TXOSTENA

E1T1
Mikel, Xabier, Sean, Erlantz

AURKIBIDEA

1 SARRERA

Dokumentu tekniko honetan sistema informatiko modernoen oinarrizko exekuzio-paradigmak eta horiek azpiko hardwarean eta sistema eragilean nola kudeatzen diren aztertuko dugu. Hiru atal nagusi ditu lanak: Programazio Motak (Konkurrentea, Paraleloa, Banatua) bereiztea, Prozesuen eta Harien arteko funtsezko desberdintasunak azaltzea, eta Windows eta Linux sistemek exekuzio-unitate horiek nola administratzen dituzten aztertzea. Helburua, informatika-ingurune modernoetan eraginkortasuna eta eskalagarritasuna bermatzen dituzten kontzeptu teorikoak eta aplikazio praktikoak ulertzea da.

2 PROGRAMAZIO MOTAK

2.1 KONKURRENTEA

Programazio konkurrentea problema batzuk aldi berean ebazteko ezarritako metodologia da, eta horrek esan nahi du programazio-lan ugari aldi berean egiten direla, eta ez modu sekuentzialean.

Nabarmendu behar da, metodologia konkurrentean oinarritzen den sistema batean, zereginekin jarraitzeko aukera dagoela, beste jarduera batzuk hasi edo amaitu beharrik gabe.

EZAUGARRI NAGUSIAK

- Aldi bereko exekuzioa: Zereginak denboran tartekatuta egiten dira. Prozesadore batek bere denboraren zati txiki bat eskain diezaioke zeregin bakoitzari, guztiak aldi berean exekutatzen ari direla irudikatuz.
- Independentzia: Hariak edo prozesuak bata bestearengandik independenteki gauza daitezke.
- Multiataza: Programa bati hainbat eragiketa aldi berean egiteko aukera ematen dio, hala nola testu-editore bat, berrikuspen ortografikoa bigarren planoan egiten duen bitartean idazteko aukera ematen duena.
- Hainbat egoera: Prozesadore bakar batean inplementatu daiteke (multiprogramazioaren bidez), hainbat prozesadoretan edo banatutako ordenagailu-sare batean.

APLIKAZIO ESPARRUAK

- Erabiltzaile-Interfaze Grafikoak (UI): Aplikazio bat blokeatzea saihesteko erabiltzen da, adibidez, fitxategi handi bat kargatzen ari den bitartean interfazeak funtzionatzen jarrai dezan.
- Web Zerbitzariak: Bezero anitzei aldi berean erantzuteko ezinbestekoa da, eskaerak tartekatuz.
- Sistema Eragileak: Zeregin anitzeko inguruneak kudeatzeko funtsezkoa da.

ABANTAILAK

- Erantzun-denbora hobetzea: Erabiltzailearen esperientzia hobea lortzen da.
- Baliabideen Erabilera Optimoa: Baliabideen inaktibitate-denbora minimizatzen da.

DESABANTAILAK

- Sinkronizazio-arazoak: Baliabide partekatuetara aldi berean sartzean lasterketa-baldintzak (race conditions) sor daitezke.
- Deadlock-ak: Prozesu batek beste baten baliabidearen zain dagoenean gertatzen den egoera, eta inork ezin du aurrera egin.

2.2 PARALELOA

Programazio paraleloa konputazioaren errendimendua handitzeko metodologia da, zereginak aldi berean (simultaneously) exekutatuz hardware anitzeko ingurune batean (nukleo, prozesadore edo makina anitz). Konkurrentzian ez bezala, hemen helburua ez da denbora tartekatzea, baizik eta denbora murriztea, lana hainbat unitate fisikoren artean banatuz.

Nabarmendu behar da, paralelismoak hardware espezifikoa behar duela (multicore prozesadoreak edo GPUak), kalkulu-intentsitate handiko arazoak azkarrago ebazteko.

EZAUGARRI NAGUSIAK

- Aldi Bereko Exekuzioa (Simultaneoa): Zereginak aldi berean (une fisiko berean) exekutatzen dira, prozesadore edo nukleo ezberdinetan banatuta.
- Hardware Anitza: Sistema multiprozesadoreetan oinarritzen da (memoria partekatua edo banatua duten sistema anitzak).
- Independentzia Egitura: Prozesuak edo hariak egitura horietan gauzatzen dira, eta askotan, datuak zati txikiagoetan banatzen dira, nukleo bakoitzak bere zatia prozesa dezan.
- Errendimendua Handitzea: Helburu nagusia exekuzio-denbora murriztea da, arazo konplexuak askoz azkarrago ebatziz (Speedup lortuz).
- Komunikazio-Kostuak: Nukleoen arteko komunikazioak eta datuen transferentziak denbora eta baliabideak eska ditzakete.

