Sveučilište u Splitu

Prirodoslovno-matematički fakultet

Završni preddiplomski rad

METODE OBRADE PODATAKA U RAČUNALNIM SUSTAVIMA

Student: Zoran Stipinović

Mentor: prof. dr. sc. Slavomir Stankov

Neposredni voditelj: mr. sc. Ani Grubišić, prof.

Split, listopad 2011.

Sadržaj

1 Uvod

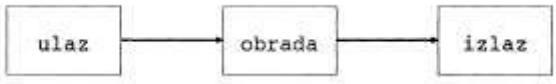
Danas je rad bez računala gotovo nezamisliv. Kućanstva, škole, tvrtke sve više ovise o uporabi računala u rješavanju svakodnevnih zadataka, koji bi bez pomoći računala zahtijevali puno više vremena i bili puno skuplji za izvođenje. U ovom radu ćemo se pozabaviti načinima na koje računalo obrađuje podatke, te nam time uvelike olakšava život. Prvo ćemo se upoznati s definicijom podatka.

Podaci su skup činjenica koji mogu biti

- brojčani, npr. brojevi telefona
- nebrojčani, npr. imena i prezimena
- kombinacije brojčanih i nebrojčanih činjenica, npr. registarski brojevi vozila.

Općenito, obrada podataka jest manipuliranje podacima radi dobivanja uporabljivih oblika. Obrada podataka obuhvaća ne samo brojčana izračunavanja već i postupke kao što su klasifikacija podataka ili njihovo premještanje s jednog mjesta na drugo. Općenito se pretpostavlja da takve operacije obavlja neka vrsta stroja (računalo), bez obzira na to što neki postupci mogu biti obavljeni ručno.

Obrada podataka sastoji se od tri temeljna koraka: ulaza, obrade i izlaza. Ta tri koraka međusobno su povezana kao što je prikazano na slici 1 i čine ciklus obrade podataka (Tuđman, Boras, Dovedan,1991.).



Slika 1 Jednostavan princip obrade podataka (Tuđman, Boras, Dovedan, 1991.).

Upoznat ćemo neke od standardnih metoda obrade podataka u računalnim sustavima.

U drugom poglavlju ćemo obraditi kriterije po kojima odabiremo željenu metodu obrade podataka.

Treće poglavlje govori o serijskoj (skupnoj) obradi podataka. Proći ćemo kratku povijest, čestu uporabu i prednostima serijske obrade.

Tema četvrtog poglavlja je obrada u stvarnom vremenu (real-time obrada).

Peto poglavlje sadrži obradu s podjelom vremena.

Šesto poglavlje se bavi višezadaćnošću (multitasking).

Sedmo poglavlje višeprogramiranje (multiprogramming).

Tipovi multiprocesorskih sustava te način na koji rade su opisani u sedmom poglavlju.

Osmo poglavlje nam opisuje teleprocessing metodu obrade podataka.

2 Klasifikacija metoda obrade podataka

Nema jedinstvene klasifikacije metoda obrade podataka. Metode se ocjenjuju prema više kriterija, od kojih većina tih kriterija ima dvije međusobno isključive vrijednosti. Pošto imamo veliki broj kombinacija vrijednosti tih kriterija, tako imamo i mnogo različitih metoda obrade podataka. Najčešći kriteriji za ocjenu metode obrade podataka su prikazani u tablici 1.

Kriteriji	**I način**	**II način**
Broj programa	Jedan	Više
Količina podataka	Mala	Velika
Komunikacija s korisnikom	Posredna	Neposredna
Vrijeme odgovora	Kritično	Nije kritično
Broj procesora	Jedan	Više
Umreženost	Da	Ne
Arhitektura	Monolitna	Razdijeljena
Organizacija	Centralizirana	Decentralizirana
Udaljenost	Mala	Velika
Elektronska povezanost	Da	Ne

Tablica 1 Kriteriji za vrednovanje metode obrade podataka (Bosilj Vukšić i Pejić Bach, 2009)

U praksi imamo tek nekoliko metoda, koje predstavljaju različite kombinacije vrijednosti kriterija. Većinom se odabire jedan kriterij prema kojem se određuje metoda, a vrijednosti ostalih kriterija se podrazumijevaju uz tu metodu ili se smatraju nevažnim. Tradicionalne metode obrade podataka su:

- serijska obrada (eng. batch processing)
- obrada u stvarnom vremenu (eng. real-time processing)
- obrada s podjelom vremena (eng. time-sharing processing)
- višezadaćna obrada (eng. multitasking)
- višeprogramska obrada (eng. multiprogramming)
- simultana obrada (eng. multiprocessing)
- daljinska obrada (eng. teleprocessing)

Tekst koji slijedi napisan je na temelju nekoliko izvora koji su najčešće korišteni pa se neće posebno na njih referencirati. Ti izvori su: Batch processing, Real-time computing, Time-sharing, Computer multitasking, Multiprogramming, Multiprocessing (http://en.wikipedia.org/wiki/) i skripta “Poslovna informatika” (Bosilj Vukšić i Pejić Bach, 2009).

3 Serijska obrada

Obrada podataka ili izvršavanje poslova, koji su prikupljeni u skupine unaprijed, tako da korisnik više ne može utjecati na njihovu obradu se zove serijska obrada. Naziv „batch“ dolazi iz toga što su podaci prikupljeni u skupine („batcheve“) datoteka te se u njima obrađuju. Skupine zadataka se prikupljaju, unose, obrađuju te se proizvode rezultati serijske obrade. Serijska obrada zahtjeva zasebne programe za unos, obradu i izlaz. Ova metoda je učinkovita za obradu velike količine podataka.

3.1 Povijest

Od najranijih dana elektroničkih računala serijska obrada je povezana s mainframe računalima 1950-ih godina. Dominirala je početkom računarstva jer su najvažniji poslovni problemi (najprofitabilniji) bili većinom računovodstveni problemi, poput naplate. Naplata je serijski-orijentiran proces (obrađuju se cijene usluga te se na kraju ispisuju na račun), a očito je svakom poslu potreban račun. Računalni resursi su tada bili vrlo skupi, stoga je serijska obrada odgovarala ograničenju resursa.

Danas su gotovo sva računala sposobna izvoditi neku vrstu serijske obrade (Unix računala, Microsoft Windows, Mac OS X, pa čak i neki pametni telefoni). Skeniranje virusa te zakazani poslovi koji brišu privremene datoteke su također vrste serijske obrade.

Serijska obrada je još uvijek važna u većini organizacija, jer su mnogi temeljni poslovi serijski-orijentirani. Novi sustavi sadrže jednu ili više aplikacija sa serijskom obradom za ažuriranje podataka na kraju dana, generiranje izvješća, tiskanje dokumenata i drugih ne-interaktivnih zadataka koji se moraju izvršiti pouzdano i u roku.

Neke od najčešćih upotreba serijske obrade podataka su :

Obrada podataka

Najčešće korištenje serijske obrade je generiranje izvješća na kraju radnog dana. Na primjer, u banci obračun kamata, generiranje izvješća i podataka za druge sustave, ispis (izjava).

Ispis

Ispisivanje je računalna procedura serijske obrade koja obično uključuje operatera koji odabire dokumente za ispis te ukazuje softveru kada i gdje treba ispisati dokumente, te njihov prioritet ispisa. Zatim se program poslan na red čekanja za ispis, odakle se šalje na pisač.

Baze podataka

Serijska obrada se također koristi za ažuriranje skupnih baza podataka i automatsku obradu transakcija.

Slike

Često se za obavljanje različitih operacija s digitalnim slikama koristi serijska obrada. Postoje računalni programi koji omogućuju promjenu veličine, konvertiranje ili uređivanje slikovne datoteke na neki drugi način.

Konvertiranje

Ovu metodu obrade podataka također koristimo i za prebacivanje datoteka iz jednog u drugi format. Datoteke postaju prenosive i svestrane.

