Departament d'Arquitectura de Computadors

UNIVERSITAT POLITÈCNICA DE CATALUNYA

# Conceptes Avançats de Sistemes Operatius

Facultat d'Informàtica de Barcelona
Dept. d'Arquitectura de Computadors

Curs 2018/19 Q2

## Sincronització / Avaluació / Virtualització

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

# Índex

- Sincronització
  - Mutex / condition variables / futex / GCD
- Avaluació de rendiment
- Virtualització

# Sincronització

- Pthreads
  - Mutex
  - Variables de condició (condition variables)
- Interfície de sistema (Linux)
  - Futex
- Grand Central Dispatch (Mac OS-X)

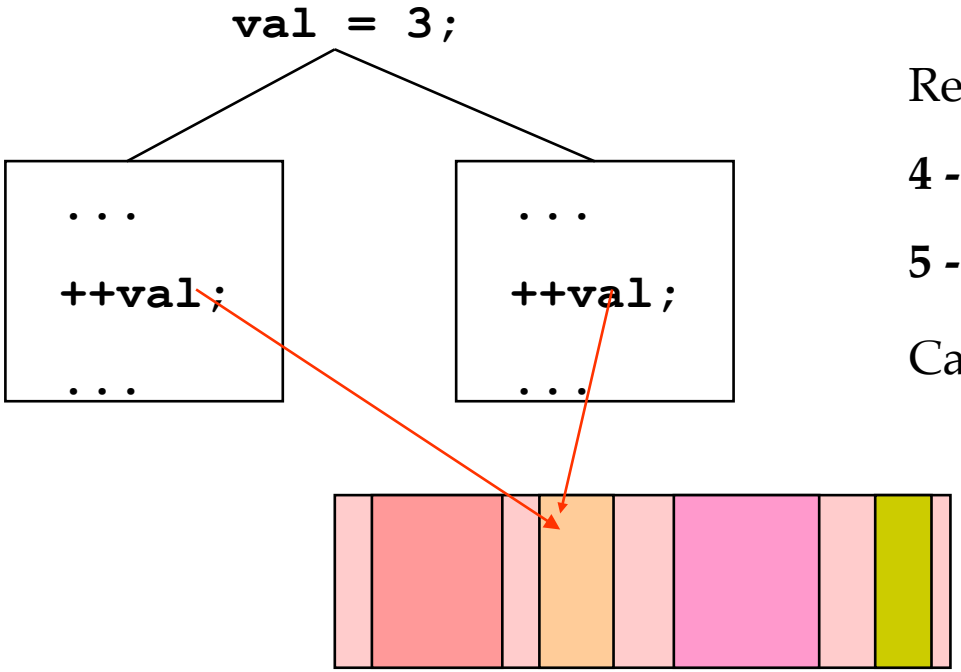
http://www.0x04.net/doc/posix/Multi-Threaded Programming with POSIX Threads - Linux Systems Programming.pdf

http://www.0x04.net/doc/posix/
Redhat - The Native POSIX Thread Library for Linux (Paper) - 2003 - (By Laxxuss).pdf
Addison-Wesley, 1997 Programming with Posix Threads

http://locklessinc.com/articles/futex_cheat_sheet/

# Pthread mutex

- Variables d'exclusió mútua
  - Donen suport a la sincronització en l'accés a variables compartides




Resultats possibles

**4 - un sol increment - incorrecte**

**5 - dos increments - correcte**

Cal exclusió mútua!!!

# Pthread mutex

- Inicialització d'un mutex

```
NAME
  pthread_mutex_init - Initializes a mutex with attributes
                       specified by the attr argument.
SYNOPSIS

  #include <pthread.h>
  int pthread_mutex_init(
       pthread_mutex_t       * mutex ,
       pthread_mutexattr_t   * attr );

PARAMETERS

  mutex
       Mutex created.

  attr Mutex attributes object that defines the characteristics of
       the created mutex. If you specify NULL, default attributes
       are used.
```

