

OS2022 - Domaći 4

Konkurentni autocomplete

Cilj domaćeg zadatka je napisati C program za Linux sistem koji korisniku obezbeđuje autocomplete funkcionalnost. Korisnik unosi proizvoljan prefiks, i na ekranu dobija reči koje počinju sa tim prefiksom. Potencijalne reči se čitaju iz tekstualnih datoteka. Reči u memoriji treba da se skladište u trie strukturi, koja treba da podržava konkurentno ubacivanje reči, kao i konkurentnu pretragu.

Ovaj zadatak ne zahteva rad u xv6 sistemu, niti mu je postavka grana xv6 repozitorijuma.

Uz zadatak su priloženi podaci koji mogu da se koriste za testiranje, u okviru arhive data.zip, koja se nalazi na materijalima. **Pri predaji se obavezno pobrinuti da se ovi podaci uklone iz projekta pre predaje.**

Zadatak podrazumeva samostalno čitanje man page dokumenata i drugih materijala radi pronalaženja odgovarajućih funkcija i sistemskih poziva za manipulaciju direktorijuma i vremena modifikacije.

Na materijalima postoji snimak koji demonstrira kako izgleda urađen zadatak.

Interakcija sa korisnikom

Korisnik sistemu može da zadaje komande preko komandne linije. Treba podržati sledeće komande:

- `_add_ <dir>`
- `_stop_`

Komanda `_add_` dodaje novi direktorijum sa tekstualnim datotekama u sistem. Pri izvršavanju ove komande treba kairati novu scanner nit koja će da skenira taj direktorijum. Naziv direktorijuma se daje bez poslednje kose crte.

Komanda `_stop_` gasi aplikaciju.

Ako korisnik na konzoli otkuca bilo šta drugo i pritisne enter, podrazumeva se da je poslednja reč uneta na toj liniji prefiks za koji korisnik želi da dobije autocomplete rezultate. Ukoliko korisnik unese više reči u istoj liniji, bilo koje pre poslednje se ignorišu. Na ekranu treba odmah ispisati sve reči koje su do sada pronađene koje počinju sa datim prefiksom. Ako pretraga još uvek traje, i pronađe se neka nova reč koja počinje sa tim prefiksom, treba je ispisati na ekranu. Korisnik ne može da unosi komande dok se ispisuju ove reči. Korisnik završava trenutnu pretragu kombinacijom tastera CTRL+D. Nakon unosa CTRL+D se zaustavlja ispis i korisnik može da unese novu komandu ili prefiks za pretragu.

Scanner nit

Svaka scanner nit obilazi svoj direktorijum i čita reči iz njega. Nit treba da ignoriše bilo kakve poddirektorijume i da čita sve datoteke, pretpostavljajući da one sadrže običan ASCII kodiran tekst. Ova nit treba da prati koje datoteke je već pročitala, kao i kada su te datoteke poslednji put menjane pomoću deljenog niza koji sadrži spisak svih do sada skeniranih datoteka, kao i njihovih poslednjih vremena izmena.

Kada nit naiđe na neku datoteku koju nije već pročitala, ili datoteku koja je menjana od prethodnog čitanja, ona započinje čitanje. Nit čita reč po reč iz datoteke. Reči su razdvojene znakovima SPACE (' '), TAB ('\t') ili ENTER ('\n'). Ako reč sadrži bilo kakve karaktere osim malih ili velikih slova engleske abecede, treba je celu ignorisati. Reči će imati maksimalnu dužinu od 63 slova. Za reči duže od toga zadržati samo prvih 63 karaktera. Kada je reč pročitana, sva velika slova treba pretvoriti u mala. Tako obrađena reč se ubacuje u trie strukturu koja je deljena između svih scanner niti.

Kada nit obiđe svoj zadati direktorijum, ona pravi pauzu od pet sekundi, nakon čega ponovo