3.2 Prednosti i nedostaci

Prednosti:

- Omogućuje dijeljenje računalnih resursa između više korisnika i programa.
- Pomiče vrijeme obrade „posla“ kada su računalni resursi manje opterećeni.
- Onemogućuje prazan hod računalnih resursa uz pomoć nadzora svake minute.
- Povećana je produktivnost.

Nedostaci:

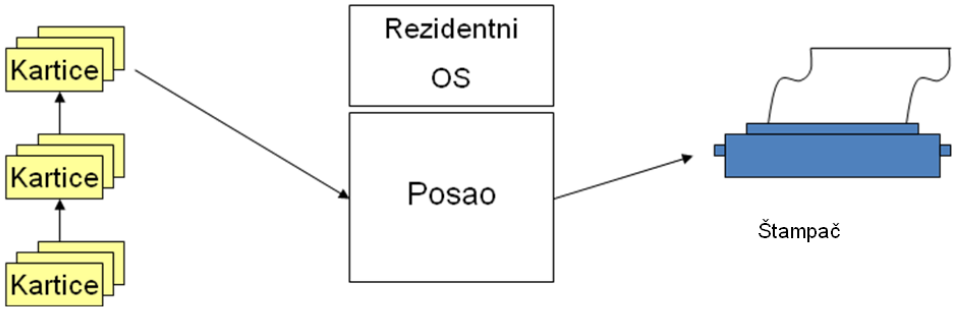
- Dugo vrijeme prikupljanja i kompletiranja ulaznih podataka
- Relativno dugo vrijeme od unosa podataka do ispisa rezultata
- Često dovodi do zagušenja u manipuliranju velikim brojem podataka

3.3 Primjer

Primjer jednostavnog „batch“ sustava je prikazan na Slici 2:

- Predaja bušenih kartica
- “Poslovi” (jobs) čekaju na stolu
- Svrstavanje poslova prema srodnosti, tj. I/O operacijama i resursima koji se za njih montiraju
- Postavljanje posla u čitač kartica
- Čitanje kartica: 300-1200 u minuti (20 u sekundi)

- Obavljanje posla: 20.000 instrukcija u sekundi (1000 puta brže nego priprema čitanjem kartica)
- Tiskanje rezultata: 120 linija u minuti (2 linije x 132 znaka u sekundi)
- Poruka o grešci na papiru



Slika 2 Primjer jednostavnog batch sustava (T. Dadić,2011)

4 Obrada u stvarnom vremenu

Obrada u stvarnom vremenu (još se naziva reaktivna) se koristi u sustavima koji imaju ograničenje u „stvarnom vremenu“. Termin „stvarno vrijeme“ znači izvršavanje zadatka bez vidljivog kašnjenja odziva sustava. Imamo kontinuirani ulaz, obradu i izlaz podataka. Programi u stvarnom vremenu moraju garantirati odziv unutar strogog vremenskog ograničenja (milisekunde, mikrosekunde), inače nastaju problemi za sustav (Ben-Ari, 1990). Sustav nije u stvarnom vremenu ako ne može jamčiti vrijeme odziva u svakoj situaciji.

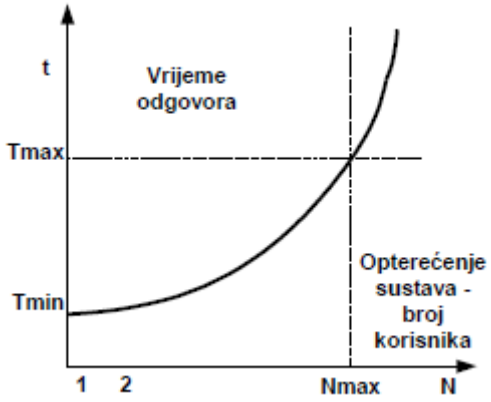
U real-time sustavima, neki zadaci koji čekaju su zajamčeni dobiti CPU kada se dogodi vanjski događaj. Real-time sustavi su dizajnirani za kontrolu mehaničkih uređaja kao što su industrijski roboti, koji zahtijevaju pravovremenu obradu.

Kriteriji za sustave u stvarnom vremenu su puno stroži nego kod drugih sustava. Osnovna svojstva tih sustava trebaju biti :

- Predvidljivo ponašanje
- Pouzdanost
- Stabilnost
- Jednostavnost

Obrada u stvarnom vremenu se nekada protumači kao obrada visokih performansi, npr. superračunalo izvršavajući znanstvenu simulaciju može ponuditi impresivne performanse, ali ne izvršava real-time obradu. Vrijeme odziva mrežnog servera može biti sporije kada je pod velikim opterećenjem, ali će (u većini slučajeva) odgovoriti prije roka. Takav mrežni server se ne smatra real-time sustavom jer su vremenske pogreške (kašnjenja,time-out itd.) obično male. Predvidivost, a ne performansa, je najvažniji uvjet real-time sustava.

Na slici 3 prikazana je eksponencijalna ovisnost vremena odgovora o opterećenju računalnog sustava.



Slika 3 Ovisnost vremena odgovora o opterećenju sustava (Bosilj Vukšić i Pejić Bach, 2009)

Za sustav se kaže da je u stvarnom vremenu ako ispravnost operacije ne ovisi samo o logičkoj ispravnosti već i o vremenu u kojem je izvršena. Klasifikacija sustava u stvarnom vremenu (prema posljedicama kašnjenja roka):

- težak (hard) : Kašnjenje roka je neuspjeh sustava.
- čvrst (firm) : Rijetko kašnjenje rokova je podnošljivo, ali može smanjiti kvalitetu usluge sustava. Ako se promaši rok, korisnosti rezultata nema.
- mekan (soft) : Korisnost rezultata se smanjuje nakon roka, a time se smanjuje i kvaliteta usluge sustava.

Cilj teškog sustava u stvarnom vremenu je osigurati da su svi rokovi ispunjeni. Ova vrsta obrade se nalazi u sustavima koji su veoma osjetljivi na pogreške, npr. sustavi kontrole leta i mnogi medicinski uređaji.

Cilj mekog sustava u stvarnom vremenu je ispunjenje određenog podskupa rokova kako bi se optimizirao kriterij neke aplikacije. Ova vrsta obrade se često koristi u audio i video sustavima (snimljeni video može propustiti okvire, ali sustav neće zakazati zbog kašnjenja ili

brisanja okvira). Umjesto toga kvaliteta videa je smanjena. Meki real-time sustavi se također koriste gdje postoje problemi istovremenih pristupa i postoji potreba da se broj priključenih sustava ažurira s promjenom situacije.

U nekim slučajevima sustav ima slabiju kvalitetu ili smanjenu učinkovitost. U kontekstu višezadaćnosti raspored sustava je obično napravljen prema prioritetu (Liu i Layland, 1973).

4.1 Povijest

Pojam „stvarno vrijeme“ je prvotno upotrebljavan u ranim simulacijama. Analogna računala, najčešće, su bila sposobna simulirati mnogo brže nego računala u stvarnom vremenu. Prvo takvo računalo je napravljeno na MIT-u 1955. godine.