PTHREAD_MUTEX_NORMAL
PTHREAD_MUTEX_ERRORCHECK
PTHREAD_MUTEX_RECURSIVE

# Pthread mutex

- Aconseguir l'exclusió mútua

```
NAME
  pthread_mutex_lock - Locks an unlocked mutex. If the mutex is
                       already locked, the calling thread blocks
                       until the mutex becomes available.
SYNOPSIS

  #include <pthread.h>
  int pthread_mutex_lock(
       pthread_mutex_t *                    mutex );

PARAMETERS

  mutex
       Mutex to be locked.
```

# Pthread mutex

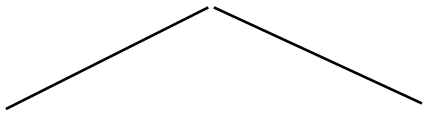
- Sortir de l'exclusió mútua

```
NAME
  pthread_mutex_unlock - Unlocks a mutex.

SYNOPSIS

  #include <pthread.h>
  int pthread_mutex_unlock(
       phread_mutex_t *                    mutex );

PARAMETERS
mutex
       Mutex to be unlocked.
```

# Pthread mutex

- Exemple d'ús
  - Inicialització

```
res = pthread_mutex_init (&mutex, attr);
if (res!=0) {
   fprintf (stderr, "pthread_mutex_init: %s\n",
            strerror (res));
   exit (1);
}
val = 3;
```



# Pthread mutex

- Exemple d'ús

```
...
res = pthread_mutex_lock (&mutex);
if (res!=0) {
   fprintf (stderr, "pthread_mutex_lock: %s\n",
            strerror (res));
   exit (1);
}
++val;
res = pthread_mutex_unlock (&mutex);
...
```

# Pthread condvar

- Variables de condició
  - Permeten que diversos fluxos esperin fins que el resultat d'avaluar una condició prengui un valor determinat
  - Sense fer esperes actives
  - Es combinen amb els mutex

# Pthread condvar

- Inicialització de la variable de condició

```
pthread_cond_t cond;
...
res = pthread_cond_init (&cond, attr);
if (res!=0) {
   fprintf (stderr, “pthread_cond_init: %s\n”,
            strerror (res);
   exit (1);
}
```

# Pthread condvar

- Estructura d'ús habitual

```
res = pthread_mutex_lock (&mutex);
while (!expr) {
   res = pthread_cond_wait (&cond, &mutex);
}
// La condició es compleix i som en exclusió mútua
// Regió crítica
res = pthread_cond_signal (&cond);
res = pthread_mutex_unlock (&mutex);
```

- pthread_cond_wait
  - de forma atòmica, allibera el mutex i es bloqueja sobre 'cond'
  - en despertar-se per un pthread_cond_signal, torna a agafar el mutex

# Pthread condvar

- pthread_cond_signal (&cond);
  - Desperta un dels fluxos bloquejats
- pthread_cond_broadcast (&cond);
  - Desperta tots els fluxos bloquejats

# Linux *futex*

- Interfície mixte entre usuari i sistema
- Cas efficient – si no hi ha contenció
  - nivell usuari

