Paralelni računarski sistemi

Potreba za sve bržim računarima ne prestaje od početka računarske ere. Do sada su se proizvodjači računara prilično dobro nosili sa zahtevima za većom brzinom obrade. 1948. godine elektronske komponente su mogle komutirati svoje[1] stanje 10000 puta u sekundi. Vreme komutacije elektronskih komponenti savremenih računara iznosi 1/10000000000 delova sekunde. Ovo ukazuje da se poslednjih pedesetak godina broj operacija koje računar može obaviti u jednoj sekundi ud-vostručavao svake dve godine. Ovo zvuči veoma impresivno, ali postavlja se pitanje koliko će još to trajati. Opšte je mišljenje da će se ovaj trend nastaviti do kraja ovog veka. Moguće je zadržati ovaj trend i nešto duže korišćenjem optičkih ili biološki baziranih komponenti. A šta posle toga?

Ako su tekuće aplikacije i aplikacije koje se imaju u planu nekakav pokazatelj, zahtevi za brzinom obrade će nastaviti da rastu istim intenzitetom i narednih godina. Već sada su potrebni računan brži od raspoloživih za. obavljanje ogromnog broja izračunavanja pri razvoju lekova za mnoge bolesti. Veoma brzi računan su neophodni u aplikacijama koje simuliraju ljudske sposobnosti u prepoznavanju kompleksnih vizuelnih i auditivnih oblika, u seizmologiji, prognozi vre mena, itd.

Navešćemo jedan ilustrativni primer. Da bi se obavila simulacija toka jednog hiruškog zahvata potrebno je, na specijalnom ekranu, prikazati rekonstruisanu trodimenzionalnu sliku tela pacijenta spremnog za operaciju. Potrebno je zatim obezbediti rotaciju slike u proizvoljnom smeru, prikaz poprečnog preseka organa nad kojim treba izvršiti zahvat, u svom realnom okruženju, zatim obaviti simulaciju operacije i posmatrati njene efekte. Minimalna brzina obrade, da bi se ovo obavilo u razumnom vremenu, iznosi 10^{15} operacija u sekundi. Ovakve zahteve danas ne mogu podržati ni tvz. superračunari čije su brzine obrade reda 10^{10} do 10^{11} operacija/sec.

U ogromnoj većini današnji računari, od najjednostavnijih do najmoćnijih, koncepcijski su vrlo slični jedni drugima. Njihova arhitektura i način rada slede, manje ili više, osnovni princip formulisan krajem četrdesetih od strane John von Neu-manna. Osnovni princip rada je veoma jednostavan: upravljačka jedinica pribavlja instrukcije i operande iz memorije i šalje ih u procesnu jedinicu gde se izvršavaju odgovarajuće operacije, a zatim se rezultat vraća u memoriju. Ovaj niz aktvnosti se ponavlja za svaku instrukciju. Postoji samo jedna jedinica od svake vrste i samo jedna, instrukcija se može izvršiti u datom trenutku.

Dugo vremena se povećanje brzine obrade računara postizalo korišćenjem bržih elektronskih komponenti u istoj osnovnoj strukturi računiara. Tako su relei bili za-menjeni vakuumskim cevima, a ove tranzistorima a oni poluprovodničkim elementima sa malim, pa srednjim i visokim stepenom integracije. Na žalost, evidentno je da će se ovaj trend uskoro zaustaviti. Fundamentalna barijera koja ograničava sve brzine izračunavanja je brzina svetlosti u vakuumu. Naime,

signali ne mogu napredovati kroz provodnik brže od brzine svetlosti. Ova brzina približno iznosi 3 x 10^8 *m/s*. Pretpostavimo da neki elektronski uredjaj može da obavi 10^{12} operacija u sekundi. Ako se dva takva uredjaj a nalaze na rastojanju od pola milimetra, tada će signalu biti potrebno duplo više vremena da predje put izmedju njih nego da se izvrši njegova obrada u bilo kom uredjaju. Drugim recima, sva dobit postignuta povećanjem brzine elektronskih komponenti gubi se dok jedna komponenta čeka da primi ulaz iz druge. Može se postaviti pitanje zašto komponente ne postaviti bliže jednu drugoj. Opet, zakoni fizike kažu da redukovanje rastojanja izmedju komponenti jednog trenutka dostiže tačku posle koje one počinju da interaguju jedna sa drugom redukujući ne samo brzinu već i pouzdanost.

Na osnovu izloženog može se zaključiti da se za dalje povećanje brzine obrade računara moraju pronaći druge metode. Jedan od načina za povećanje brzine obrade je uvodjenje *protočnosti* ili *obrade sa preklapanjem* kod izvršenja instrukcija. Protočnost se veoma mnogo koristi kod modernih superračunara ali i kod savre-menih mikroprocesora. Brzine obrade koje se mogu postići uvodjenjem protočnosti su reda 10^9 do 10^{11} operacija/sec.