APLIKAZIO ESPARRUAK

- Kalkulu Zientifikoa eta Ingeniaritza: Simulazio konplexuak (adibidez, meteorologia, fisika kuantikoa, fluidodinamika).
- Datu Masiboen Prozesamendua (Big Data): Datu-base eta analisi erraldoien prozesamendua (adibidez, MapReduce bezalako paradigmak).
- Irudi eta Bideo Prozesamendua: Errendatze-lanak, grafikoak eta jokoak (GPUen erabilera).
- Adimen Artifiziala (IA) eta Machine Learning: Eredu konplexuen entrenamendu azkarra, datu-multzo handiekin.

ABANTAILAK

- Abiadura eta Eraginkortasuna: Exekuzio-denbora nabarmen murriztea, zeregin handiak epe laburragoan amaituz.
- Arazo Konplexuak Ebaztea: Bestela ebazteko ezinezkoak edo eraginkortasun gutxikoak liratekeen arazoak konpontzeko gaitasuna.
- Eskalagarritasun Potentziala: Nukleo edo prozesadore gehiago gehituz errendimendua hobetzeko aukera.

DESABANTAILAK

- Hardware Kostua: Ekipamendu garestiagoa behar du (prozesadore, memoria eta interkonexio hobeak).
- Programazio Zailtasuna: Kodea garatzeko zailagoa da, datuen banaketa eta komunikazioa behar bezala kudeatu behar direlako.
- Komunikazio Latentzia: Nukleoen arteko datu-trukean sortzen den atzerapena.
- Amdahl-en Legea: Programaren zati guztiak ezin direnez paraleloki exekutatu, beti egongo da errendimendua mugatuko duen zati sekuentzial bat.

2.3 BANATUA

Programazio banatua, funtsean, programa bat elkarri sare bidez konektatutako makina edo nodo desberdinetan exekutatzean datza. Sistema Banatuetan, makina bakoitzak bere memoria propioa eta prozesadorea du, eta ez dute memoria partekatzen; horregatik, komunikazioa mezu-bidalketaren bidez egiten da.

Helburu nagusiak eskalagarritasuna, erredundantzia eta akatsen tolerantzia dira.

EZAUGARRI NAGUSIAK

- Independentzia Geografikoa eta Fisikoa: Makinak kokapen eta hardware desberdinetan egon daitezke.
- Baliabide Banatuak: Makina bakoitzak bere baliabideak (memoria, CPU, diskoa) kudeatzen ditu.
- Komunikazioa Sare Bidez: Prozesuen arteko komunikazioa mezuak trukatuz egiten da, sareko protokoloak erabiliz (adibidez, TCP/IP).
- Sistema Heterogeneoak: Askotan, makina ezberdinez edo sistema eragile desberdinez osatuta daude.
- Baliabideen Partekatzea: Erabiltzaileek baliabideak partekatzeko aukera dute (fitxategiak, inprimagailuak, datu-baseak) sareko protokoloen bidez.

APLIKAZIO ESPARRUAK

- Hodeiko Konputazioa (Cloud Computing): Amazon Web Services (AWS), Google Cloud edo Azure bezalako plataformak.
- Banatutako Datu-Baseak: Datu-base handien kudeaketa, non datuak zerbitzari desberdinetan gordetzen diren.
- Web Zerbitzu Handiak: Google edo Facebook bezalako plataforma masiboak, karga banatzeko.
- Blockchain Teknologiak: Sareko nodo desberdinetan erregistroak modu deszentralizatuan mantentzea.
- Sareko Fitxategi-Sistemak: Erabiltzaileek fitxategi berdinak atzi ditzakete hainbat makinatik.

ABANTAILAK

- Eskalagarritasun Handia: Sisteman makina gehiago erraz gehi daitezke, eskaera handiagoei erantzuteko.
- Akatsen Tolerantzia: Makina batek huts egiten badu, gainerakoek lanean jarrai dezakete (erredundantzia).
- Baliabideen Aprobetxamendua: Baliabide lokal eta geografiko desberdinak erabil daitezke.
- Kostuen Eraginkortasuna: Ordenagailu pertsonal merkeak erabil daitezke superkonputagailu garestien ordez.