4.2 Prednosti i nedostaci

Prednosti:

- Sustav se odmah ažurira
- U većini slučajeva sustav neće uzrokovati kašnjenje procesa

Nedostaci:

- Sustav uvijek mora biti online
- Može biti stresno za procesor

4.3 Primjer

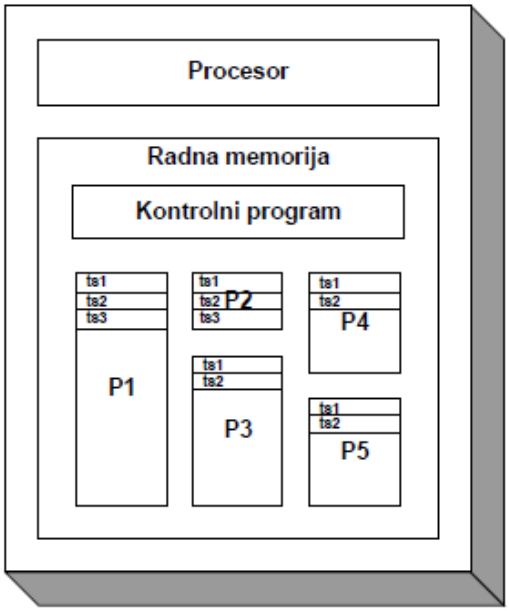
Neke vrste programske podrške, kao što su mnogi šah programi, mogu upasti u bilo koju kategoriju. Npr. program šaha dizajniran da se igra na turniru sa satom će morati odlučiti o pokretu prije određenog roka ili će izgubiti, te stoga spada u obradu u stvarnom vremenu, ali program šaha kojemu je dopušteno da radi neodređeno vrijeme prije poteza nije. U oba slučaja, visoka performansa je poželjna: što više operacija program šaha za turnir obavi u predviđenom vremenu, to će potezi biti bolji i što brže radi slobodan šahovski program to će prije napraviti potez. Ovaj primjer također prikazuje bitnu razliku između obrade u stvarnom vremenu i drugih obrada: ako turnirski program ne obavi potez u roku gubi igru (neuspješna obrada) dok u drugom scenariju, držanje roka nije potrebno.

5 Obrada s podjelom vremena

Podjela vremena je tehnika zakazivanja vremena računala da se dijeli između više zadataka i korisnika tako da svaki korisnik ima privid da se njegov proračun provodi bez prekida. Budući da je središnji procesor puno brži nego većina periferne opreme (npr. video display, pisač) ima dovoljno vremena riješiti nekoliko diskretnih problema tijekom procesa ulaza/izlaza. Pristup i preuzimanje iz sustava s podjelom vremena se čini trenutačno, iako procesor rješava probleme u sekvencama.

Zadatak koji se izvršava se mora odreći CPU-a, bilo dobrovoljno bilo radi vanjskog događaja kao što je prekid tehničke podrške. Sustavi s podjelom vremena su dizajnirani kako bi se dopustilo nekoliko programa da se izvršavaju naizgled istovremeno.

Uvođenje podjele vremena 1960-ih predstavlja veliki tehnološki pomak. Omogućujući pojedincima i organizacijama korištenje računala bez da ga posjeduju, dramatično je smanjen trošak pružanja računalnih mogućnosti te se promicala interaktivna uporaba računala i razvoj novih aplikacija.



Slika 4 Koncept multiprogramske obrade s podjelom vremena (Bosilj Vukšić i Pejić Bach, 2009)

Na slici 4 prikazan je koncept ovog načina obrade podataka gdje je više programa aktivno u radnoj memoriji, a kontrolni program svakom korisničkom programu naizmjenično dodjeljuje kratki vremenski interval za korištenje procesora. Po isteku intervala slijedeći program zauzima procesor i tako u krug. Na taj način će kraći programi koji zahtijevaju manje vremena biti prije izvršeni. Time se osigurava dobra iskoristivost računalnih resursa, ali i prihvatljiva vremena odgovora za brojne korisnike.

5.1 Povijest

Podjela vremena je razvijena zbog toga što je jedan korisnik neučinkovit, dok velika skupina korisnika nije. U većini slučajeva korisnici unose informacije te slijedi duga pauza, no kada skupina korisnika radi istodobno, pauza jednog korisnika se koristi za aktivnosti drugih. Cijeli proces može biti veoma efikasan, ako se optimizira veličina skupine. Vrijeme potrošeno na čekanje diska, trake ili mrežnog ulaza može se dati drugim korisnicima.

Razviti sustav koji podržava više korisnika u isto vrijeme je bilo teško. „Stanje“ svakog korisnika i njegovih programa treba biti čuvano i brzo se treba mijenjati između njih. Obrada podjelom vremena je postajala sve bolji izbor kako su računala postajala sve brža.

Bob Berner je prvi javno opisao koncept u „Automatic Control Magazine“ 1957. godine. Prvi projekt koji implementira time-sharing sustav je pokrenuo John McCarthy krajem 1957., na modificiranom IBM 704, a kasnije i na dodatno modificiranom IBM 7090 računalu. Iako je otišao raditi na MAC i drugim projektima, jedan od rezultata projekta, poznat kao Kompatibilan Time-Sharing Sustav (CTSS), je demonstriran 1961. CTSS je vjerojatno prvi time-sharing sustav i ostao je u uporabi sve do 1973. Drugi kandidat za prvi demonstrirani time-sharing sustav je PLATO II, kojeg je izradio Donald Bitzer u javnoj demonstraciji u parku Roberta Allertona u blizini sveučilišta Illinois rane 1961. Prvi komercijalno uspješan time-sharing sustav je bio Darthmouth sustav s podjelom vremena.

1960-ih su tvrtke počele pružanje usluga podjele vremena, koristeći Teletype model 33 KSR ili ASR ili model 35 KSR ili ASR u ASCII okruženjima, i IBM Selectric pisaće. Povezali bi se na središnje računalo preko dial-up modema ili akustički povezanih modema. Sustav je osiguravao kompletno operacijsko okruženje, uključujući i raznovrsne programibilne jezične procesore, razne softverske pakete, pohranu podataka, ispis i off-line pohranu. Korisnici su plaćali najam za terminale, naknadu za sate provedene spojeni, naknadu za sekunde vremena CPU-a i naknadu za pohranu kilobajta po mjesecu.

Tijekom kasnih 1960-ih i 1970-ih, računalni terminali su postavljeni na velika institucionalna mainframe računala (središnji računalni sustavi). Obično su računalni terminali korišteni na području fakulteta na istim mjestima kao i stolna ili osobna računala. U najranijim danima upotrebe osobnih računala, mnogi su koristili pametne terminale sa time-sharing sustavima.

Podjela vremena je nestala s razvojem mikroprogramiranja ranih 1980-ih jer su mikroprocesori dovoljno jeftini da jedna osoba može imat vrijeme CPU-a posvećeno samo sebi, čak i kada miruje. Internet je opet popularizirao koncept podjele vremena, pošto skupi serveri mogu poslužiti tisuće korisnika odjednom. Web stranice rade na mahove aktivnosti nakon kojih slijedi razdoblje mirovanja. Stoga mnogi korisnici mogu koristiti web stranice odjednom i nitko neće primijetiti kašnjenje dok se serveri ne preopterete.

Kod sustava sa serijskom obradom sigurnost nije bila veliki problem, pa ništa više od korisničkog imena kao mjera sigurnosti nije bilo potrebno. Korisnici sustava sa podjelom vremena su zahtijevali puno veću zaštitu, pa je prva međunarodna konferencija o računalnoj sigurnosti koja se održala 1971. u Londonu inicirana od strane korisnika podjele vremena i industrije. Rane zajednice hakera skupljale su se oko mjesta gdje je pristup korisničkom računu bio lagan. Nisu bile raspodijeljene po mrežama, nego fizičke. Hakeri su morali dijeliti stroj, te bi pad sustava poremetio desetke programera koji sjede u istoj terminalnoj sobi.

5.2 Prednosti i nedostaci

Prednosti:

- Procesor se uvijek koristi
- Više ljudi se poslužuje „odjednom“

Nedostaci:

- Postoji vrijeme kašnjenja između korisnika, jer su svi izlazni uređaji spojeni na jedno računalo
- Proces može zauzeti vrijeme drugim procesima, te se usporava odziv drugim korisnicima

5.3 Primjer

Primjer izvođenja zadataka različitih korisnika dijeljenjem vremena (Slika 5):

Korisnik 1	**Izvodi**			**Izvodi**		
Korisnik 2		**Izvodi**			**Gotov**	
Korisnik 3			**Izvodi**			**Izvodi**
O ms	10 ms	20 ms	30 ms	40 ms	50 ms	56 s

Slika 5 Time sharing primjer (Dadić,2011)

- Svaki korisnik ima na raspolaganju sustav jedan vremenski odsječak, npr. 10 ms
- Kada istekne vrijeme od 10 ms, timer prekida glavni procesor
- Stanje CPU registara se sprema u Tablicu procesa
- U CPU registre se učitavaju vrijednosti procesa koji je na redu izvođenja

6 Višezadaćna obrada

Višezadaćna obrada je metoda gdje više zadataka (procesa) dijele zajedničke resurse kao što je CPU. Kada imamo računalo sa jednim CPU u trenutku vremena se može izvršavati samo jedan zadatak. Problem se rješava raspoređujući koji se zadatak može izvršavati u nekom trenutku i kada drugi zadatak dolazi na red. Preraspodjela CPU-a s jednog zadatka na drugi se naziva „context switch“ (prebacivanje konteksta), te se tako postiže iluzija paralelnog rada. Višezadaćna obrada omogućava izvođenje mnogo više zadataka čak i kod višeprocesorskih računala. To ne znači da se neograničen broj zadataka može izvršavati u isto vrijeme. Svaki zadatak troši resurse i prostor za pohranjivanje. Što se više zadataka obavlja, sustav može usporiti ili ostati bez prostora za pohranu.