Drugi prilaz za povećaiij brzine izračunavanja je korišćenje *paralelne obrade*. Ovde se uzima u obzir činjenica da mnoga izračunavanja u programu ne zavise jedna od drugih i da se mogu izvršavati jednovremeno i tako redukovati vreme izračunavanja. Ova ideja nije nova. Naime, odavno je uočeno da različite komponente računara mogu obavljati različite aktivnosti u isto vreme. Na primer, dok centralna jedinica obavlja izračunavanje, U/I procesor može obaviti čitanje sa magnetne trake i generisati izlaz na štampaču. Kod savremenih mašina postoji više jednostavnih procesora, svaki specijalizovan da izvršava odredjene zadatke, kao, na primer, operacije nad podacima predstavljenim u pokretnom zarezu.

Medjutim, ni jedan od navedenih primera,, striktno govoreći, ne predstavlja paralelni računiar. U savremenoj paradigmi, paralelni računar je računar koji se sa.stoji od više procesnih jedinica, ili procesora, približno jednake moći obrade. Problem koji treba da se resi razbija se na potprobleme, a zatim se oni rešavaju simultano na različitim procesorima. Na kraju se rezultati objedinjuju da bi se dobio odgovor na polazili problem.

Broj procesora u paralelnom računam može varirati od nekoliko desetina do nekoliko miliona. Kao posledica toga vreme potrebno za rešavanje problema se znatno smanjuje. Ovaj prilaz je atraktivan iz nekoliko razloga. Prvo, za mnoge računarske probleme prirodno rešeiije je paralelno. Drugo, cena i dimenzije računarskih komponenti su se tako drastično smanjivale poslednjih godina da su paralelni računari sa velikim brojem procesora postali realnost. I treće, kod paralelne obrade je moguće odabrati paralelnu arhitekturu koja je najpogodnija za rešavaiije posmatranog problema ili klase problema od interesa. Zaista, projektanti paralelnih arhitektura imaju slobodu u odlučivanju o broju procesora, o moći izračunavanja svakog od njih, o načinu njihovog medjusobnog povezivanja, o tome da li koriste zajedničku memoriju, do koje mere njihov rad treba da bude sinhron, itd. Ovako veliko mogućnosti izbora odrazile su se u postojanju velikog broja teoretskih modela paralelnih izračunavanja. Empirijski dokazi dobijeni na prototipovima često potvrdjuju rezultate teorijskog proučavanja. Od nedavno,

paralelni računari su postali komercijalno raspoloživi na tržištu. Efikasno korišćenje paralelnih računara zahteva odgovarajuće paralelne algoritme, programske jezike, kompilatore i operativne sisteme. Sve navedene kom ponente privlače veliku pažnju istraživača.

Istrorijski razvoj računara

U proteklih pet decenija računari su u toku svog razvoja prošli kroz pet generacija. Odmah istaknimo da i pored toga što je generacijska klasifikacija korisna, ona se zasniva na velikom broju ne baš tako jasnih kriterijuma. Ono što je najvažnije, evolucija računara traje i trajaće i u ogromnoj meri zavisi od tehnologije izrade komponenata i sklopova. Cesto projektantske inovacije moraju sačekati pogodnu tehnološku bazu da bi se ideje mogle implementirati. Sa druge strane zakoni tržišta su takodje imali izuzetno veliki uticaj na evolucioni proces računara.

Prva generacija

Računari prve generacije su posedovali jednu centralnu procesorsku jedinicu (CPU), koja je koristila programski brojač (PC), akumulator (ACC), obavljala instrukcije grananja i izvršavala aritmetičke operacije u fiksnom zarezu. Sve operacije u sistemu, tj. prenos reči izmedju ulazno—izlaznih (U/I) uredjaja i memorije, su zahtevale direktnu intervenciju centralnog procesora. Programi prvih računara su bili pisani u binarnom kodu, tj. na mašinskom jeziku. Ovakve programe je bilo teško pisati, još teže identifikovati njihove aktivnosti, a svakako najteže ispraviti greške. Značajan napredak, za to doba, u programiranju postignut je ranih pedesetih godina uvodjenjem simboličkog programiranja, tj. programiranja na asemblerskom jeziku.

Druga generacija

Računari druge generacije karakterišu period od 1955 do 1964 godine. Računari ove generacije su najviše prepoznatljivi po promeni tehnologije, tj. prelasku sa vakuumskih cevi na tranzistore. Druge važne inovacije karakteristične za ovu generacju su::

- Najveći broj logičkih sklopova je realizovan poluprovodničkim komponentama -tranzistori i diode
- CRT (Cathode Ray Tube)
- Memorije i memorije zasnovane na linijama za kašnjenje zamenjene su fefritnim jezgrima i magnetnim dobošima
- kompaktnije komponente za realizaciju glavne memorije
- Memorijski podsistemi su projektovani za rad u vremenskom multipleksu
- uskladjivanje brzine rada centralnog procesora i memorije
- Uvedeni su indeksni registri i realizovan je hardver za aritmetiku u pokretnom zarezu — povećana je moć i brzina obrade centralnog procesora
- U cilju pojednostavljenja programiranja uvedeni su mašinsko nezavisni programski jezici visokog nivoa, kao što su Algol, Cobol i Fortran - povećana efikasnost u programiranju
- Razvijeni su specijalni U/I procesori čime se CPU oslobodio detalja i

aktivnosti u vezi rada U/I podsistema - brži i efikasniji U/I prenos

- Proizvodjači računara su počeli da nude sistemski softver kao što je bač monitor, bibliotečki potprogrami, kompilatori, idr. -- povećana efikasnost u radu