DESABANTAILAK

- Komunikazio Latentzia: Sarean zehar bidaltzen diren mezuen atzerapenak errendimendua mugatu dezake.
- Koordinazio Zailtasunak: Makina desberdinen artean denbora eta egoera sinkronizatzea oso konplexua izan daiteke.
- Segurtasun Erronkak: Sarearen bidezko komunikazioak zaurgarritasunak eta segurtasun-arriskuak handitzen ditu.
- Software Zailtasuna: Sistema banatuak kudeatzen dituen softwarea garatzea eta mantentzea konplexua da.

3 PROZESUAK ETA HARIAK

Sistema eragile baten oinarrian, bi kontzeptu nagusi daude lana kudeatzeko: prozesua eta haria (thread). Biak exekuzio-unitateak dira, baina erabat desberdinak dira euren baliabideen kudeaketan eta inplementazio-kostuan.

3.1 PROZESUAK

Prozesua exekutatzen ari den programa baten instantzia bat da. Sistema eragileak kudeatzen duen oinarrizko unitatea da, eta baliabide multzo oso bat du bereizita.

EZAUGARRI NAGUSIAK

- Memoria Espazio Bereizia: Prozesu bakoitzak bere memoria-helbideen espazio propioa du. Horrek esan nahi du prozesu batek ezin duela zuzenean beste prozesu baten memorian sartu, eta elkarrengandik isolatuta daude.
- Baliabideak: Prozesu bakoitzak bere baliabide multzoa du: memoria-pilak (heap), pila (stack), fitxategi irekiak eta segurtasun-informazioa.
- Independentzia: Akats batek prozesu batean bakarrik eragingo du; gainerako prozesuak, teorian, ez dira kaltetuak izango isolamendu horri esker.
- Sorkuntza-Kostua: Prozesu berri bat sortzea denbora eta baliabide asko kontsumitzen dituen operazioa da (baliabideen kopia egin behar da, memoria esleitu...).

3.2 HARIAK

Haria prozesu baten barruko exekuzio-unitatea da. Prozesu bat zeregin anitzetan banatzeko modua ematen du, eta hariaren helburua prozesuaren konkurrentzia edo paralelismoa ustiatzea da.

EZAUGARRI NAGUSIAK

- Memoria Partekatua: Hari guztiak prozesu baten barruan bizi dira, eta ondorioz, prozesu horren memoria-espazio bera partekatzen dute (heap eta kode-segmentua). Bakoitzak bere pila (stack) du.
- Baliabide Partekatzaileak: Prozesuaren fitxategiak, socket-ak eta kodea partekatzen dituzte.
- Independentzia txikiagoa: Hari batek akats bat sortzen badu, prozesu osoa (hari guztiak) eror daitezke.
- Sorkuntza-Kostua: Hari berri bat sortzea kostu txikikoa da, memoria-espazio berria sortu ordez, bakarrik pila berri bat eta erregistro gutxi batzuk esleitu behar direlako.

3.3 ERLAZIOA

Prozesuak eta hariak elkarrekin lanean aritzeko eta datuak trukatzeko komunikatu behar dira, baina isolamendu-maila desberdinak dituztenez, komunikazio-mekanismoak ere desberdinak dira.

PROZESUEN ARTEKO KOMUNIKAZIOA (IPC)

Sistema eragileak berak hornitu behar du, prozesuak beren memoria-espazioetan erabat isolatuta daudelako. IPCk bi eredu nagusi ditu: Mezu-Trukaketa eta Memoria Partekatua. Mezu-trukaketaren bidez, prozesuek informazioa bidaltzen dute Tutuak (Pipes), Mezu-Ilarak edo Sareko Socket-ak erabiliz. Bestalde, Memoria Partekatua IPC mekanismo azkarrena da, non hainbat prozesuk memoria-eskualde bera atzitu dezaketen, baina prozesuek eurek kudeatu behar dute sarbidea sinkronizazio-mekanismoen bidez, datuen osotasuna bermatzeko.