Dio vremena u kojem se dopušta izvođenje procesora se zove „time slice“ (kriška vremena). Za svaku krišku vremena, planer se pokreće jednom da odabere koji će se sljedeći proces izvršiti. Operacijski sustav vrši prebacivanje između procesa kada njihova kriška vremena istekne pomoću prekida. Tako se procesorsko vrijeme efektivno dijeli, stvarajući iluziju da se zadaci izvršavaju istovremeno.

Procese koji su u cijelosti neovisni nije teško programirati. Složenost višezadaćnog sustava leži u dijeljenju računalnih resursa između zadataka i usklađivanju rada kooperativnih zadataka.

Veći sustavi su ponekad građeni s centralnim procesorom (procesorima) i nekoliko U/I procesora, vrsta asimetričnih multi-obrada.

Tijekom godina višezadaćni sustavi su poboljšani, te moderni sustavi obično uključuju detaljne mehanizme koji postavljaju prioritete procesa.

Kooperativna višezadaćnost

Rani višezadaćni sustavi koristili su metodu gdje su aplikacije dobrovoljno ustupale vrijeme jedna drugoj. Ta metoda se zove kooperativna višezadaćnost.

Jer kooperativni sustav ovisi o tome da programi redovito ustupaju vrijeme procesora, jedan loši program može zauzeti cijelo vrijeme CPU-a ili prouzročiti zastoj sustava. Zbog toga se cijeli softver mora evaluirati i odobriti za korištenje prije nego se instalira na glavni server ili će program koji ne funkcionira ispravno usporiti ili zamrznuti cijelu mrežu.

Preventivna višezadaćnost

Privremeni prekid zadatka koji se izvršava, bez njegove suradnje, te s namjerom da se zadatak nastavi kasnije zove se preventivnost. Prebacivanje konteksta provodi povlašteni zadatak ili preventivni planer.

Preventivna višezadaćnost omogućuje računalnom sustavu da pouzdanije jamči svakom procesu redovne dijelove operativnog vremena. Također omogućuje sustavu brzo bavljenje važnim vanjskim događajima kao što su ulazni podaci. Operacijski sustavi su razvijeni kako bi se iskoristile ove sposobnosti tehničke podrške, npr. ova metoda se implementirala u najranijoj verziji Unix-a 1969., te je standard u Unix-u (DRI, 2011).

Procesi koji se izvode su podijeljeni u dvije kategorije: oni koji čekaju na ulaz ili izlaz (U/I vezani procesi) i oni koji u potpunosti koriste CPU (CPU vezani procesori). U/I vezani procesu se „blokiraju“ (stavljaju na čekanje) do dolaska potrebnih podataka, te se dopušta drugim procesima korištenje CPU-a. Dolazak traženog podatka generira prekid, te se blokirani procesi vraćaju na izvršavanje.

Višenitnost

Niti proizlaze iz ideje da je najučinkovitiji način za kooperativne procese da razmjene podatke da dijele svoj memorijski prostor. Niti su procesi koji se izvode u kontekstu iste memorije, te su klasificirane kao lagane jer prebacivanje između niti ne zahtijeva prebacivanje memorijskog konteksta.

Višenitnost je sposobnost programa ili procesa da ga koriste više od jednog korisnika bez da se više kopija tog programa izvršava u računalu. Svaki korisnikov zahtjev za programom se vodi kao nit sa zasebnim identitetom, te se status niti čuva dok se ona ne izvrši (unatoč prekidima drugih programa).

Niti su planirane preventivno, a ako su planirane kooperativno zovemo ih vlaknima. Vlakna su lakša od niti i lakše ih je programirati. Neki sustavi izravno podupiru višenitnost u tehničkoj podršci.

Zaštita memorije

Program koji se loše ponaša (nehotice ili namjerno) može prebrisati memoriju koja pripada drugom programu, pa čak i operativnom sustavu. Operativni sistem stoga ograničava dostupnost memorije programu koji se izvršava, te se program koji pokušava pristupiti memoriji izvan dopuštenog raspona odmah zaustavlja prije nego što može promijeniti memoriju koja pripada drugim procesima.

Još jedna ključna inovacija je ideja privilegiranih razina. Zadacima niske privilegije nije dopušten pristup nekoj vrsti memorije i ne smiju obavljati određene instrukcije. Kada

zadatak pokuša izvesti privilegiranu operaciju upada u zamku i nadzorni program odlučuje kako će reagirati.

Zamjena memorije

Zamjenom datoteke ili particije operativni sustav osigurava više memorije nego je fizički dostupno tako što drži dijelove primarne memorije u sekundarnoj. Pošto zamjena memorije omogućuje da se više zadataka učita u isto vrijeme, često se koristi u kombinaciji sa višezadaćnošću. Višezadaćni sistem dopušta drugom procesu izvođenje kada proces koji se trenutno izvodi dođe do situacije kada treba čekati neki dio da se učita iz sekundarne memorije.

6.1 Povijest

Kada se korištenje računala promijenilo iz serijske obrade u interaktivni način rada, svaki korisnik je htio vidjeti izvršavanje svog programa kao da je jedini u računalu. Korištenje podjele vremena je to omogućilo.

Rani višezadaćni sustavi koristili su pristup poznat kao kooperativna višezadaćnost koja se danas još koristi u RISC operativnim sustavima.

Najraniji operacijski sustav sa preventivnom višezadaćnošću dostupan kućnim korisnicima je Sinclair QDOS na Sinclair QL, objavljen 1984. (Robert Klein,1998.,1999.), ali je vrlo malo ljudi kupilo stroj. Commodore-ov Amiga, objavljen sljedeće godine je bio prvi komercijalno uspješno kućno računalo koje je koristilo ovu tehnologiju, i njegove multimedijske mogućnosti čine ga pretečom suvremenih osobnih računala.

Iako se sada rijetko koristi u većim sustavima, kooperativna višezadaćnost se koristila od strane Microsoft Windowsa (prije Windows 95 i Windows NT) i Mac OS-a (prije Mac OS X). Mrežni operacijski sustav NetWare koristio je kooperativnu višezadaćnost do NetWare 6.5. Microsoft je preventivnu višezadaćnost učinio glavnom značajkom njihovih operativnih sustava ranih 90-ih kada su razvijali Windows NT 3.1. te poslije Windows 95. Kasnije ju je koristio i Apple u Mac OS 9.x.

