukupna vrednost (*score*) bude jednaka bar dužini skupa M ($|M|$).

2.2 Genetsko programiranje

2.2.1 Implementacija jedinke

Svaka jedinka se predstavlja preko *apstraktnog sintaksnog stabla* (AST). U korenu stabla se nalazi karakter '.' i koren uvek ima dva deteta. Elementi u unutrašnjim čvorovima se biraju *random* iz skupa *Function*, a elementi u listovima su *random* izabrani iz skupa *Terminal*.
Na osnovu izabranog elementa za unutrašnji čvor dobijamo informaciju koliko dece će taj čvor imati - .∗, .+, .?, (.), [.], [^.] će imati jedno dete; .., .|. će imati dva deteta; .{.,.}+ ima tri deteta. Od ovako kreiranog drveta se dobija niska koja predstavlja validan regularni izraz.
Klasa *Individual* koja predstavlja jedinku takođe sadrži i brojeve n_m i n_u - broj reči iz

skupova M i U, redom, koje su opisane dobijenim regularnim izrazom.
Za svaku jedinku računamo i *fitnes* funkciju po formuli:

$$f(x) = w_i * (n_m - n_u) - length(r)$$

gde je w_i unapred zadata konstanta za dati primer, a $length(r)$ je dužina regularnog izraza. Pošto želimo što kraći regularni izraz koji obuhvata sve reči iz M i nijednu reč iz U, ovako definisanu funkciju *fitnes* maksimizujemo.

2.2.2 Parametri genetskog programiranja

2.2.3 Selekcija, ukrštanje, mutacija

Za *selekciju* koristimo turnirsku selekciju veličine 7. Jedinke za selekciju biramo random i uzimamo najbolju jedinku tj. onu koja ima najveći fitnes među odabranim.

3 Rezultati

4 Zaključak

5 Izvori