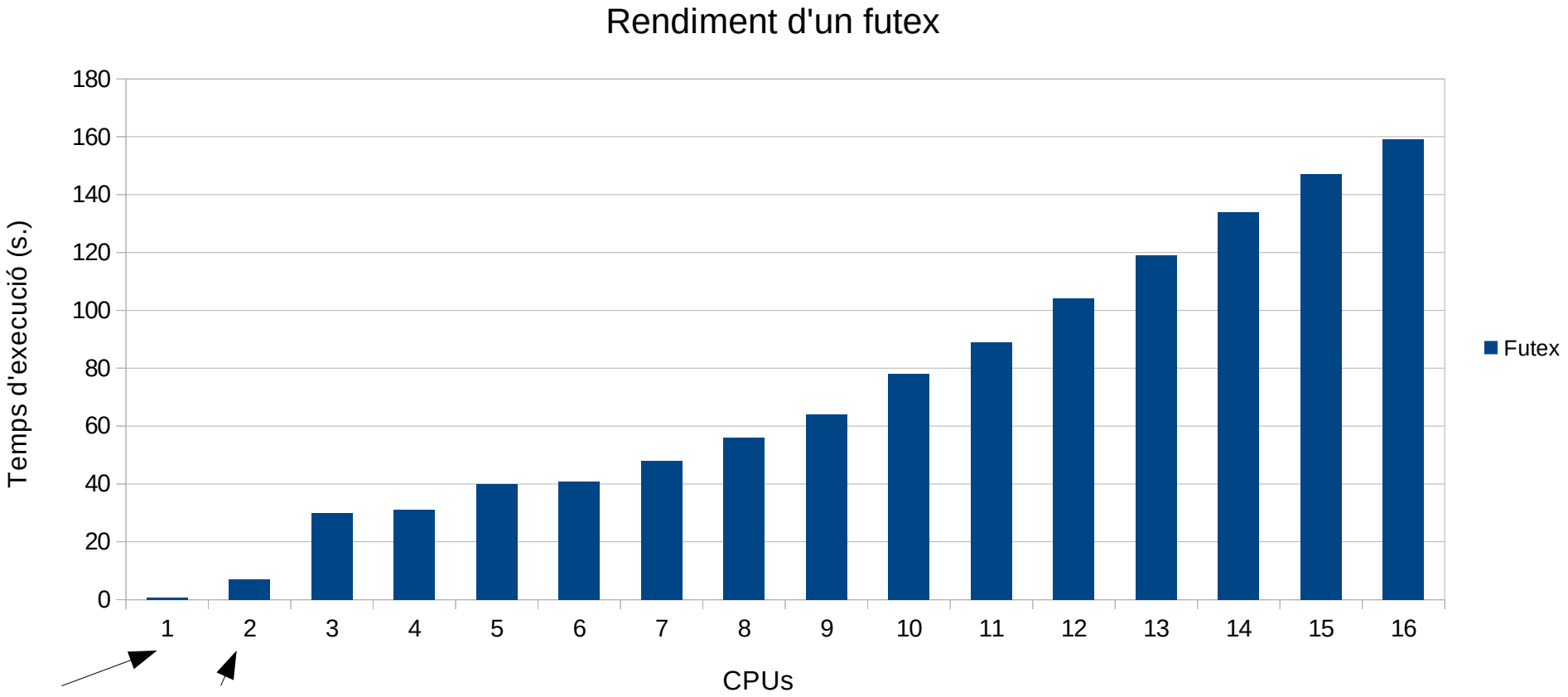
```
// inicialització
futex_lock = 1;
```

```
// Entrada a la regió crítica
while (__sync_sub_and_fetch(&futex_lock, 1) < 0)
   {  futex_block(&futex_lock);   }
```

```
// Sortida de la regió crítica
if (__sync_add_and_fetch(&futex_lock, 1) < 1)
   { futex_unblock(&futex_lock);  }
```