Treća generacija

Za početak treće generacije računara smatra se 1965. godina. Ipak, razlika izmedju druge i treće generacije se ne može tako strogo napraviti. Glavne inovacije kod računara treće generacije su sledeće:

- Blokovi realizovani diskretnim komponentama (tranzistorima) zamenjeni su integrisanim kolima - došlo je do značajnog smanjenja fizičkih dimenzija računara i cene računara

- Memorije sa magnetnim jezgrima zamenjene su poluprovodničkim memorijama (ROM i RAM tipa) — povećana brzina rada i kapacitet glavne memorije, smanjeni gabarit i cena sistema

- Prvi put je uvedena tehnika mikroprogramiranja — pojednostavljeno projektovanje CPUa i povećana njegova efikasnost

—Uvedena je tehnika protočnosti — povećana efektivna brzina izvršenja programa

- Implementiran je multiprogramski režim rada - ideja je da se izvrši preklapanje u radu CPUa i U/I aktivnosti u toku izvršenja većeg broja korisničkih programa. Ovo je dovelo do razvoja operativnih sistema sa deobom vremena, koji koriste princip rada sa virtuelnom memorijom ili naglašeniju deobu ili miultipleksiranije u toku korišćenja resursa (procesora, memorijskog prostora, U/I uredjaja)

—Instalirana je keš memorija — koristeći osobine lokalnosti koda i programa, prvenstveno je namenjena da efektivno skrati vreme pristupa glavnoj memoriji

Četvrta generacija

Od početka šezdesetih godina dominantna tehnologija za izradu računarskih komponenti postala je tehnologija integrisanih kola. Ova tehnologija je evoluirala od integrisanih kola (IC) realizovanih sa svega nekoliko tranzistora, SSI (Single Scale Integration), do kola sa stotine hiljada tranzistora, VLSI (Very Large Scale Integration). Osnovne karakteristike ove generacije računara su:

VLSI dizajn — masovna proizvodnja IC računarskih komponenti u obliku standardnih mikroprocesora, mikroračunara na čipu, mikrokontrolera i signal procesora, učinila je mogućim da projektanti realizuju za veoma kratko vreme standardne i nestandardne (po porudžbini) računarske sisteme

Paralelna obrada - ideja o realizaciji superračunara koji je sposoban da paralelno izvršava veći broj instrukcija datira još od pedesetih godina, ali su cena hardvera i kompleksnost programiranja ograničavali njegovu pojavu na tržištu i primenu u aplikacijama. Danas se mašine ovakvog tipa masovno koriste kako za naučne tako i za ostale tipove aplikacija. Tipični predstavnici su vektorske mašine koje u velikoj meri koriste protočnost u obradi. Alternativno rešenje su višeprocesorski sistemi koji veći stepen paralelizma ostvaruju koristeći princip nezavisnog rada većeg broja procesora

Višeprocesorski operativni sistemi - razvoj ovih tipova operativnih sistema

pokrivao je sledeća dva pravca: (1) da se ostvari veća efikasnost kod sitno zrnastog paralelizma tj. paralelizma na nivou instrukcija; (2) da se postigne veća brzina kod krupno-zrnastog paralelizma tj. paralelizma na nivou poziva procedura/

- Paralelni programski jezici i kompilatori - zasnivaju se na klasičnim programskim jezicima visokog nivoa kojima su dodate jezičke konstrukcije (proširenja) koje imaju za cilj da olakšaju programiranje i povećaju efikasnost u radu.

Peta generacija

1991. godina karakteriše početak ere pete generacije računara. Izučavanja koja se odnose na tipove ovih arhitektura, način kreiranja algoritama, ocena performansi i dr., biće u žiži interesovanja ove knjige. Za sada, ukažimo samu ukratko na. neke najvažnije osobine ove grupe računara:

1. Masivni paralelizam - masivni paralelni računar čini desetine, hiljade, ili milioni procesora. Obično se sistemi sastavljeni od manjeg broja procesora (do nekoliko hiljada) realizuju sa autonomnim procesorskim jedinicama sposobnim da manipulišu sa brojevima u pokretnom zarezu. Sa druge strane, sistemi sa velikim brojem procesora (reda milion), se realizuju sa jedno bitnim procesorima koji operišu ritmično i kojima se radi povećanja procesne moći pridružuju odgovarajući koprocesori za rad sa podacima predstavljenim u pokretnim zarezu.

2. Proširljivost (skalbihiost) imperativ je da svaki masivno paralelni sistem bude proširljiv. Osobina proširljivosti se odnosi kako na paralelne arhitekture tako i na paralelne algoritme, pri čemu je glavni cilj da se postigne/očuva visok stepen efikasnosti paralelne obrade kada se broj procesora povećava.