6.2 Prednosti i nedostaci

Prednosti:

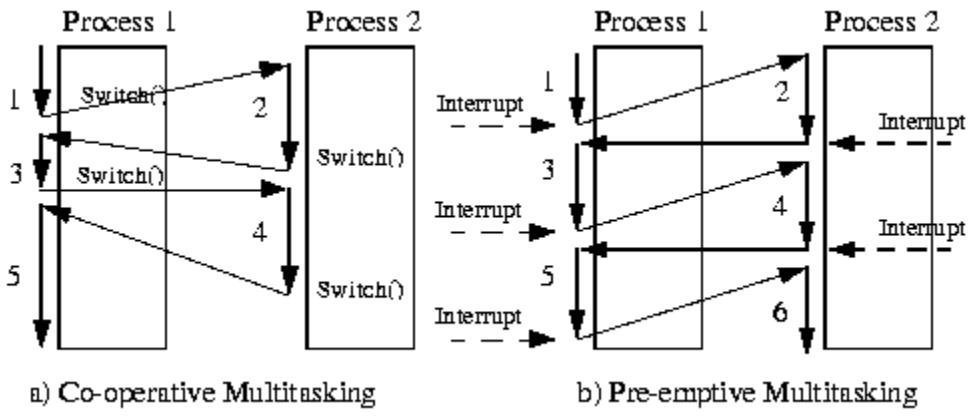
- Velika produktivnost, jer više programa može biti upaljeno
- Podaci se mogu lako kopirati između programa
- Dizajner aplikacija može napraviti aplikacije koje koriste više procesa
- Programi koji se ažuriraju se mogu odmah vidjeti,npr. ako primimo novi e-mail

Nedostaci:

- Zahtijeva više resursa
- CPU treba biti brz
- Ako se puno zadataka izvršava odjednom usporava se rad i nekada može stati

6.3 Primjer

Primjer na slici 6 pokazuje razliku između kooperativnog i preventivnog višezadaćnog sustava. U kooperativnom sustavu sami zadaci pozivaju funkciju (Switch() u ovom slučaju) te predaju kontrolu nad procesorom. Kod preventivnih sustava tajmer za prekide uzrokuje promjenu zadataka. Kao što je prikazano vrijeme procesora je jednako podijeljeno kod preventivnih višezadaćnih sustava (Ojala, 1996).



Slika 6 Kooperativna i preventivna višezadaćnost (preuzeto od Ojala, 1996)

7 Višeprogramska obrada

U ranim danima računarstva, CPU vrijeme je bilo skupo i periferni uređaji su bili vrlo spori. Kada se pokretao program koji treba pristup perifernom uređaju, CPU bi trebao zaustaviti izvršavanje instrukcija, što je vrlo neučinkovito.

U sustavima sa višeprogramiranjem, zadatak se izvršava sve dok ne izvrši operaciju koja zahtijeva čekanje nekog vanjskog događaja (npr. čitanje s trake) ili dok raspored računala prisilno zamijeni zadatak van CPU-a. Sustavi s višeprogramiranjem su dizajnirani kako bi se povećala iskoristivost procesora.

Višeprogramska obrada je raspodjela računalnog sustava i njegovih resursa na više od jedne istovremenih aplikacija, poslova ili korisnika („programa“ u ovom slučaju). Pošto se koristi samo jedan procesor, nema prave simultane obrade.

Višeprogramska obrada ne podrazumijeva (ili nema sposobnost) prave višezadaćnosti, ali višezadaćna obrada uglavnom implicira korištenje nekih metoda višeprogramiranja.

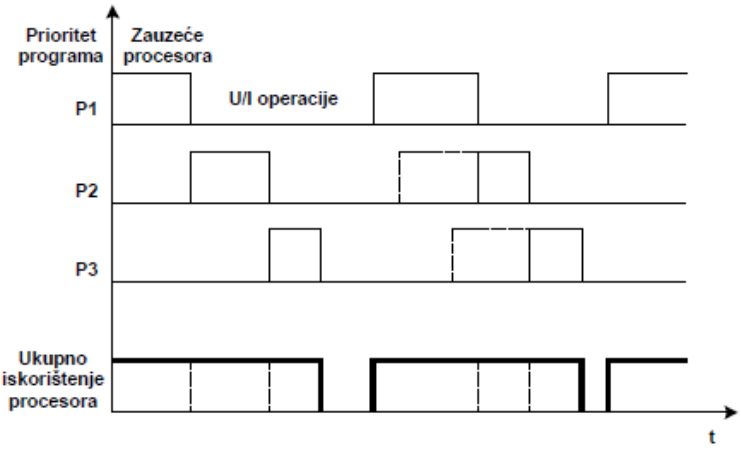
U ovome kontekstu riječ program se ne mora nužno odnositi na sastavljenu aplikaciju, već na bilo koji skup naredbi koje korisnik šalje na izvršavanje. Cijeli interaktivni proces možemo nazvati programom.

Program obuhvaća brojne zadatke, a zadaci su mala skupina procesorskih instrukcija. Često zadaci završavaju s zahtjevom premještanja podataka, što se može iskoristiti za dopuštanje korištenja resursa drugom programu.

Istodoban rad se postiže kada operacijski sustav iskoristi mogućnost prekida jednog programa (dok čeka za unos/ispis) te prebacuje kontrolu na procesima drugom programu (aplikacija, posao ili korisnik). Prema dizajnu programa koji se izvršavaju (koliko se često mogu prekidati) ovisi sposobnost sustava da dijeli resurse. Višezadaćnost eliminira tu ovisnost, te sa višeprogramiranjem omogućuje operacijskom sustavu da prekine programe usred zadatka i prebaci kontrolu nad procesorom tako brzo da se svakom programu osigurava dio svake sekunde, što prekide čini neprimjetnima.

Višeprogramska obrada ne daje nikakvo jamstvo da će se program izvoditi pravodobno. Prvi se program može izvoditi satima bez potrebe pristupa perifernom uređaju. Ovakav način nije bio problem pošto nije bilo korisnika koji su čekali na terminalu. Samo bi predali bušene kartice operatoru i vraćali se nekoliko sati kasnije za ispis rezultata. Uvelike se smanjilo vrijeme čekanja.

Na slici 7 je prikazana povećana iskoristivost procesora, što je i glavni cilj višeprogramske obrade.



Slika 7 Dijagram zauzeća procesora pri višeprogramskoj obradi (Vukšić i Pejić Bach, 2009)

7.1 Povijest

U početku, ova tehnologija je tražena zbog optimizacije računalnog sustava, jer su vrijeme i resursi bili potraćeni na jedan posao koji je čekao ljudsku interakciju. Razvilo se kao značajka operacijskih sustava kasnih 1950-ih i počelo se redovito upotrebljavati u mainframe računarstvu sredinom 1960-ih. Išlo je uz stopu sa razvojem tehničke podrške koja posjeduje sklopnu logiku i skupove instrukcija koji omogućuju prijenos kontrole između operacijskog sustava i jedne ili više neovisnih aplikacija i korisnika.

Prvo računalo koje je koristilo sustav s višeprogramskom obradom bio je britanski Leo III u vlasništvu J.Lyons and Co. Programi bi se učitali u seriju te bi se prvi pokrenuo. Kada bi prvi program došao do instrukcije koja čeka periferne uređaje, stanje se pohranjivalo, a počeo bi se izvršavati drugi program i tako se proces ponavljao do završetka svih programa.

Ako računalo ima sposobnost obavljati prekide, operacijski sustav će izvršavati program u nekom intervalu, preuzeti natrag kontrolu te izvršavati drugi program u nekom intervalu itd. Ako nema tu sposobnost operacijski sustav počinje izvršavanje programa u nadi (bez sigurnosti) da će mu program eventualno vratiti kontrolu.

Dolaskom virtualne memorije i virtualnih mašina korištenje višeprogramiranja se povećalo. Individualnim programima je omogućeno korištenje memorijskih resursa i operacijskog sustava kao da su neki drugi konkurentni programi (nepostojeći i nevidljivi).

