Paralelizacija igre *Pacman*

David Vidović

Februar 2024

1 Uvod

Projektni zadatak je nastao kao poslednja predispitna obaveza iz predmeta Multiprocesorski sistemi koji u devetom semestru predaje prof. Vuk Vranjković, na MAS Embeded sistemi i algoritmi. Zadatak uključuje serijsku implementaciju popularne igre *Pacman* u C++ programskom jeziku koristeći biblioteku za obradu slike *OpenCV*. Potom je potrebno implementirati paralelizovanu verziju igre gde će različiti procesori ili različite niti podeliti određene funkcionalnosti igre i izvršavarti ih samostalno. Mentor na projektu je predmetni asistent Anja Tanović.

2 Implementacija serijske verzije

Popularna igra *Pacman* za podlogu ima lavirint koji se u slučaju ovog projekta prikazuje kao 2D matrica nula i jedinica, od koje je korišćenjem OpenCV klase `cv::Mat` konstruisan objekat koji prikazuj ovu matricu tako da koristi ugrađeni tip podatka `CV_8UC3` koji predstavlja 8-bitne RGB piksele. Drugim rečima, na lavirintu koji je određen dimenzijom matrice moguće je iscrtati piksele bilo koje osmobitne RGB nijanse.

Kao početni korak program kreira ovu matricu i pozivom funkcije `create_map()` iscrtava na matrici crnim pikselima zidove lavirinta, i sivim prohodne hodnike. Pored toga, na matrici je potrebno prikazati pacmana kojim igrač kontroliše, kao i duhove tj. protivnike u igri. Igra je implementirana tako da na početku igre, nakon kreiranja crno-sive matrice, igra na nasumičnim mestima kreira pacmana označenog plavom bojom i potreban broj duhova roze boje. Ova funkcionalnost se postiže `create_object(int *target_x, int *target_y, int r, int g, int b)` funkcijom koja će sigurno kreirati traženi objekat na legalnom polju (na sivom polju, a ne na crnom) i kooridante toih objekta smestiti na memorijske lokacije na koje pokazuju pokazivači *target_x* i *target_y*. Trenutne pozicije pacmana i svih duhova se tokom cele igre čuvaju u globalnim promenjivima.

Biblioteka OpenCV nudi i ostale klase i metode klasa koji pomažu oko grafičkog interfejsa aplikacije, pa se pomoću `cv::namedWindow` kreira novi prozor u kojem će se prikazivati igra, sa `cv::resizeWindow` podesimo veličinu tog

prozora na optimalnu a pozivom `cv::imshow` prikazujemo matricu kao sliku u utvorenom prozoru.

Sve što je još potrebno za sekvecnijalnu verziju jeste napraviti mehanizam kretanja pacmana i duhova. Za kretanje igrača po mapi koristi se funkcija `move_pacman()` koja u sebi koristi OpenCV funkciju `cv::waitKey` koja kao parametar zahteva celobrojnu vrednost koji predstavlja broj milisekundi koliko će program da čeka na toj liniji koda da se desi pritisak nekog tastera na tastaturi. Ukoliko se dogodi pritisak ova funkcija vraća kod pritisnutog tastera, u suprotnom vraća -1 vrednost ukoliko tajmer istekne. Zatim je jednostavnim *switch* mehanizmom proverava koji je taster pritisnut i osvežavaju se koordinate pacmana. U sekvencijalnoj verziji ove igre delay je podešen na 300ms i on diktira brzinu igre i pomeranja duhova. Ukoliko svakih 300ms nije pritisnut taster, tajmer će isteći i duhovi će napraviti novi korak. Duhovi takođe prave korak svaki put kada se registruje taster i pacman napravi korak. Ovakva je implementacija je neophodna s obzirom da se igra odvija na jednoj niti i za pomeranje duhova nezavisno od pacmana i pritisnutog tastera je potrebno koristiti sistemski *interrupt* za tastere tastature. Očito je da je pristup implementiran u ovoj verziji igre daleko jednostavniji, tim više što će ovaj problem u logici biti rešen u implementaciji igre na više niti. Dodatno, ova funkcija osvežava stanje matrice tako što obilježava polja na kojima je pacman boravio. Kasnije će ova polja biti označena belom bojom od strane funkcije `update_maze()` tako da igrač može da razazna koji deo lavirinta je ostao neposećen.