3. Rekonfigurabilnost -- realizacija mašino pralelnih rekonfigurabihiih arhitektura u direktnoj je vezi sa VLSI i WSI implementacijom, pa je u tom smislu neophodno da ove arhitekture poseduju prilagodljivost i tolerantnost na greške.

3. Heterogena obrada - Sposobnost da se reše problemi velikog obima koristeći
mrežu heterogenih računara sa deljivom virtelnom memorijom.

4. Razvoj neuronskih mreža i optičkih računara predstavljaju dve oblasti istraživanja koje u budućnosti mnogo obećavaju. Neuronske mreže karakterišu sledeće osobine: Masivni paralelizam, jako izražena medjusobna povezanost procesora, jednostavni procesori, tolerantnost na greške, kolektivno izračunavanje i sposobnost adaptacije arhitekture. Prednosti optičkih u odnosu na. elektronske računare su: propagacija signala po kanalima se vrši bez interferencije, optički signali putuju brzinom svetlosti, opteretljivost kanala može biti velika; zbog velike brzine minimizira se problem redova čekanja; rekonfigurabilnost se ostvaruje u realnom vremenu korišćenjem holograma; problem globalne distribucije takta i signala nije više kritičan.

Podela paralelnih računarskih sistema

Podela. računarskih sistema koju ćemo ovde dati zasniva se na mehanizmima upravljanja. Konvencionalni računan, tj. računari *von Neumannovog tipa,* se zasnivaju na mehanizmu upravljanja koji je eksplicitno odredjen tokom instrukcija u programu. Drugu klasu računara čine *Dataflow* računari koji se zasnivaju na mehanizmu upravljanja tokom podataka koji dozvoljava izvršenje bilo koje instrukcije čiji su operandi raspoloživi. Ovi računari imaju naglašeni visoki stepen paralelizma na sitno zrnastom nivou, tj na nivou instrukcija. Treću klasu računara. čine tvz. Reduction računari koji se zasnivaju na mehanizmu upravljanja po zahtevu koji inicira izvršenje neke operacije na osnovu zahteva za njenim rezultatom od strane neke druge instrukcije.

Računan upravljam programskim, tokom,

Konvencionalni von Neumannovi računari koriste programski brojač da bi odredili redosled izvršenja instrukcija u programu. Ovaj sekvencijalni stil izvršenja zove se programski upravljan jer tokom izvršenja programa upravlja programer navodeći eksplicitno u svom programu redosled izvršenja instrukcija. Računari ovog tipa koriste deljivu memoriju za pamćenje instrukcija i podataka. Promenljive u deljivoj memoriji se mogu ažurirati od strane više instrukcija. Izvršenje jedne instrukcije može uticati na druge instrukcije jer je memorija deljiva. U mnogim slučajevima ovo može sprečiti paralelizaciju programa. Medjutim, mehanizam upravljanja programskim tokom se može paralelizovati korišćenjem paralelnih jezičkih konstrukcija ili paralelnih kompilatora. U ovoj knjizi mi ćemo najviše pažnje posvetiti ovoj klasi računara i tehnikama za njihovo programiranje.

Računari upravljani tokom podataka

Kod Dataflow računara izvršenje instrukcije je odredjeno raspoloživošću podataka a ne programskim tokom. Teorijski, bilo koja instrukcija može postati spremna za izvršenje čim su njeni operandi raspoloživi. Instrukcije u programu čije je izvršenje upravljano tokom podataka nisu uredjene ni na koji način. Umesto da se podaci pamte u deljivoj memoriji, oni se direktno drže u instrukciji. Rezultati izračunavanja se prenose direktno izmedju instrukcija. Rezultat generisan od strane jedne instrukcije biće umnožen i distribuiran svim instrukcijama kojima je potreban. Podatak iskorišćen od strane jedne instrukcije nije više raspoloživ za korišćenje drugim instrukcijama.

Mehanizam upra.vljanja tokom podataka ne zaliteva deljivu memoriju, programski brojač i sekvencer. Medjutim, on zaliteva specijalne mehanizme za detekciju raspoloživosti podataka, tj. za uparivanje podataka sa instrukcijama kojima su potrebni, i mehanizam koji će omogućiti lančanu reakciju asinhronog izvršenja podataka. S obzirom da nema deljive memorije sa podacima, nema nikakvih sporednih efekata koji sprečavaju paralelno izvršenje instrukcija. Asinhronost u radu ukazuje na potrebu za. nekom handshake procedurom.Računali ove klase poseduju sitno-zrnasti paralelizam na nivou instrukcija. Masivni paralelizam kod ovih računara bio bi će moguć ako bi se ovaj mehanizam upravljanja mogao implementirati po razumnoj ceni. Za sada

postoji samo nekoliko eksperimentalnih modela računara ovog tipa.