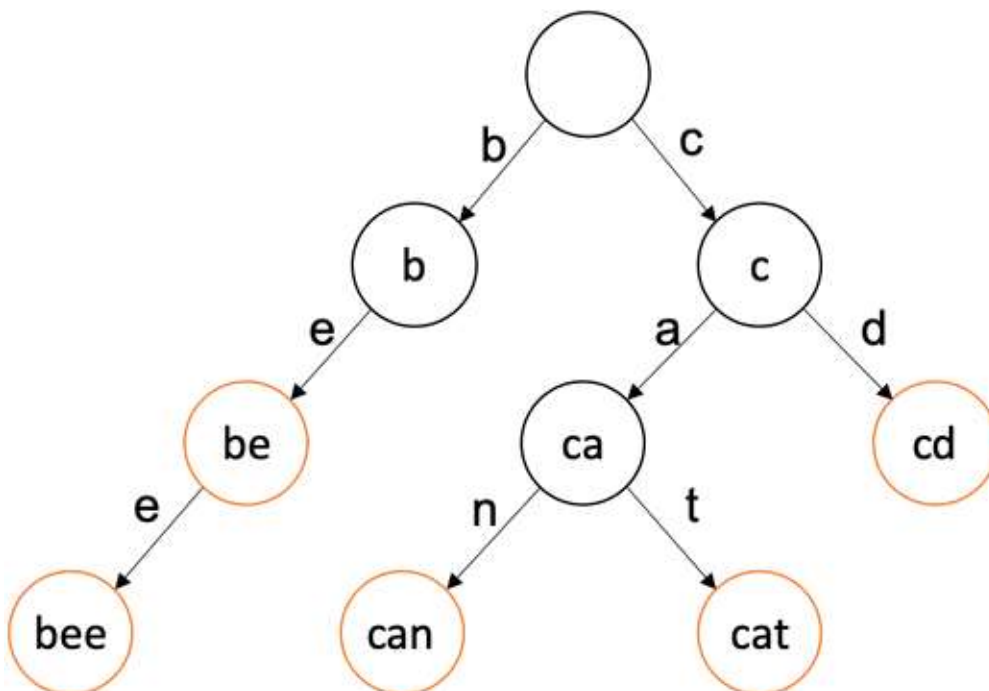
počinje obilazak kompletnog direktorijuma. U ovom novom obilasku će se čitati samo nove i izmenjene datoteke. Nit treba da obavlja ovaj posao u beskonačnoj petlji. Ovo znači da sistem treba da detektuje i reaguje na dodavanje nove datoteke, kao i izmenu postojećih datoteka.

Trie struktura

Trie (ili prefiksno stablo) je struktura koja u svakom svom čvoru sadrži:

- Slovo
- Potencijalnih 26 pokazivača na decu čvorove
- Pokazivač na roditelja
- Flag koji naznačava kraj reči
- Broj podreči, ne računajući samu sebe

Koren strukture je tipično prazan. Reč u trie strukturi se predstavlja kao niz čvorova od korena do lista, gde svaki naredni čvor u putanji predstavlja jedno slovo reči. Na narednoj slici je prikazan trie u koji su upisane reči: be, bee, can, cat, cd.



Narandžasti čvorovi imaju postavljen flag koji naznačava kraj reči. Primera radi, čvor **b** bi kod naše strukture imao 2 kao broj podreči, dok bi čvor **c** imao 3 kao broj podreči. Prikupljanje “autocomplete” rezultata za neki prefiks se svodi na obilazak čitavog podstabla, počevši od

čvora koji predstavlja traženi prefiks. Broj podreči održavamo da bismo na početku tog obilaska mogli da inicijalizujemo strukturu koja će predstavljati rezultat pretrage. Ovaj brojač se osvežava praćenjem roditeljskih veza nakon ubacivanja nove reči kao lista u trie. Bitno je da u rezultatu bude tačno onoliko reči koliko piše u brojaču na čvoru prefiksa. Nije dozvoljeno da trie sam vrši ispis rezultata, već mora da može da ga vrati kao strukturu nekom pozivaocu. Rezultati koji su pronađeni naknadno treba da se automatski ispišu na ekranu. Kada korisnik unese CTRL+D ovaj automatski ispis treba da se zaustavi.

Konkurentnost

U aplikaciji će postojati $N + 1$ nit, gde N predstavlja broj izvršenih `_add_` komandi, a jedna dodatna nit je main nit, iz čijeg konteksta se izvršava pretraga deljene trie strukture. Sve ove niti treba da rade sa deljenom trie strukturom uz sledeća ograničenja:

- Nikada ne smeju da se ispisuju reči koje nisu tražene nekim prefiksom.
- U nizu skeniranih datoteka treba da se nalaze sve datoteke koje su do sada skenirane, sa njihovim tačnim poslednjim vremenima modifikacije.
- Nije dozvoljeno zaključavati čitavu trie strukturu.
- Ubacivanje čvorova, kao i pretraga trie strukture treba da mogu da se obavljaju konkurentno. Pritom je dozvoljeno da operacije dodavanja ili pretrage koje se rade nad istim prefiksom bilo koje dužine to rade uz međusobno isključivanje. Npr:
 - Dodavanje “vader” i “khan” treba da se izvršava konkurentno.
 - Dodavanje “vader” i “vargoth” treba da se međusobno isključuje.
 - Pretraživanje rezultata za prefiks “luk” i dodavanje reči “jaina” treba da se izvršava konkurentno.
 - Pretraživanje rezultata za prefiks “luk” i dodavanje reči “lucas” treba da se međusobno isključuje.

Poenta ove vrste međusobnog isključivanja je da se obezbedi što je više moguće konkurentnog rada, dok i dalje imamo garanciju da pretraga neće da bude prekinuta ubacivanjem, te da proizvede neočekivane rezultate, kao i da dva ubacivanja u istom delu stabla neće da se dešavaju istovremeno, i da se na taj način napravi loša struktura stabla.