7.2 Prednosti i nedostaci

Prednosti:

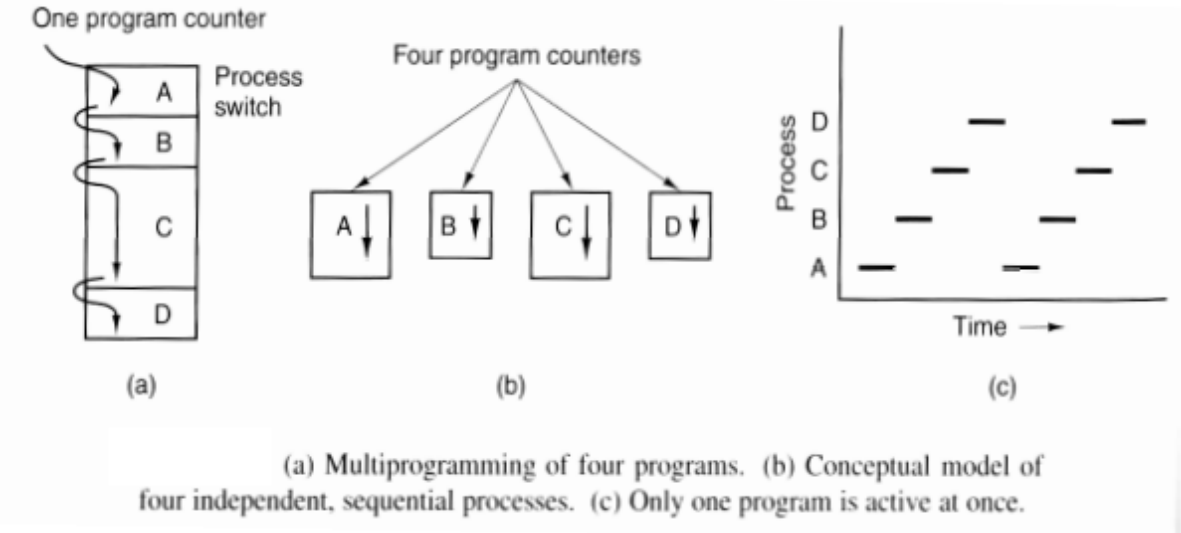
- Povećano je korištenje CPU, smanjeno stanje mirovanja
- Operacijski sustav donosi odluke za korisnika
- Kraće vrijeme odziva

Nedostaci:

- Velika glavna memorija
- Raspoređivanje CPU-a

7.3 Primjeri

Slika 8. prikazuje višeprogramsku obradu 4 programa. Svaki program ima svoj zaseban programski brojač, a u nekom trenutku izvodi se samo jedan program.



(a) Multiprogramming of four programs. (b) Conceptual model of four independent, sequential processes. (c) Only one program is active at once.

Slika 8 Višeprogramska obrada 4 programa (Kovačić, 2008)

8 Simultana obrada

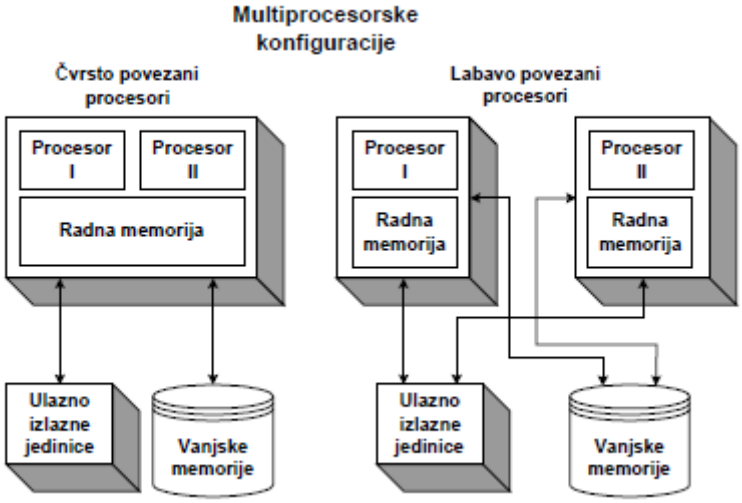
Korištenje dviju ili više centralnih procesorskih jedinica (CPU-a) unutar jednog računalnog sustava nazivamo simultanom obradom (Joch, 2000.). Termin se također odnosi na sposobnost sustava da podržava više od jednog procesora i/ili sposobnost za alociranje zadataka između njih. Definicija simultane obrade varira prema funkciji po kojoj su CPU definirani.

Naziv se ponekad i koristi za izvršavanje više istovremenih procesa programske podrške, međutim izraz višezadaćnosti ili višeprogramiranja prikladnije opisuje ovaj koncept, koji se implementira uglavnom u programskoj podršci, dok je simultana obrada prikladnija za opis korištenja više CPU-a. Sustav može biti simultani i višeprogramirani, samo jedno od ta dva ili niti jedno od njih dvoje.

Svi CPU mogu biti jednaki ili su neki rezervirani za posebne namjene. Dizajn operacijskog sustava u kombinaciji s tehničkom podrškom određuje simetriju procesora u sustavu, npr. samo jedan CPU odgovara na sve prekide tehničke podrške, dok se svi drugi poslovi u sustavu mogu jednako distribuirati među ostalima.

Asimetrična simultana obrada je kada neki procesori obavljaju samo sistemske zadatke, a ostali izvršavaju samo aplikacije. Ovakav način gubi na performansi kada sustav treba obaviti mnogo sistemskih zadataka,a nijedan korisnikov zadatak i obratno. Simetrična simultana obrada je fleksibilnija i ima bolje performanse jer dopušta obavljanje svih zadataka na bilo kojem procesoru.

Slika 9 prikazuje dvije konfiguracije višeprocesorske obrade.



Slika 9 Konfiguracije simultane obrade podataka (Bosilj Vukšić i Pejić Bach, 2009)

Sustavi koji sadrže više CPU povezanih na bus razini zovemo čvrsto povezani višeprocesorski sustavi. Procesori mogu imati pristup središnjoj dijeljenoj memoriji ili mogu sudjelovati u memorijskoj hijerarhiji sa lokalnom ili dijeljenom memorijom.

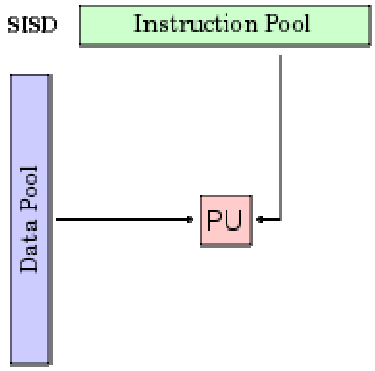
Slabo povezani višeprocesorski sustavi (često se nazivaju klasteri) temelje se na više samostalnih računala s jednim ili dva procesora koji su povezani komunikacijskim sustavom velike brzine (Gigabit Ethernet uobičajeno). Linux Beowulf klaster je primjer slabo povezanih sustava.

Čvrsto povezani sustavi su mnogo učinkovitiji od klastera po potrošnji, zbog toga što se komponente od početka dizajniraju da rade zajedno u čvrsto povezanim sustavima, dok za slabo povezane sustave komponente nisu nužno posebno namijenjene za korištenje u takvim sustavima.

Michael J. Flynn je načinio jednu od najranijih klasifikacija sustava za paralelna i sekvencijalna računala i programe, danas poznatijom kao Flynnova taksonomija. Flynn je klasificirao programe i računala po tome rade li upotrebom jednog ili više seta instrukcija te koriste li instrukcije jedan ili više seta podataka.

Jedna-instrukcija-jedan-podatak (SISD – Single Instruction Single Dana) simultana obrada

Kod računala s jednim setom instrukcija i jednim setom podataka, procesor sekvencijalno procesira instrukcije, a svaka instrukcija procesira jedan podatak (npr. „von Neumann“-ova arhitektura). Dobavljanje uputa i cjevovod su česti primjeri u računalima sa ovom arhitekturom (Quinn, Michael 2004).

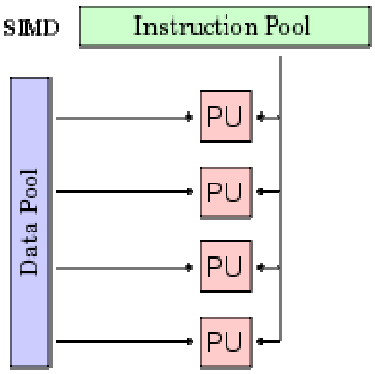


Slika 10 Single Instruction stream, Single Data stream (SISD processing, Wikipedia)

Jedna-instrukcija-više-podataka (SIMD – Single Instruction Multiple Data) simultana obrada

U računalu sa jednim setom instrukcija i više seta podataka, procesor obrađuje tok instrukcija, od kojih svaki može izvoditi izračune paralelno na više podatkovnih lokacija.