Duhove pomera funkcija `move_ghosts(int)` koja komunicira sa matricom prepreka i za svaki prosleđeni redni broj duha (tj parametar i) osvežava poziciju tog duha shodno sa njegovim smerom kretanja. Početni smer kretanja je određen nasumično pri kreiranju svakog duha. Kada duh dođe do prepreke po trenutnom smeru kretanja, novi smer može da izabere nasumično ili determinsitički, zavisno od toga da li je u zaglavlju definisan makro `RANDOM_GHOSTS`.

Validnost igre se proverava pozivom funkcije `check_game()` koja proverava da li je došlo do poklapanja koordinata nekog od duhova i pacmana što rezultuje porazom, ili da li je pacman sakupio sve bodove (sva siva polja) što rezultuje pobedom.

3 Paralelizacija programa

Već razvijeni sekvencijalni program pretočen je u paralelnu verziju korišćenjem *OpenMP* biblioteke i paralelnih sekcija. Pomenute funkcije za upravljanje i nadgledanje igre su podaljene na posebne niti unutar paralelne sekcije, korišćenjem naredbe *section* označeno je da samo jedna od svih kreiranih niti u paralelnom delu može da izvršava tu sekciju.

S obzirom da se radi o igri u kojoj mora postojati komunikacija između određenih celina, vođeno je računa da što manje promenjivih bude deljeno između niti, da bude što manje kritičnih sekcija i da ostali podaci ili podaci čija se vrednost često menja budu privatni za niti kako ne bi došlo do konflikta. Na primer, promenjiva *game_over* sadrži informaciju da li je igra gotova ili ne,

nju setuje nit zadužena za izvršavanje provere validnosti igre. Kada se ova promenjiva promeni, sve ostale niti moraju znati da je igra gotova, s toga ovo je deljena promenjiva. Pored nje još su deljene koordinate pacmana i duhova, jer su ovi podaci potrebni istovremeno i funkciji za ispis igre na ekran i funkcijama za pomeranje objekata. Svakoj sekciji pridružen je naredni mehanizam: unutar lokalne while(true) petlje vrši se čitanje deljene promenjive *game_over* (ovaj dio je za svaki slučaj postavljen u kritičnu sekciju, u slučaju *read* hazarda), ukoliko je promenjiva setovana tako da označava da je igra gotova, naredbnom *break* svaka nit napušta svoju while petlju, u suprotnom svaka nit izvršava funkciju koja je njoj namenjena.

Za razliku od niti za kontrolu pacmana, iscrtavanje igre na ekranu i proveru igre, niti zadužene za pomeranje duhova su realizovane u tzv. *ugneždenom paralelizmu*. Pošto je potrebno da jedna nit kontroliše jednog duha, u opštem slučaju sa tri duha potrebne su nam dodatne tri niti koje će raditi u paraleli. Ugneždeni paralelizam predstavlja pozivanje OpenMP biblioteke da stvori novu paralelnu sekciju unutar već postojeće paralelne sekcije. Na većini računara ovo je po *defaultnoj* vrednosti zabranjeno, pa je potrebno pre izvršavanja u kodu pozvati `omp_set_nested(1)` funkciju koja *setuje* promenjivu okruženja i dozvoljava ugneždeni paralelizam.

Postoje dva pristupa rešenju ovog problema: imati ugneždeni paralelizam unutar sekcije, kada će jedna nit da uđe u sekciju i kreira nove tri niti koje će upravljati duhovima, u tom slučaju master nit neće više imati posla, ili realizovati paralelizam bez dodatne sekcije, kada će jedna od niti da stvori ostale dve i zatim će sve tri da upravljaju duhovima, s tim da su dve na drugom nivou paralelizma a master nit je na prvom. Drugi pristup je bolji jer ne kreira bespotrebnu nit koja će biti nezaposlena ostaka programa, te je on izabran. Realizacija pomeranja duhova je zatim identično realizovana kao ostale tri funkcije s tim da duhove pomeramo na svakih 200 miliskeundi, za šta se koristi funkcija `usleep`.

Rukovanje sa deljenim promenjivima poput koordinata pacmana je realizovano oprezno, tako što svaka funkcija koja koristi ove promenjive na početku unutar kritične sekcije u svoje lokalne kopije kopira vrednosti koordinata, manipuliše sa lokalnim kopijama za vreme izvršavanja i na kraju unutar kritične sekcije osveži globalnu vrednost sa lokalnim kopijama.

Po završetku igre izlazi se iz paralelnog regiona i master nit prikazuje sliku koja označava ili pobedu ili porez, i čeka da korisnik zatvori prozor.