HARIEN ARTEKO KOMUNIKAZIOA

Errazagoa da, hari guztiak prozesu baten barruan bizi eta memoria-espazio bera partekatzen dutelako. Komunikazio-bide naturala memoria partekatua den arren, hainbat harik aldi berean datu berak aldatzea saihesteko, ezinbestekoa da Sinkronizazio Mekanismoak erabiltzea. Mekanismo horien artean daude Mutex-ak (Mutual Exclusion), baliabide kritiko bat hari bakar batek atzitu ahal izatea bermatzen dutenak, eta Semaforoak, sarbide-kopuru mugatu bat kontrolatzeko balio dutenak, datu-inkonsistentziak saihestuz eta exekuzio konkurrente segurua ahalbidetuz.

4 PROZESUAK KUDEATZEKO ERAK

4.1 WINDOWS

Windows-en prozesuen administrazioa oso hierarkikoa eta objektuetan oinarritua dago. Sistema eragile honek bereizketa zorrotza egiten du baliabide-edukiontzia (Prozesua) eta exekuzio-unitatearen (Haria) artean.

EGITURA ETA KONTZEPTU ADMINISTRATIBOAK

- Prozesua: Prozesu bakoitza objektu bat da, eta berezko baliabideak ditu (helbide-espazio pribatua, handle-ak, etab.). Prozesu bat amaitzen denean, bere baliabide guztiak (hari guztiak barne) askatzen dira.
- Haria (Thread): Prozesu baten barruko unitate exekutagarri txikiena da. Hariak kudeatzailearen bidez (kernel) sortu, eseki eta amaitu daitezke. Prozesu bakoitzak gutxienez hari bat izan behar du.
- Job (Lan): Prozesu multzo bat batera kudeatzeko eta baliabide-mugak (memoria edo exekuzio-denbora) ezartzeko erabiltzen den edukiontzia.

PLANIFIKAZIO-POLITIKA (SCHEDULING)

- Lehentasunetan Oinarritutako Preemptive-a: Windows-ek 32 lehentasun-maila erabiltzen ditu hariak antolatzeko. Planifikatzaileak beti ematen dio CPUa lehentasun handiena duen hariari.
- Planifikazio Dinamikoa: Hariek beren lehentasuna alda dezakete exekuzio-garaian (adibidez, I/O eragiketa baten ondoren denbora laburrez igotzea).
- Erantzuna Bermatzea: Erabiltzailearen interfaze-hariak (UI) besteak baino gehiago lehenesten dira, interfazea blokeatuta ez geratzeko.

ADMINISTRAZIO-TRESNAK

- Task Manager (Ataza-Kudeatzailea): Erabiltzaileentzako tresna grafiko nagusia, prozesuak eta baliabideen erabilera ikusteko, eta harien lehentasunak eskuz aldatzeko.
- Process Explorer / Process Monitor: Tresna aurreratuagoak, prozesuen handle-ak eta DLL-en erabileraren jarraipena egiteko.

4.2 LINUX

Linux-en prozesuen administrazioa sinpleagoa eta malguagoa izan ohi da, batez ere Unix filosofian oinarritzen delako, non dena fitxategi edo zeregin gisa tratatzen den.

EGITURA ETA KONTZEPTU ADMINISTRATIBOAK

- Prozesua / Zeregina (Task): Kernel mailan, Linux-ek prozesu arinak (lightweight processes / LWP) sortzen ditu. Prozesuak eta Hariak Zeregin gisa kudeatzen dira funtsean, baina partekatzen dituzten baliabide kopuruan bereizten dira (clone() sistema-deiaren bidez sortuak izan diren).
- PID eta TID: Prozesu guztiek Prozesu Identifikatzaile (PID) bakarra dute. Hariak, berriz, Zeregin Identifikatzaile (TID) batekin identifikatzen dira, baina PID bera parteka dezakete.

PLANIFIKAZIO-POLITIKA (SCHEDULING)

- Completely Fair Scheduler (CFS): Linux-en planifikatzaile modernoen erregina. CFS-aren helburua ez da lehentasun handienak dituen zeregina exekutatzea, baizik eta justizia (fairness) ziurtatzea.
- Denbora Birtuala: CFS-k zeregin bakoitzari eman behar zaion CPU denbora kalkulatzen du (denbora birtuala), eta beti exekutatzen du denbora gutxien erabili duen zeregina. Horrela, CPU baliabideak modu justuan banatzen direla ziurtatzen da, starvation-a saihestuz.
- Lehentasun Eragina: Lehentasun handiagoak (nice value txikiagoa) dituzten zereginek denbora birtuala mantsoago igotzea eragiten dute, eta horrela maizago exekutatzen dira.