Ovaj način je dobar za paralelnu obradu gdje se velika skupina može podijeliti na dijelove koji su podvrgnuti identičnim, ali neovisnim operacijama. Set instrukcija usmjerava rad procesorskih jedinica na istovremeno obavljanje istih zadataka nad velikim količinama podataka (Patterson i Hennessey, 1998).



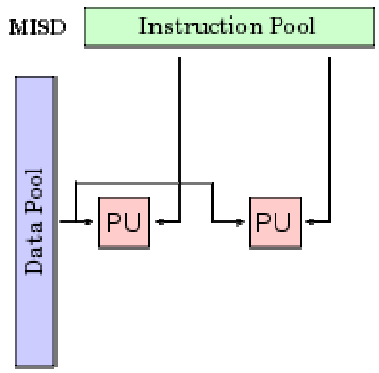
Slika 11 Single Instruction stream, Multiple Data stream (SIMD processing, Wikipedia)

Ovaj tip arhitekture može dati ogromno povećanje učinkovitosti (prema vremenu potrebnom za dovršenje zadatka) nekim vrstama aplikacija. Nedostatak ove arhitekture je taj da velik dio sustava miruje kada se izvodi program koji se ne može podijeliti u manje jedinice. Također programi se moraju posebno pisati da maksimalno iskoriste arhitekturu, a često se dizajniraju i specijalni prevodioci za optimizaciju. Neki prevodioci dopuštaju programerima izravno odrediti operacije koje će se izvoditi paralelno.

Jedna-instrukcija-više-podataka arhitektura se dosta koristi u pojedinim područjima kao što su računalne simulacije, ali je malo korišten u stolnim računalima opće namjene i poslovnim računalnim okruženjima.

Više-instrukcija-jedan-podatak (Multiple Instruction Single Data) simultana obrada

Više-instrukcija-jedan-podatak arhitektura nudi prednost redundancije, budući da više procesorskih jedinica obavlja iste zadatke na istim podacima, smanjujući šanse netočnih rezultata ako jedna od jedinica ne uspije. Osim redundancije i sigurnosti od neuspjeha ima još par prednosti, ali ne poboljšava performanse i veoma je skupo. Koristi se kod nanizanih procesora i računala koja toleriraju pogreške, te se može implementirati da je transparentno za programsku podršku.

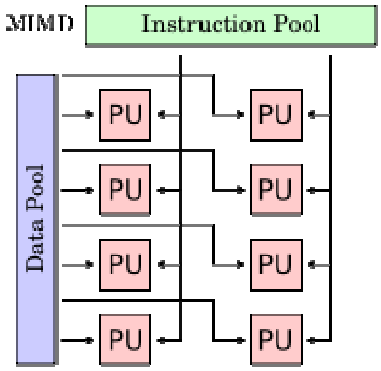


Slika 12 Multiple Instruction stream, Single Data stream(MISD processing, Wikipedia)

Više-instrukcija-više-podataka (MIMD – Multiple Instruction Multiple Data) simultana obrada

Više-instrukcija-više-podataka arhitektura je pogodna za širok raspon zadataka u kojima se potpuno neovisno i paralelno izvršavaju instrukcije koje su u doticaju sa različitim skupinama podataka. Zbog toga i jer se lako implementira ova arhitektura dominira.

Obrada je podijeljena na više niti (svaka s vlastitim stanjem procesora) unutar jednog ili više softverski definiranog procesa. Ova arhitektura vrlo učinkovito koristi resurse tehničke podrške, ako ima više niti (sistemske ili korisnikove) koje čekaju otpremu.



Slika 13 Multiple Instruction stream, Multiple Data stream (MIMD processing, Wikipedia)

Arhitektura više-instrukcija-više-podataka zahtijeva posebno kodiranje u operacijskom sustavu računala, jer se niti mogu na nepredvidive načine sudarati pri pristupu resursima, ali ne zahtijeva promjene aplikacija osim ako sami programi ne koriste više niti (MIMD je transparentan jedno-nitnim programima kod većine operacijskih sustava). Ponekad sustav treba koristiti softverske tvorevine kao što su vrata, kako bi se spriječilo ometanje među linijama. Ovaj proces poveća složenost koda, smanjuje performanse i uvelike povećava količinu potrebnog testiranja, iako obično ne dovoljno da bi negirali prednosti simultane obrade.

Slični konflikti mogu nastati na razini tehničke podrške između procesora (sukob i korupcija cache memorije), i mora se obično riješiti na toj razini ili s kombinacijom programske i tehničke podrške.

8.1 Povijest

Za računala veličine sobe 1960-i 1970-ih jeftin i učinkovit način za povećanje računalne snage je dodavanje drugog CPU-a. Pošto su to već bila najbrža računala na raspolaganju (prema omjeru cijene i performansi), dva CPU-a standardne brzine su bili mnogo jeftiniji nego duplo brži CPU. Prva višeprocesorska računala su bili BurroughsB5000, DECsystem-1055 i IBM System/360 model 65MP.

8.2 Prednosti i nedostaci

Prednosti:

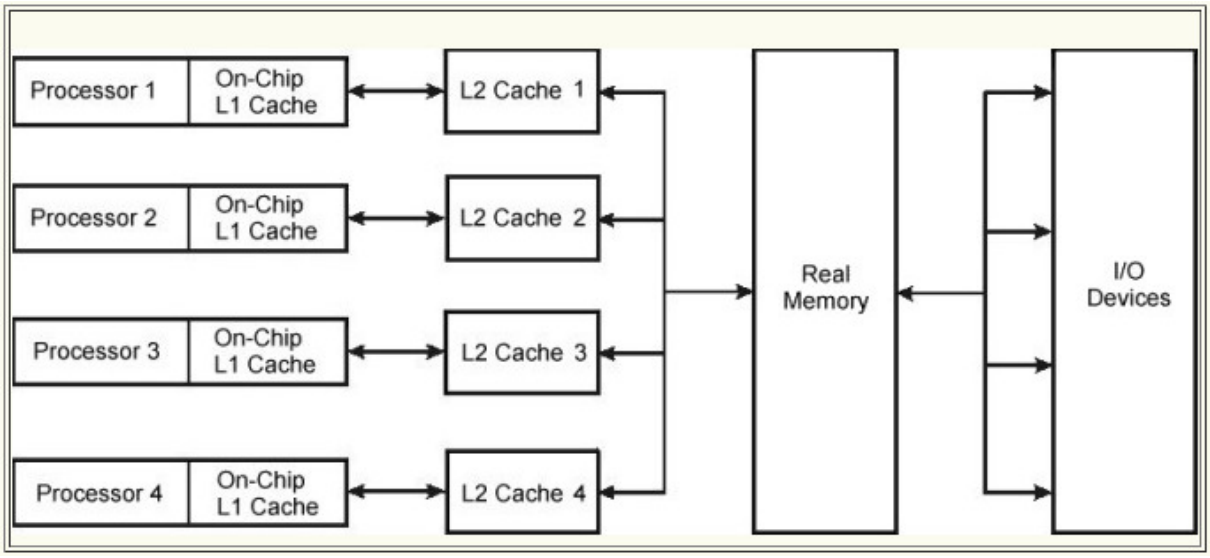
- Sposobnost višezadaćnosti je uvelike povećana
- Sve više i više programa je višenitno (višenitni program može koristiti više od jednog procesora)

Nedostaci:

- Cijena tehničke podrške
- Velika potrošnja

8.3 Primjer

Slika 14 prikazuje 4 procesora. Podaci se prenose između L1 cache procesora na L2 cache koji je dodijeljen tom procesoru. Od L2 podaci se mogu prenijeti u memoriju i natrag ili slati u U/I uređaje za obradu.