# Rendiment dels *futex*

- Incrementant el volum de treball



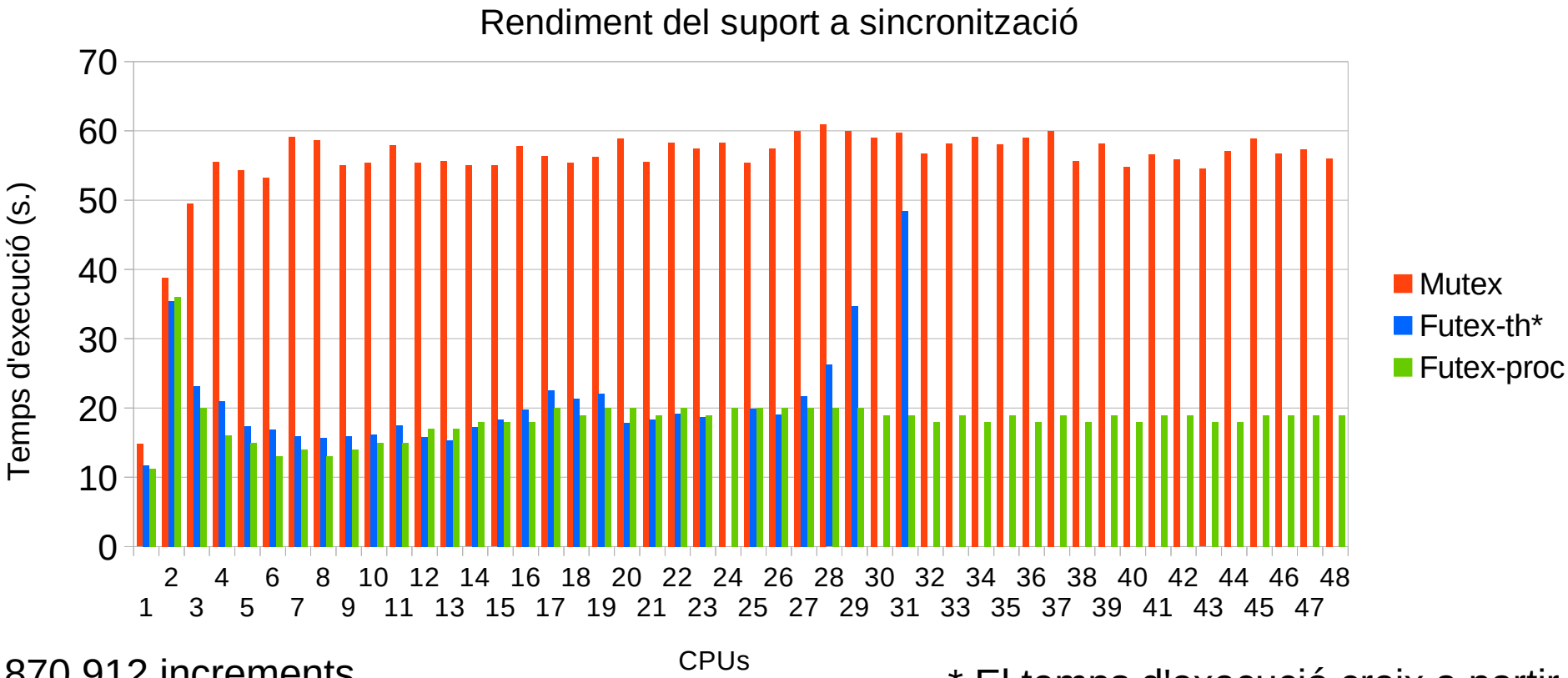

536.870.912 inc.

1.073.741.824 inc.

...

# Rendiment dels *futex*

- Dividint el treball entre els processos
- Comparant amb fluxos i pthread_mutex / futex




536.870.912 increments

48 AMD Opteron 6172 2.1Ghz - KTH

* El temps d'execució creix a partir dels 28 processadors

# Rendiment dels *futex*

- Comportament dels fluxos

```
top - 09:00:26 up 87 days, 20:12, 20 users,  load average: 12.40, 11.01, 8.63
Tasks: 1205 total,  20 running, 1181 sleeping,   4 stopped,   0 zombie
Cpu(s):  2.0%us, 34.3%sy,  0.0%ni, 60.0%id,  3.7%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  65969452k total, 21020344k used, 44949108k free,   693268k buffers
Swap: 16777208k total,  3428472k used, 13348736k free, 14266856k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  WCHAN     COMMAND
 9467 g_xavim   20   0  3916  356  288 R 93.4  0.0   0:04.86 -         wait_fute
 9473 g_xavim   20   0  3916  356  288 R 93.4  0.0   0:04.88 -         wait_fute
 9465 g_xavim   20   0  3916  356  288 R 93.1  0.0   0:04.85 -         wait_fute
 9466 g_xavim   20   0  3916  356  288 R 93.1  0.0   0:04.86 -         wait_fute
 9470 g_xavim   20   0  3916  356  288 R 93.1  0.0   0:04.87 -         wait_fute
 9459 g_xavim   20   0  3916  352  288 R 92.8  0.0   0:04.83 -         wait_fute
 9460 g_xavim   20   0  3916  356  288 R 92.8  0.0   0:04.84 futex_wai wait_fute
 9461 g_xavim   20   0  3916  356  288 R 92.8  0.0   0:04.87 -         wait_fute
 9464 g_xavim   20   0  3916  356  288 R 92.8  0.0   0:04.85 -         wait_fute
 9468 g_xavim   20   0  3916  356  288 R 92.8  0.0   0:04.83 -         wait_fute
...
```

