Konkurrentzia.

Sarrera

Programak, prozesuak eta konkurrentzia

- **Algoritmoa**: Problema jakin bat ebazten duen metodoa
- **Programa**: Algoritmo kodetua. Agindu-multzoa. Estatikoa.
- **Prozesua**: Programa bat exekuzioan. Dinamikoa.

- **Konkurrentzia**:

  Bi prozesu konkurrenteak dira,
  haietako baten lehenengo agindua egikaritzen bada
  - bestearen lehenengo aginduaren ondoren
  - eta azkenekoaren aurretik.

Programa sekuentzialak vs programa konkurrenteak

Programa sekuentzial
baten exekuzioa
↓
Bata bestearen ondoren egikaritzen diren aginduen sekuentzia
↓
prozesu bat

kontrol hari bakar bat

Programa konkurrente
baten exekuzioa
↓
Agindu sekuentzialez osatutako hainbat multzoren aldibereko exekuzioa
↓
prozesu anitz

hainbat kontrol hari

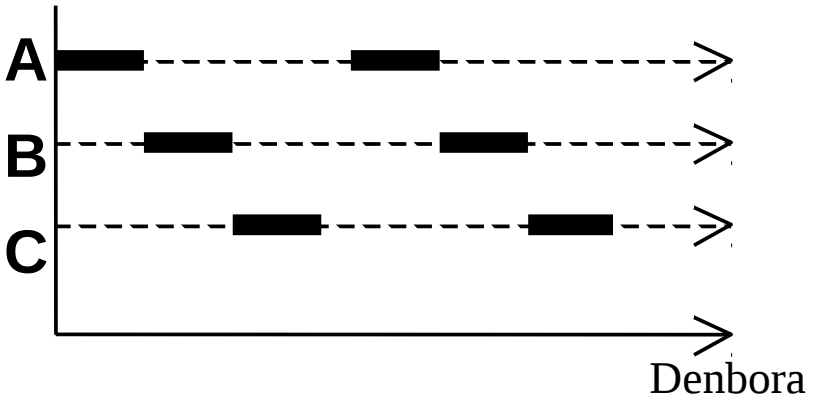
haria : thread

Zergatik programazio konkurrentea?

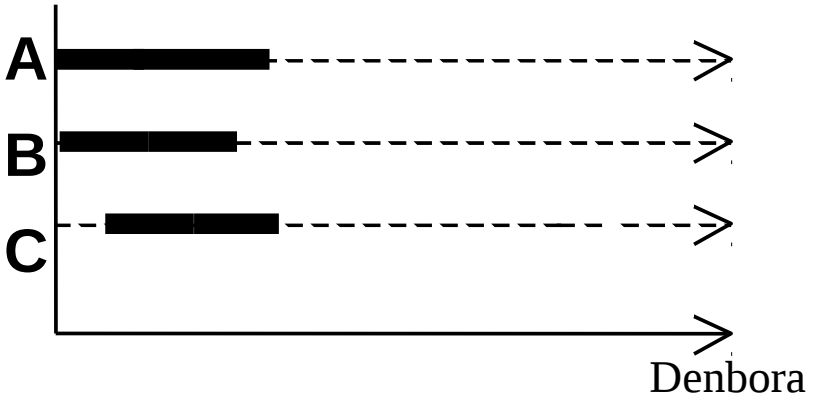
- Prozesadore anitzekin **paralelismoa** lortzeko
- Aplikazioaren **errendimendua** handitzeko
  - S/I dei batek hari bakar bat blokeatzen du.
- Aplikazioaren **erantzun-gaitasuna** handitzeko
  - Erabiltzaileen eskaerentzat lehentasun handiko haria
- Egitura **egokiagoa**
  - Ingurunearekin elkarreragiten duten, ekintza anitz kontrolatzen dituzten eta gertaera anitz erabiltzen dituzten programetarako
  - Kontrol-sistemak, web-teknologiak erabiltzaile-interfazeetan oinarritutako aplikazioak, simulazioa, DBKS,…

Konkurrentzia eta paralelismoa

- ***Konkurrentzia***
  - **Aldibereko prozesaketa *logikoa***.
    Ez du prozesaketa anitzerako elementurik behar.
    Prozesadore bakarrean tartekaturiko exekuzioa behar du.



- ***Paralelismoa***
  - **Aldi bereko prozesaketa *fisikoa.***
    Prozesadore anitz behar ditu.



Bai konkurrentziak eta bai paralelismoak konpartituriko baliabideen atzipen kontrolatua behar dute.

Sistema konkurrenteen ezaugarriak

- **Aginduen egikaritze-ordena**
  - Prozesu ezberdinetako aginduak exekutatzeko

    ordena erlatiboa arbitrarioa da,

    baina prozesu bakoitzeko aginduak exekutatzeko ordena mantentzen da
    - Tartekatzea (interleaving)
- **Indeterminismoa**
  - Sarrerako datu multzo berdinaren gainean egikaritzean,

    emaitza ezberdinak eman ditzake exekuzio diferenteetan
- **Exekuzio-denbora**
  - Abiadura arbitrarioa (denboraz ahaztu egiten gara)
    - Asinkronoa

Zer egikaritu daiteke konkurrenteki?

```
x = x+1;
y = x+2;
```

Ez

```
x = 1;
y = 2;
```

Bai

Bernstein-en baldintzak:

- $Ir(S_i) \cap Id(S_j) = \emptyset$
- $Id(S_i) \cap Ir(S_j) = \emptyset$
- $Id(S_i) \cap Id(S_j) = \emptyset$

non:

- $Ir(S_k)=\{a_1, a_2,\ldots, a_n\}$
  S_k aginduaren irakurketa multzoa: agindu horren exekuzioan irakurtzen diren aldagaiak
- $Id(S_k)=\{b_1, b_2,\ldots, b_m\}$
  S_k aginduaren idazketa multzoa: agindu horren exekuzioan idazten diren aldagaiak

Ariketa:
Ondoko aginduetatik zeintzuk ahal dira konkurrenteki egikaritu?

```
S1  a = x + y;
S2  b = z - 1;
S3  c = a - b;
S4  w = c + 1;
```

Ekintza atomiko batek
prozesu baten egoeran aldaketa zatiezinak egiten ditu.

Programa konkurrenteen zuzentasuna

Programa konkurrente bat zuzena izango da,
betebeharreko eskakizun funtzionalak betetzeaz gain,
ondoko propietateak betetzen baditu:

- **Segurtasun-propietateak**: ez da **ezer** txarrik gertatzen
  - Elkar-bazterketa
  - Baldintzen sinkronizazioa
  - Elkar-blokeaketa (deadlock)
- **Bizitasun-propietateak**: zerbait **ona** *noizbait* gertatzen da.
  - Elkar-blokeaketa aktiboa (livelock)
  - Gosez hiltzea (starvation)