Funkcije i strukture

Sledi spisak predloga za strukture i funkcije unutar domaćeg zadatka. Dozvoljeno je i podstiče se izmena svega navedenog onako kako mislite da je prigodno da biste ispunili zahteve u zadatku.

```
#define MAX_WORD_LEN 64 //najveca dozvoljena duzina reci, uz \0
#define LETTERS 26 //broj slova u abecedi i broj dece u trie
```

```

extern void scanner_init(); //poziva se jednom na pocetku rada sistema
extern void *scanner_work(void *_args); //funkcija scanner niti

extern void trie_init(); //poziva se jednom na pocetku rada sistema
extern void trie_add_word(char *word); //operacija za dodavanje reci
extern search_result *trie_get_words(char *prefix); //operacija za pretragu
extern void trie_free_result(search_result *result); //rezultat se dinamički alocira pri
pretrazi, tako da treba da postoji funkcija za oslobadjanje tog rezultata

typedef struct trie_node //cvor unutar trie strukture
{
    char c; //slovo ovog cvora
    int term; //flag za kraj reci
    int subwords; //broj reci u podstablu, ne racunajuci sebe
    struct trie_node *parent; //pokazivac ka roditelju
    struct trie_node *children[LETTERS]; //deca
} trie_node;

typedef struct search_result //rezultat pretrage
{
    int result_count; //duzina niza
    char **words; //niz stringova, svaki string duzine MAX_WORD_LEN
} search_result;

typedef struct scanned_file //datoteka koju je scanner vec skenirao
{
    char file_name[256]; //naziv datoteke
    time_t mod_time; //vreme poslednje modifikacije datoteke
} scanned_file;

```

Bodovanje

Zadatak se boduje na sledeći način:

- | | | |
|---------------------------|----------|-----------|
| • Konkurentni skeneri | - | 4 boda |
| • Trie | - | 16 bodova |
| • Konkurentno dodavanje | - | 6 bodova |
| • Konkurentna pretraga- | 6 bodova | |
| • Reagovanje na dodavanje | - | 4 boda |

U slučaju da je neka od stavki implementirana parcijalno, biće dodeljeni parcijalni poeni.

Predaja i rokovi

Zadatak se predaje putem mail-a na bmilojkovic@raf.rs, lhudak@raf.rs aalhakan@raf.rs.

Direktorijum koji sadrži kod imenovati na sledeći način: "os_2022_d4_ime_prezime_ind".
Npr. "os_2022_d4_student_studentic_rn0101".

Arhivirati ovaj direktorijum (.zip) i arhivu poslati kao attachment uz mail. Pobrinuti se da se u arhivi nalazi isključivo izvorni kod, i da nema podataka ili izvršnih datoteka. Naziv arhive mora da bude u obliku: "os_2022_d4_ime_prezime_ind.zip"
Npr. "os_2022_d4_student_studentic_rn0101.zip"

U tekstu mail-a obavezno navesti:

- Ime i prezime
- Broj indeksa
- Grupa, po zvaničnom spisku, ili "ponovac" za ponovce

Subject mail-a mora da bude u obliku: "[OS 2022] D4 ime_prezime_ind".
Npr. "[OS 2022] D4 student_studentic_rn0101"

Rok za predaju je:

- Ponedeljak, 6. jun 23:59:59 za grupu koja sluša OS ponedeljkom.
- Utorak, 7. jun 23:59:59 za grupe koje slušaju OS utorkom.
- Četvrtak, 9. jun 23:59:59 za grupu koja sluša OS četvrtkom.

Rok je definisan po grupi kojoj student zvanično pripada. Studenti koju slušaju vežbe van svog termina i dalje moraju da poštuju termine na osnovu zvaničnog spiska. Za sve ponovce se primenjuje najkasniji rok, nezavisno od toga da li su obnovili čitavu godinu ili ne.

Neće se pregledati zadaci (tj. biće dodeljeno 0 poena) ako se desi bilo koje od:

- Sadržaj mail-a nije po navedenom obliku.
- Subject mail-a nije po navedenom obliku.
- Naziv arhive nije po navedenom obliku.
- Predaja se desi nakon navedenog roka.
- Kod se ne kompajluje.
- Kod nije uredno uvučen.

Odbrana domaćih zadataka je obavezna. Termin za odbranu trećeg domaćeg zadatka će biti zakazan ubrzo posle isteka roka. Odbrane će se vršiti po grupama. Ako ste iz bilo kog razloga

sprečeni da prisustvujete odbrani, obavezno to najavite što pre, kako bismo mogli da zakažemo vanredni termin za odbranu.

Svrha odbrane je da se pokaže autentičnost zadatka. Ovo podrazumeva odgovaranje na pitanja u vezi načina izrade zadatka, ili izvršavanje neke izmene nad zadatkom na licu mesta. U slučaju da odbrana nije uspešna, zadatak se boduje sa -5 bodova umesto namenjenog broja bodova.