Računan upravljani po zahtevu

Kod Reduction mašina izvršenje neke instrukcije je inicirano potrebom za njenim rezultatom. Posmatrajmo izračunavanje sledećeg ugnježdenog aritmetičkog izraza *a = ((b* + 1) x *c — (d/f.))*. Izračunavanje upravljano tokom podataka koristi prilaz "odozdo- naviše" kod izračunavanja startujući od najdublje ugnježdenih operacija *b* + l i *d/e*. zatim nastavlja sa operacijom x, i završava sa operacijom —. Takvo izračunavanje se zove "halapljivo" jer se operacije izvode čim su odgovarajući operandi dostupni.

Izračunavanje upravljano po zahtevu koristi prilaz " s-vrha-na niže" tako što se prvo zaliteva vrednost a koja zaliteva izračunavanje izraza u sledećem nivou (*b*+1) x *c* i *d*/e. koji pak dalje zahtevaju izračunavanje *b*-\-1 u najdublje ugnježdenom nivou. Rezultati se zatim vraćaju višim nivoima u obrnutom redosledu od redosleda zaliteva dok se ne izračuna vrednost *a*.

Izračunavanje upravljano po zahtevu odgovara "lenjom izračunavanju". jer se operacije izvršavaju samo onda kada se njihov rezultat zaliteva od strane druge instrukcije. Mehanizam upravljanja po zahtevu se prirodno slaže sa konceptom funkcionalnog programiranja.

Podela von Neumannovih računara

Svaki računar, sekvencijalni ili paralelni, radi tako što izvršava neke instrukcije nad podacima. Niz instrukcija (algoritam) kaže računaru šta da radi u svakom koraku. Ove instrukcije deluju na niz podataka. U zavisnosti da li postoji jedan ili nekoliko ovakvih nizova razlikuju se četiri klase računara:

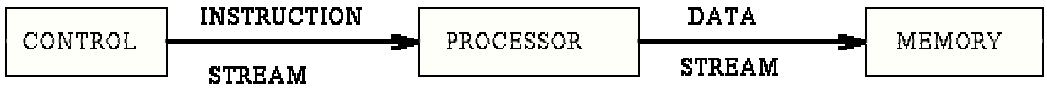
1. računan sa. jedinstvenim instrukcionim tokom i jedinstvenim tokom podataka (SISD);
2. računari sa. višestrukim tokom instrukcija i jedinstvenim tokom podataka.
   (MISD);
3. računari sa jedinstvenim tokom instrukcija i višestrukim tokom podataka.
   (SIMD);
4. računari sa višestrukim tokom instrukcija i višestrukim tokom podataka
   (MIMD).

Sada ćemo ukratko navesti osobine pojedinih klasa. U ovom razmatranju neće biti govora o ulazu, izlazu ili perifernim jedinicama računara.

SISD RAČUNARI

Računari ove klase sastoje se od jedne procesne jedinice koja prima jedinstveni tok instrukcija koje deluju na jedinstveni tok podataka, kao što je prikazano na Slici 1.1. U svakom koraku obrade upravljačka jedinica izdaje jednu instrukciju koja specificira obradu nad podatkom pribavljenim iz memorijske jedinice. Ta instrukcija može biti, npr., neka aritmetička ili logička operacija.

Algoritmi za računare ove klase su sekvencijalni algoritmi. Oni ne poseduju nikakav paralelizam. Razlog je očigledan: postoji samo jedan procesor. Da bi se bilo kakav paralelni oblik rada postigao, potrebno je više procesora. Ovo poseduju sledeće tri klase- računara.

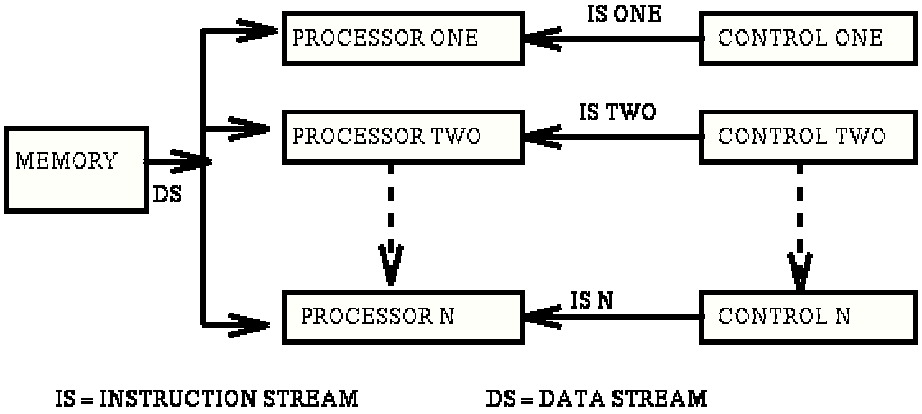


Slika 1.1. SISD računar

MISD RAČUNARI

Ovde postoji *N* procesora, svaki sa svojom sopstvenom upravljačkom jedinicom, koji dele zajedničku memoriju u kojoj su smešteni podaci (Slika 1.2a). Postoji *N* tokova instrukcija i jedan tok podataka. U svakom koraku procesori jednovremeno izvršavaju različite instrukcije, koje dobijaju iz svojih upravljačkih jedinica,, nad jednim podatkom pribavljenim iz zajedničke memorije.