# Rendiment dels *futex*

- Comportament dels processos

top - 09:02:10 up 87 days, 20:14, 20 users,  load average: 14.08, 11.90, 9.20
Tasks: 1099 total,  11 running, 1084 sleeping,   4 stopped,   0 zombie
Cpu(s):  0.1%us, 25.3%sy,  0.0%ni, 59.7%id, 14.9%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  65969452k total, 21021124k used, 44948328k free,   693292k buffers
Swap: 16777208k total,  3428472k used, 13348736k free, 14266860k cached

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | WCHAN | COMMAND |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 9526 | g_xavim | 20 | 0 | 3916 | 356 | 288 | D | 70.4 | 0.0 | 0:23.20 | - | wait_fute |
| 9529 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 69.7 | 0.0 | 0:23.02 | blk_backi | wait_fute |
| 9522 | g_xavim | 20 | 0 | 3916 | 352 | 288 | R | 69.4 | 0.0 | 0:23.31 | sync_page | wait_fute |
| 9524 | g_xavim | 20 | 0 | 3916 | 356 | 288 | D | 69.4 | 0.0 | 0:23.38 | sync_page | wait_fute |
| 9528 | g_xavim | 20 | 0 | 3916 | 356 | 288 | D | 69.4 | 0.0 | 0:23.23 | sync_page | wait_fute |
| 9536 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 69.4 | 0.0 | 0:23.24 | sync_page | wait_fute |
| 9525 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 68.7 | 0.0 | 0:23.30 | sync_page | wait_fute |
| 9527 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 68.7 | 0.0 | 0:23.19 | - | wait_fute |
| 9537 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 68.7 | 0.0 | 0:23.23 | - | wait_fute |
| 9539 | g_xavim | 20 | 0 | 3916 | 356 | 288 | D | 68.7 | 0.0 | 0:23.11 | - | wait_fute |
| 9523 | g_xavim | 20 | 0 | 3916 | 356 | 288 | R | 68.4 | 0.0 | 0:22.96 | sync_page | wait_fute |
| 9535 | g_xavim | 20 | 0 | 3916 | 356 | 288 | D | 68.4 | 0.0 | 0:23.23 | sync_page | wait_fute |

# Grand Central Dispatch

- Mac OS/X
- Substitueix la interfície de pthreads
  - Pot implementar-se a sobre pthreads
- Versions per Linux

**https://github.com/nickhutchinson/libdispatch**

http://developer.apple.com/mac/articles/cocoa/introblocksgcd.html

http://developer.apple.com/mac/library/documentation/General/Conceptual/ConcurrencyProgrammingGuide/Introduction/Introduction.html

http://developer.apple.com/mac/library/documentation/Performance/Reference/GCD_libdispatch_Ref/Reference/reference.html