Slika 14 Tipični simetrični višeprocesorski sustav (Performance Management Guide, IBM)

9 Daljinska obrada podataka

Metoda obrade podataka u kojoj se podaci obrađuju na udaljenoj lokaciji te se rezultati vraćaju natrag zove se daljinska obrada podataka (*teleprocessing*). Podaci se prenose preko neke udaljenosti pa moramo imati odgovarajuće ulazno izlazne uređaje i telekomunikacijske sustave. S obzirom na veze između centralnog računala i udaljenih uređaja razlikuju se dva temeljna oblika daljinske obrade podataka:

- *on-line* obrada, što znači da između računala i udaljenih uređaja postoji neposredna stalna elektronska veza.
- *off-line* obrada, što znači da ne postoji stalna elektronska veza. Ponekad se ona uopće ne može uspostaviti, ili se uspostavlja prema potrebi.

On-line obrada

- Obrada podataka se obavlja u onom vremenu kada korisnik unese podatke u računalni sustav.
- To je ujedno i način obrade podataka pri kojem se podaci obrađuju na udaljenoj lokaciji u odnosu na onu gdje nastaju i gdje se koriste rezultati obrade.
- Aplikacija čeka da joj korisnik kaže što i kada učiniti
- Korisnik čeka neko vrijeme na odgovor.

9.1 Povijest

Daljinska obrada podataka se pojavila ranih 1960.-ih kada su se javile ideje o povezivanju računala, kako bi mogli slati i dijeliti podatke. Ubrzo su se pojavile tehnike udaljenih perifernih uređaja (pisači i čitači bušenih kartica) koji su izravno spojeni sa središnjim računalom.

Faze razvitka daljinske obrade podataka

1. Prostorno ograničena daljinska obrada podataka

- pomoću kablova računala povezana s centralnim računalom
- nedostatak - do udaljenosti 600 metara (iznad toga se događaju greške)

2. Prostorno neograničena daljinska obrada podataka

- računala su putem modema i telefonske mreže povezana s centralnim računalom (preteča Interneta)
- modem pretvara analogni telefonski signal u digitalne računalu razumljive impulse
- nedostatak - centralno se računalo koristi za cjelokupnu obradu podataka zbog čega dolazi do preopterećenja centralnog računala i pada računalnog sustava
- kao rezultat tih nedostataka javljaju se distribuirane računalne mreže

3. Distribuirane računalne mreže

- računala su povezana s nekoliko servera
- povećava se otpornost cijelog sustava na kvarove
- korisnik ne zna koje računalo obrađuje njegov zahtjev
- korisnik bira lokaciju gdje će biti dostavljeni rezultati

4. multimedijske mreže

- računalne mreže koje osim računala i servera obuhvaćaju i uređaje drugih namjena (videokamere, sastanci na daljinu, …)

9.2 Prednosti i nedostaci

Prednosti:

- Struktura organizacije (npr. bankama su potrebne lokalne i globalne transakcije)
- Poboljšana dostupnost (ako jedan dio sistema ne radi, neće pasti cijeli sistem)
- Poboljšane performanse

Nedostaci:

- Kompliciranost rada i dizajna
- Cijena održavanja
- Manja sigurnost

9.3 Primjer

Najčešći primjer komunikacije između računala je računalna mreža. Svako računalo ima svoju odgovornost i priznata je od strane drugih računala u mreži kao autonomna. Terminali i printeri spojeni na računalo s podjelom vremena je također učestali primjer.

10 Zaključak

Prije pojave računala podaci su se obrađivali ručno od strane ljudi. Takav način je bio nepouzdan (sklon greškama) i dugotrajan. Metode obrade podataka u računalnim sustavima uvelike olakšavaju i ubrzavaju rješavanje zadataka, bilo za tvrtke ili obične korisnike. Brzina, preciznost i pouzdanost ovise o metodi koju izaberemo, te o resursima kojima raspolažemo.

Serijska obrada je najjednostavniji oblik organizacije rada računala, jer se u jednom trenutku izvodi samo jedan korisnički program koji raspolaže svim računalnim resursima. Obrada podataka teče onim redoslijedom kako su programi i podaci grupirani na ulaznim jedinicama.

Višeprogramska obrada podataka nastala je kao odgovor na zahtjev da se što efikasnije koriste računalni resursi, tako da se omogući istovremeno izvođenje više korisničkih programa na računalu koje ima samo jedan procesor. Stvorene su tehnike koje omogućavaju paralelni rad više programa, pri čemu se stvara dojam simultan obrade, ali se naziva i kvazisimultana obrada.

Obrada s podjelom vremena (Time sharing) omogućava da se brže dobiju rezultati kratkih obrada nego kod drugih načina rada, ako istovremeno više korisnika traži izvođenje svojih programa.

Simultana obrada podataka (Multiprocessing) koristi više procesora koji su sastavni dio jednog računala. Procesori rade međusobno neovisno, dijele ukupni posao obrade podataka i imaju istovremeni pristup do zajedničke centralne memorije. Ovakva konfiguracija ima čvrsto povezane procesore,a druga se konfiguracija sastoji od dvije centralne jedinice koje koriste zajedničke ulazno/izlazne jedinice i kaže se da imaju slabo povezane procesore.

Obrada u realnom vremenu (Real-time processing) se primjenjuje u informacijskim sustavima kod kojih trenutno stanje baze podataka mora odgovarati stvarnom stanju sustava na koji se odnosi. Računalo mora unutar zadanog vremenskog intervala primiti podatke, obraditi ih i izdati rezultate obrade. Taj vremenski interval se naziva vrijeme odgovora, a teče od izdavanja zahtjeva do završetka obrade.

Daljinska obrada (Teleprocessing) je metoda obrade podataka pri kojem se podaci obrađuju na drugoj, udaljenoj lokaciji u odnosu na onu gdje nastaju i gdje se koriste rezultati obrade

Svaka od ovih metoda je učinkovito rješenje za obrađivanje podataka, te se zbog njihovih različitih osobina upotrebljavaju i dan danas.

11 Literatura

Ben-Ari M., "Principles of Concurrent and Distributed Programming", Prentice Hall, 1990. ISBN 0-13-711821-X. Poglavlje 16, Stranica 164

Bosilj Vukšić V., Pejić Bach M. (ur.): "Poslovna informatika", Element, Zagreb, 2009.

Dadić T., Skripta iz Operacijskih sustava, PMFST, 2011.

David A. Patterson, John L. Hennessey, *Computer Organization and Design: the Hardware/Software Interface*, 2nd Edition, Morgan Kaufmann Publishers, Inc., San Francisco, Califoria, 1998, p.751

DRI: The Digital Research Initiative, http://ibiblio.org/team/index.html (07/10/2011)

Joch A., „Chip multiprocessing", 2000.

Klein R. : http://www.staff.uni-mainz.de/roklein/ql (07/10/2011)

Kovačić B., Skripta iz Operacijskih sustava, 2008.

Liu C., Layland J, Scheduling Algorithms for Multiprogramming in a Hard Real-time Environment. Journal of the ACM, 20(1):46--61, Jan. 1973.

Ojala P., "A Pre-emptive Multitasking Operating System for an Embedded Emulation Tool",1991.

Quinn, Michael J. "Chapter 2: Parallel Architectures." Parallel Programming in C with MPI and OpenMP.Boston: McGraw Hill, 2004

Tuđman M., Boras D., Dovedan Z., „Uvod u informacijske znanosti", Zagreb, 1991.

Serijska obrada : http://en.wikipedia.org/wiki/Batch_processing (29/09/2011)

Obrada u stvarnom vremenu: http://en.wikipedia.org/wiki/Real-time_computing (29/09/2011)

Obrada s podjelom vremena : http://en.wikipedia.org/wiki/Time-sharing (29/09/2011)

Višezadaćna obrada : http://en.wikipedia.org/wiki/Computer_multitasking (29/09/2011)

Višeprogramska obrada : http://en.wikipedia.org/wiki/Multiprogramming (29/09/2011)

Simultana obrada : http://en.wikipedia.org/wiki/Multiprocessing (29/09/2011)