ADMINISTRAZIO-TRESNAK

- ps (Process Status): Prozesuen zerrenda estatikoa eta haien PID, estatu eta baliabide-erabilera erakusteko.
- top / htop: Prozesuen eta baliabideen erabilera dinamikoa, denbora errealean, monitorizatzeko (kontsolako tresna nagusiak).
- kill / killall: Prozesuak beren PID edo izenaren bidez gelditu edo seinaleak bidaltzeko erabiltzen diren komandoak.

5 ONDORIOAK

Gaur egungo konputazio-sistemak ezin dira ulertu hiru paradigma nagusi horien konbinaziorik gabe, errendimendu, erantzun-denbora eta akats-tolerantzia hobetzeko ezinbestekoak baitira. Dokumentu honen bidez, ondorioztatu dugu prozesu eta harien arteko hautaketa zereginaren izaeraren (isolamendua vs. komunikazio arina) araberakoa dela beti.

Batez ere, Linux-ek CFS bidez duen planifikazio-ereduak eraginkortasunaren eta justiziaren arteko oreka lortzen du, ingurune anitzeko lan-kargetarako egokia dena. Azken batean, lan honetan jasotako ezagutza kritikoa da goi-mailako softwarea diseinatzeko, non garatzaileek aukeratutako paradigma eta sistema eragilearen muga eta aukerak ulertu behar dituzten aplikazio eraginkorrak eta fidagarriak eraikitzeko.

6 BIBLIOGRAFIA

CódigoFacilito. (s.f.). Threads y procesos. https://codigofacilito.com/articulos/threads-procesos

CódigoFacilito. (s.f.). Programación concurrente. https://codigofacilito.com/articulos/programacion-concurrente

El Baúl del Programador. (s.f.). Introducción a los procesos. https://elbauldelprogramador.com/introduccion-los-procesos/#:~:text=Caracter%C3%ADsticas,representar%20en%20forma%20de%20%C3%A1rbol.

Ferestrepoca. (s.f.). Paradigmas de Programación. https://ferestrepoca.github.io/paradigmas-de-programacion/paralela/paralela_teoria/index.html

KeepCoding. (s.f.). ¿Qué es la Programación Concurrente? https://keepcoding.io/blog/que-es-la-programacion-concurrente/#:~:text=La%20programaci%C3%B3n%20concurrente%20se%20define,iniciar%20o%20finalizar%20otras%20actividades.&text=Las%20labores%20de%20la%20programaci%C3%B3n,o%20bien%20en%20diferentes%20procesadores.

KeepCoding. (s.f.). Ventajas y desventajas de la programación concurrente. https://keepcoding.io/blog/ventajas-y-desventajas-programacion-concurrente/#:~:text=Como%20una%20de%20las%20ventajas,ser%C3%ADan%20procesadas%20a%20la%20vez.

KeepCoding. (s.f.). Ventajas y desventajas del paralelismo. https://keepcoding.io/blog/ventajas-y-desventajas-paralelismo/

KeepCoding. (s.f.). ¿Qué es la Programación Modular y cómo funciona? https://keepcoding.io/blog/que-es-la-programacion-modular-y-como-funciona/#:~:text=Esta%20t%C3%A9cnica%20no%20solo%20facilita,el%20trabajo%20de%20los%20dem%C3%A1s.

Lifeder. (s.f.). Programación modular. https://www.lifeder.com/programacion-modular/

Scribd. (2023). Administración de procesos Windows y Linux. https://es.scribd.com/document/629205152/Administracion-de-procesos-Windows-y-Linux#:~:text=Windows%20y%20Linux)-,El%20documento%20resume%20c%C3%B3mo%20se%20administran%20los%20procesos%20en%20Windows,interfaces%20gr%C3%A1ficas%20para%20administrar%20procesos-,El%20documento%20resume%20c%C3%B3mo%20se%20administran%20los%20procesos%20en%20Windows,interfaces%20gr%C3%A1ficas%20para%20administrar%20procesos).

Unir. (s.f.). Computación paralela. https://www.unir.net/revista/ingenieria/computacion-paralela/

Wayakna. (s.f.). Programación modular: cómo mejora el desarrollo de software. https://arangoya.org/programacion-modular-como-mejora-el-desarrollo-de-software/