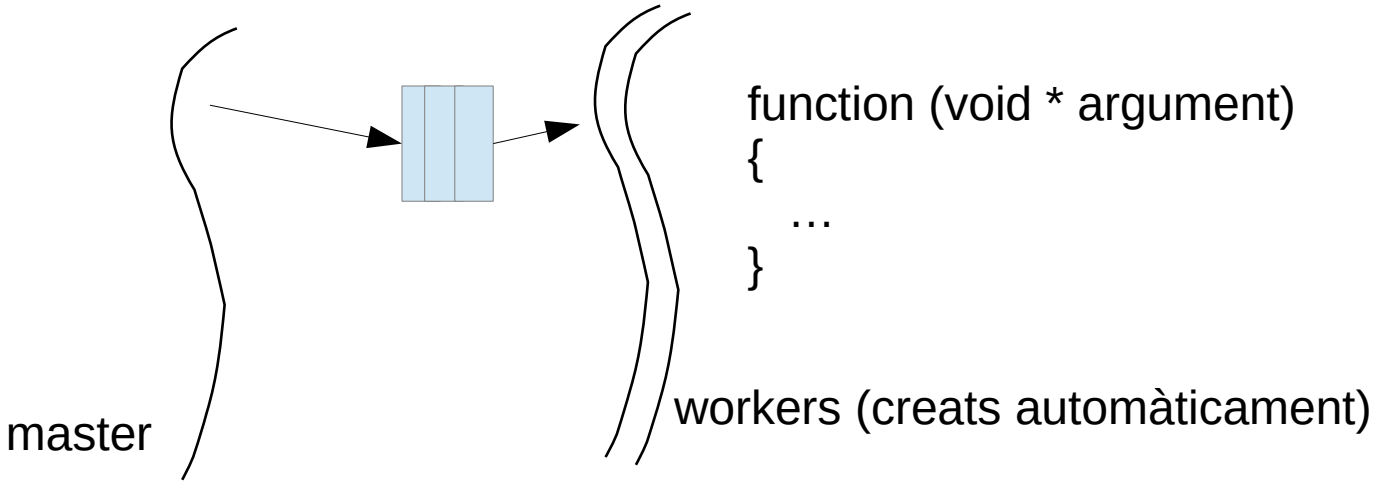
# Grand Central Dispatch

- Interfície basada en cues de treball
  - El programa principal crea una sèrie de cues

```
queue = dispatch_queue_create("com.apple.libdispatch.test_readsync", NULL);
```

  - I encua funcions per executar

```
dispatch_async_f(queue, argument, function);
```



# Grand Central Dispatch

- Els *workers* es creen automàticament
- L'aplicació acaba cridant al planificador de GCD

```
dispatch_main();
```

  - Buida les cues i acaba quan no hi ha més feina per fer
  - Veure: WWDC15

# Índex

- Sincronització
  - Mutex / condition variables / futex / GCD
- Avaluació de rendiment
- Virtualització

# Avaluació de rendiment

- Mètriques
  - Temps d'execució
  - Acceleració (speedup)
    - la relació entre el temps d'execució en seqüencial i el temps d'execució en paral·lel
  - Ample de banda (bandwidth)
    - la relació entre les dades transmeses i el temps que s'ha invertit en transmetre-les
  - Latència
    - Cost d'iniciar operacions o comunicacions

# Avaluació de rendiment

- Ús d'estadístics
  - Mitjana: tenir en compte la variabilitat de les mesures
    - Repetir els experiments un cert número de vegades i fer la mitjana
  - Desviació estàndard: mostra la dispersió dels resultats, respecte la mitjana

# Avaluació de rendiment

- Ús d'estadístics
  - Temps d'execució: fer la mitjana dels N resultats obtinguts
  - Speedup: relació entre mitjanes
    - mitjana seqüencial / mitjana paral·lel
  - Bandwidth: mitjana del bandwidth obtingut en N experiments diferents
  - Latència: mitjana de les latències obtingudes en N experiments diferents

# Avaluació de rendiment

- Eines de sistema
  - top, htop, ps, time, /usr/bin/time
  - vmstat, iostat

```
$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
 0  0    224 358856  54624 816732    0    0     0     0  150  162  0  1 100  0
 0  0    224 358856  54624 816732    0    0     0     0  149  155  1  0 100  0
 0  2    224 187892  54628 983756    0    0    56 84944  520  439  1 11 67 21
 0  2    224 134152  54628 1036116    0    0     0 21592  420  430  1  3 26 71
 0  2    228  97716  53540 1072712    0    4     0 41476  804  909  2  2 25 72
 0  2   2272 121468  49720 1052416    0 2044     0 23036  383  323  1  2 40 58
 0  2   2272 139512  48884 1035992    0    0     0 31744  630  523  0  2 23 76
 0  3   2272 122772  48884 1052376    0    0     0 31232  450  531  1  1 33 66
 0  2   2272 143052  47436 1034380    0    0     0 14464  632  644  1  1 15 83
 0  2   2272 145624  47440 1031804    0    0     4 32664  279  364  0  1 23 76
 1  2   2272 135128  47440 1040700   32    0    32 16488 1382 2372  6  3  6 86
 0  1   2268 136340  47440 1040656    0    0     0     0  393  539  2  1 60 37
```

