Odabrana poglavlja iz operativnih sistema – projektni zadatak

Raspoređivač zadataka i paralelna obrada multimedijalnih podataka (50 bodova)

U proizvoljnom objektno-orijentisanom programskom jeziku realizovati raspoređivač zadataka koji ima mogućnost eksploatacije paralelizma na nivou procesorskih jezgara. Definisati programski API za raspoređivač i realizovati logički odvojenu GUI aplikaciju za raspoređivanje zadataka koja koristi definisani API. Kreirati tip zadatka koji vrši paralelnu obradu multimedijalnih podataka.

Napomene za izradu projektnog rada

Pravilno dokumentovati projekat. Program se mora moći kompajlirati, izvršiti i testirati. Uz priloženi rad treba da se obezbijede i primjeri koji se mogu iskoristiti za testiranje programa. Obezbijediti nekoliko jediničnih testova kojima se demonstriraju funkcionalnosti iz stavki u specifikaciji projektnog zadatka (nije potrebno za GUI aplikaciju i fajl-sistem). Pridržavati se:

- Principa objektno-orijentisanog programiranja i SOLID principa
- Principa pisanja čistog koda i pravilnog imenovanja varijabli, funkcija, klasa i metoda
- Konvencija za korišteni programski jezik

(25) Raspoređivač zadataka

(8) Pravilno realizovati osnovne funkcionalnosti za raspoređivač zadataka. Raspoređivač zadataka mora biti realizovan u sklopu biblioteke. Obezbijediti jednostavan API koji korisniku omogućava da vrši raspoređivanje zadataka. Omogućiti da se za raspoređivač specificira broj procesorskih jezgri na raspolaganju i maksimalan broj zadataka koji se konkurentno izvršavaju. Omogućiti da se za svaki zadatak specificira dozvoljeno ukupno vrijeme izvršavanja i rok do kojeg zadatak mora biti izvršen ili prekinut. Omogućiti da se za svaki zadatak specificira i dozvoljeni nivo paralelizma (npr. dozvoljen broj iskorištenih procesorskih jezgri). Dozvoljeno je da se definiše sopstveni API koji može biti korišten u implementaciji zadataka, kao način da se omoguće funkcionalnosti iz specifikacije projektnog zadatka. Dozvoljeno je i da se izvršavanje svakog zadataka pokreće kao zaseban proces. Obezbijediti jednostavne jedinične testove koji demonstriraju rad raspoređivača.

(2) Omogućiti zaustavljanje, pauziranje i nastavak zadataka.

(2) Omogućiti serijalizaciju i deserijalizaciju zadataka. Serijalizovanja polja mogu da uključuju, kao primjer, niz putanja na ulazne fajlove i putanju gdje treba da se generišu izlazni fajlovi.

(2) Omogućiti raspoređivanje u realnom vremenu, tj. raspoređivanje zadataka za vrijeme dok raspoređivač izvršava već raspoređene funkcije.

(2) Omogućiti prioritetno raspoređivanje, tako da se za svaki zadatak može podesiti prioritet.

(4) Implementirati preventivno raspoređivanje sa prioritetima. Po potrebi koristiti mehanizme kooperativnog raspoređivanja u cilju postizanja efekata preventivnog raspoređivanja.

(5) Proširiti funkcionalnosti preventivnog raspoređivanja sa prioritetima, tako da zadaci mogu da raspolažu sa resursima. Zadatak može da zaključa resurs, tako da ga drugi zadaci ne mogu upotrebljavati. Fajlovi i folderi se mogu tretirati kao resursi. Obezbijediti mehanizme za sinhronizaciju između zadataka i pristup dijeljenim resursima. Obezbijediti mehanizme za prevenciju ili detekciju i razriješavanje *deadlock* situacija. Riješiti problem inverzije prioriteta (PIP ili PCP algoritam). Jediničnim testovima demonstrirati i razriješavanje osnovne *deadlock* situacije, kao i pristup dijeljenim resursima sa rješavanjem problema inverzije prioriteta.

(10) GUI aplikacija za raspoređivanje zadataka

(6) Obezbijediti aplikaciju sa neblokirajućim grafičkim korisničkim interfejsom za raspoređivanje zadataka. Konstruisati aplikaciju tako da se može lako proširiti novim tipovima zadataka. Omogućiti korisniku aplikacije da specificira način raspoređivanja po pitanju prioriteta i prevencije. Omogućiti korisniku aplikacije da specificira sva svojstva kreiranog zadatka (dozvoljeno vrijeme izvršavanja, rokove, itd.). GUI aplikacija treba da pruži podršku za pregled zadataka u toku izvršavanja (uz prijavu toka obavljanja zadataka), kao i za kreiranje, započinjanje i uklanjanje zadataka.

(2) Omogućiti zaustavljanje, pauziranje i nastavak zadataka kroz GUI aplikaciju.

(2) Omogućiti da se interno stanje aplikacije može serijalizovati (broj zadataka i stanje obrade). Omogućiti da se, u slučaju bilo kakvog zaustavljanja aplikacije (uključujući i neočekivano zaustavljanje), zadaci mogu nastaviti izvršavati (npr. *autosave* mehanizam) od tačke u kojoj su se prethodno zaustavili.

(10) Obrada multimedijalnih podataka

(5) Definisati tip zadatka za netrivijalnu obradu multimedijalnih fajlova. Konkretan algoritam odabrati na kursu predmeta. U GUI aplikaciji obezbijediti podršku za rad sa zadacima definisanog tipa. Zadatak definisanog tipa treba da zadovoljava sljedeća ograničenja:

- Zadatak se može rasporediti na izvršavanje upotrebom raspoređivača, tj. koristi API za zadatke koji je obezbijeđen od strane raspoređivača
- Zadatak može da vrši paralelnu obradu nad većom količinom ulaznih multimedijalnih fajlova

(5) Realizovati zadatak koji dodatno može da vrši paralelnu obradu i nad samo jednim multimedijalnim fajlom. Pri tome se treba ostvariti ubrzanje (testirati nad multimedijalnim fajlom adekvatne veličine).

(5) Fajl-sistem u korisničkom prostoru

(5) Korištenjem Dokan ili FUSE biblioteke, napisati drajver za fajl-sistem u korisničkom prostoru. Fajl-sistem treba da koristi API raspoređivača i da omogući izvršavanje bar jednog tipa zadatka za obradu multimedijalnih podataka, na sljedeći način:

- Fajl-sistem treba da sadrži folder *input* u koji se mogu kopirati multimedijalni fajlovi
- Nakon što su fajlovi kopirani u folder *input*, započinje obrada tih fajlova
- Nakon što završi obrada fajlova, rezultati treba da budu dostupni u folderu *output*