Druga definicija MISDa se odnosi na protočni sistem kod koga se izlazni podaci jednog stepena koriste kao ulazi u naredni stepen. Jedinstveni tok podataka se sekvencijalno prenosi kroz sistem i vrši se njegova transformacija u svakom stepenu (Slika 1.2b)



Slika 1.2. MISD računar

Primer obrade, koja se veoma uspešno može obaviti na MISD računaru, je prepoznavanje objekata. Svaki procesor može prepoznati objekte iz jedne klase izvršavajući odgovarajuće testiranje. Podatak vezan za objekat koji treba

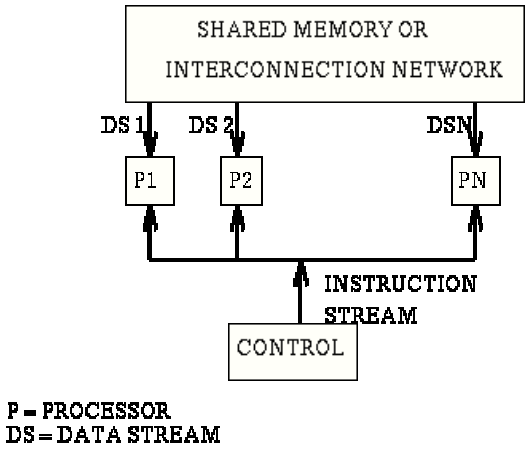
prepoznati šalje se jednovremeno svim procesorima gde se zatim obavlja testiranje.

Izračunavanja koja se mogu obaviti na ovim računarima su veoma specijalizovane prirode. Za većinu aplikacija MISD način, obrade je nepogodan. Paralelni računari koji su mnogo fleksibilniji, a samim tim i pogodniji za širu klasu problema, su računari SIMD i MIMD tipa.

SIMD RAČUNARI

Računari ove klase sastoje se od *N* identičnih procesora, kao što je prikazano na Slici 1.3. Svaki procesor ima svoju lokalnu memoriju u koju može da smešta i program i podatke. Svi procesori rade pod upravljanjem jedinstvenog niza instrukcija iz centralne upravljačke jedinice ili, što je ekvivalentno, svaki procesor može u svojoj lokalnoj memoriji posedovati identičnu kopiju jednog programa. Postoji *N* tokova podataka, po jedan iz svakog procesora. Svi procesori rade sinhrono. U svakom koraku svi procesori izvršavaju istu instrukciju, svaki nad različitim podatkom.

Ponekad može biti potrebno da samo odredjeni podskup procesora izvršava instrukciju. Ova informacija može biti kodirana samom instrukcijom kazujući procesoru da li treba da bude aktivan (izvršava instrukciju) ili pasivan (čeka sledeću instrukciju).



Slika **1.3.** SIMD računar

Razmena podataka izmedju procesora u SIMD računani može se ostvariti na dva načina, pa, shodno tome, postoje i dve klase ovih računara: SIMD računari kod kojih se komunikacija obavlja preko *deljive memorije* i SIMD računari kod kojih se komunikacija obavlja preko *sprežne mreže.*

Kod prvog modela SIMD računara svi procesori dele zajedničku memoriju. Kada dva procesora žele da komuniciraju, oni to rade preko zajedničke memorije. Neka, na primer, procesor P_l treba da pošalje podatke procesoru *Pj*. Komunikacija se obavlja u dva koraka. Prvo procesor P*i* upisuje podatak u deljivu memoriju u lokaciju čija je adresa poznata procesoru *Pj*. Zatim procesor

Pj čita podatak iz te lokacije.
Za vreme izvršenja paralelnog algoritma *N* procesora pristupa deljivoj memoriji radi čitanja podataka, čitanja i upisa medjurezultata i upisa konačnih rezultata. Osnovni model ove klase računara dozvoljava svim procesorima da pristupaju deljivoj meoriji jednovremeno ako oni pristupaju različitim memorijskim lokacijama. Dalja podela ovih računara može biti učinjena u odnosu na to da li dva, ili više procesora, mogu pristupati istoj memorijskoj lokaciji jednovremeiio. Shodno tome postoje četiri podklase SIMD računara sa zajedničkom memorijom (skraćeno SM SIMD) i to :

1. *Ekskluzivno-čitanje, Ekskluzivni-upis (EREW) SM SIMD računai.* Pristup memorijskoj lokaciji je isključiv. Drugim recima, nije dozvoljeno da dva ili više procesora pristupaju istoj memorijskoj lokaciji, bilo da se radi o upisu ili o čitanju.
2. *Konkurentno-čitanje, Ekskluzivni-upis (CREW) SM SIMD računari.* Više procesora može jednovremeno da čita podatak sa iste memorijske lokacije, ali pravo upisa je ekskluzivno.
3. *Ekskluzivno-čitanje, Konkurentni-upis (ERCW) SM SIMD računari.* Više procesora može da vrši upis u istu memorijsku lokaciju, ali je pravo čitanja ekskluzivno.
4. *Konkurentno-čitanje, Konkurentni-upis (CRCW) SM SIMD računari.* Dozvoljen je višestruki pristup memorijskoj lokaciji i radi čitanja i radi upisa.

Dozvola višestrukog pristupa istoj memorijskoj lokaciji radi čitanja u principu ne uzrokuje nikakve probleme (izuzev možda tehnoloških, o kojima ovde nije reč). Medjutim, problemi nastupaju kada je dozvoljen višestruki pristup radi upisa. Ako nekoliko procesora pokušava jednovremeno da upiše podatke (moguće i različite) u istu memorijsku lokaciju, postavlja se pitanje kome od njih omogućiti pristup. Predloženo je nekoliko politika za razrešenje konflikata oko upisa. Neke od njih su sledeće:

a) Dozvoljava se upis procesoru sa najnižim rednim brojem (tj. adresom), a svim ostalim se zabranjuje pristup.
b) Dozvoljava se pristup svim procesorima pod uslovom da su vrednosti podataka koji se upisuju jednake. U protivnom se zabranjuje pristup svim procesorima.
c) Suma vrednosti svih podataka koje procesori žele da upišu se upisuje.

Prvi model SM SIMD paralelnih računara je bez sumnje najslabiji jer je pristup datoj memorijskoj lokaciji dozvoljen samo jednom procesoru u posmatranom trenutku. Na žalost, i ovaj model izračunavanja je nerealan. Paralelni računar sa proizvoljnim brojem procesora baziran na ovom modelu, ne može biti napravljen jer ovaj model zahteva postojanje memorija koje dozvoljavaju jednovremeni pristup različitim memorijskim lokacijama svim procesorima. Ovakva memorija bila bi i suviše skupa. Jedan način, da se redukuje cena je da se memorija podeli na više blokova, pri čemu bi jednom bloku u datom trenutku mogao da pristupi samo jedan procesor.

Kod druge klase SIMD računara ne postoji zajednička (deljiva) memorija kojoj mogu pristupati svi PEovi. Umesto toga svaki PE poseduje poseban registar za prenos podataka (ili par registara, jedan za ulaz i drugi za izlaz). Ovaj registar je

preko sprežne mreže povezan sa odgovarajućim registrima drugih PEova. Kada se obavlja komunikacija izmedju PEova izmenjuju se sadržaji ovih registara. Veze izmedju PEova mogu biti statičke, tj. dodeljene unapred, bez mogućnosti menjanja (rekorifiguracije) da bi se ostvarila sprega sa drugim PEom, ili dinamičke sa mogućnošću rekonfigurisanja. Shodno tome, može se govoriti o statičkim i dinamičkim sprežnim mrežama kod SIMD sistema. Statičke sprežne mreže mogu biti klasifikovane shodno zahtevanim dimenzijama na: jednodimenzionalne (linearne), dvodimenzionalne (prsten, zvezda, stablo) i trodimenzionalne (potpuno povezane, 3—kub). Dinamičke sprežne mreže mogu se podeliti na jednostepene i višestepene, u zavisnosti od toga koliko stepena komutacionih elemenata koriste.

SIMD način obrade ograničen je na probleme koji se mogu podeliti na skupove identičnih potproblema, koji se zatim jednovremeno rešavaju istim skupom instrukcija. Medjutim, postoji mnogo problema koji se ne mogu resiti na ovaj način. Takvi problemi zahtevaju podelu na potprobleme koji ne moraju biti identični i ne mogu se resiti izvršavanjem istog skupa instrukcija. Da bi se ovi problemi resili paralelno, potrebni su MIMD računari.

MIMD RAČUNARI

Ova klasa računara je najopštija i najmoćnija klasa paralelnih računara. MIMD računar ima *N* procesora, *N* nizova instrukcija i *N* nizova podataka, kao što je prikazano na Slici 1.4. Svi procesori u sistemu su SISD računan u smislu da svaki poseduje sopstvenu upravljačku jedinicu, lokalnu memoriju i aritmetičko logičku jedinicu. Svaki procesor radi pod upravljanjem niza instrukcija koje izdaje njegova jedinica za upravljanje. Procesori mogu izvršavati različite programe nad različitim podacima pri rešavanju različitih potproblema datog problema. Ovo znači da procesori i rade asinhrono. Kao i kod SIMD računara, komunikacija izmedju procesora može se odvijati preko deljive memorije ili sprežne mreže. MIMD računan koji dele zajedničku memoriju obično se zovu multiprocesori (ili čvrsto spregnuti sistemi), a oni kod kojih se komunikacija ostvaruje slanjem poruka kroz sprežnu mrežu mul-tiračunari (ili slabo spregnuti sistemi).

Osnovne karakteristike multiprocesorskog sistema su sledeće:

- Multiprocesor sadrži dva ili više homogenih procesora podjednakih mogućnosti obrade.
- Svi procesori dele pristup zajedničkoj memoriji, mada svaki procesor može imati svoju lokalnu memoriju.
- Komunikacija izmedju procesora ostvaruje se preko deljive memorije korišćenjem zajedničkih promenljivih.
- Svi procesori dele pristup U/I uredjajima.
- Radom celog sistema upravlja jedinstveni operativni sistem.
- Procesori nisu usko specijalizovani.

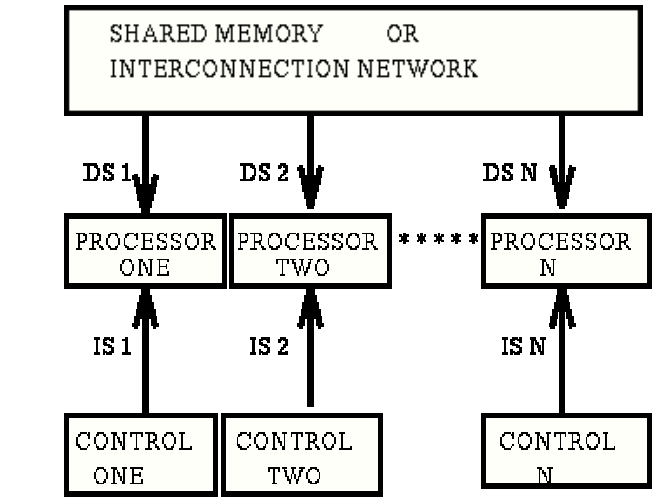
Kod slabo spregnutih MIMD sistema svaki procesor ima svoju lokalnu memoriju i svoj sopstveni skup U/I uredjaja. Procesor sa svojom memorijom i U/I uredjajima zove se računarski modul ili čvor. Skup procesora može biti heterogen (8—bitni, 16-bitni, 32-bitni). Ne postoji zajednička memorija. Komunikacija izmedju procesora ostvaruje se slanjem poruka preko sprežne

mreže.

Može se postaviti pitanje da li postoji razlika izmedju slabo-spregnutog MIMD sistema (multiračunara) i mreže računara (RM). Odgovor je da, a razlike su sledeće:

- Slabo spregnuti MIMD sistem je jedan računar i broj procesora (čvorova) je transparentan za korisnika. Svi procesori u sistemu su angažovani na reša.vanju jednog problema (za korisnika to je virtuelni jednoprocesorski sistem). Radom multiračunara upravlja jedinstveni operativni sistem (OS).
- Kod RM svaki računar ima svoj OS.
- Kod RM korisnik je "svestan" gde se izvršava njegov program. Ako je potreban prenos neke datoteke podataka, korisnik RM mora da zna gde se datoteka nalazi i da izda eksplicitnu komandu za prenos.
- Otkaz jednog računara u RM je vidljiv za. korisnika; otkaz jednog čvora u slabo spregnutom sistemu izazvaće samo degradaciju performansi sistema, ali sistem i dalje može da radi.

Kao što je već napomenuto, MIMD model paralelnog izračunavanja je najopštiji i najmoćniji model. Računati ove klase se koriste za paralelno rešavanje onih problema koji ne poseduju regularne strukture koje zahteva SIMD model.



Slika 1.4. MIMD računar

Kada je reč o MIMD modelu izračunavanja, treba praviti razliku izmedju *procesa i procesora.* Asinhroni algoritam je skup procesa, pri čemu se neki ili svi izvršavaju simultano na odredjenom broju raspoloživih procesora. Inicijalno su svi procesori slobodni. Paralelni algoritam startuje sa izvršavanjem na proizvoljnom procesoru. Nakon toga on kreira više zadataka ili procesa. Kada je proces kreiran, on mora biti izvršen na procesoru. Ako postoji slobodan procesor, proces se dodeljuje procesoru na izvršenje. U protivnom proces se stavlja u red čekanja. Kada procesor izvrši proces, postaje slobodan (raspoloživ). Ako postoje procesi u redu čekanja, jedan od njih se dodeljuje procesoru. U protivnom procesor čeka da proces bude kreiran.

Raspoloživost procesora nije uvek dovoljan uslov da otpočne izvršenje nekog procesa iz reda čekanja. Može se desiti da moraju biti zadovoljeni i neki dodatni uslovi da bi izvršenje procesa moglo da otpočne.

Ovo su samo neki od problema koji se javljaju pri programiranju MIMD računara. Ovi problemi nisu prisutni kod manje fleksibilnih, ali lakših za programiranje, SIMD sistema.

Završavajući razmatranje o modelima izračunavanja prirodno se nameće pitanje: Kako izabrati paralelni računar iz skupa raspoloživih modela? Napomenimo da se jedan model izračunavanja može iskoristiti za simulaciju algoritma projektovanog za drugi model. U suštini, i na jednoprocesorskom sistemu se može izvršiti bilo koji paralelni algoritam. Ovo ukazuje da su svi modeli paralelnih računara ekvivalentni u smislu problema koji *mogu da reše*. Ono što ih medjusobno razlikuje je lakoća i brzina kojom rešavaju odredjeui problem. Zbog toga će oblast aplikacije u kojoj se koristi računar i vreme za koje se očekuje odgovor na dati problem biti faktori pri odlučivanju koji paralelni računar upotrebiti. Medjutim, kao i sa mnogim drugim stvarima u životu, izbor paralelnog računara je uglavnom diktiran ekonomskim faktorima.