# Avaluació de rendiment

- iostat

```
$ iostat -m -d /dev/sda9 1
Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9              1.00        0.30         0.00          0          0

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9             65.00        0.00        31.00          0         31

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9             63.00        0.00        31.00          0         31

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9             52.00        0.00        24.13          0         24

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9             64.00        0.00        26.61          0         26

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
sda9             54.00        0.01        24.50          0         24

Device:            tps   MB_read/s    MB_wrtn/s    MB_read    MB_wrtn
```

# Avaluació de rendiment

- Eines de la interfície de sistema
  - gettimeofday

```
struct timeval {
        time_t      tv_sec;     /* seconds */
        suseconds_t tv_usec;    /* microseconds */
};

int gettimeofday(struct timeval *tv, struct timezone *tz);
```

    - Retorna el número de segons i microsegons que han passat des de l'Epoch (1970-01-01 00:00:00 +0000 (UTC))

# Avaluació de rendiment

- Exemple d'ús de gettimeofday

```
struct timeval tv0, tv1;
double secs;

res = gettimeofday(&tv0, NULL);
if (res < 0) {
    perror ("gettimeofday");
}

// codi del qual volem mesurar el temps

res = gettimeofday(&tv1, NULL);
if (res < 0) {
    perror ("gettimeofday");
}

secs = (((double)tv1.tv_sec*1000000.0 + (double)tv1.tv_usec) -
       ((double)tv0.tv_sec*1000000.0 + (double)tv0.tv_usec))/1000000.0;

printf ("temps %lf segons\n", secs);
```

# Avaluació de rendiment

- clock_gettime(clockid_t clock_id, struct timespec *tp);
- clock_gettime (2) Permet al procés que fa la crida recuperar el valor d'un rellotge, identificat per `clock_id`
- `clock_id` pot ser qualsevol dels 8 valors predefinits.
  - Si val `CLOCK_REALTIME` el resultat és el de gettimeofday(2)

| Linux Programmer's Manual | BSD Library Functions Manual |
|---|---|
| CLOCK_REALTIME | CLOCK_REALTIME |
| CLOCK_REALTIME_COARSE | CLOCK_MONOTONIC |
| CLOCK_MONOTONIC | CLOCK_MONOTONIC_RAW |
| CLOCK_MONOTONIC_COARSE | CLOCK_MONOTONIC_RAW_APPROX |
| CLOCK_MONOTONIC_RAW | CLOCK_UPTIME_RAW |
| CLOCK_BOOTTIME | CLOCK_UPTIME_RAW_APPROX |
| CLOCK_PROCESS_CPUTIME_ID | CLOCK_PROCESS_CPUTIME_ID |
| CLOCK_THREAD_CPUTIME_ID | CLOCK_THREAD_CPUTIME_ID |

# Avaluació de rendiment

- **Quin rellotge fer servir?**
- CLOCK_REALTIME quan volen saber l'hora del dia del sistema (en segons des de l'epoch)
- CLOCK_MONOTONIC quan volen saber temps transcorregut. S'incrementa linealment, però es veu afectat pel daemon de NTP.
- CLOCK_MONOTONIC_RAW no afectat per NTP. Kernels > 2.6.28. Depèn de hardware.
- CLOCK_PROCESS_CPU_TIME_ID només mesura el temps de CPU consumit pel procés.
- CLOCK_THREAD_CPUTIME_ID només mesura el temps de CPU gastat en el fil que realitza la sol·licitud.

# Avaluació de rendiment

CLOCK_GETTIME(2) vs GETTIMEOFDAY(2)

```
struct timespec {
          time_t   tv_sec;       /* seconds */
          long     tv_nsec;      /* nanoseconds */
        };
res = clock_gettime(CLOCK_REALTIME, &ts0);
secs = ((double)ts0.tv_sec*1000000000.0 +
(double)ts0.tv_nsec))/1000000000.0;
```

```
struct timeval {
          time_t      tv_sec;     /* seconds */
          suseconds_t tv_usec;    /* microseconds
*/
        };
res = gettimeofday(&tv0, NULL);
secs =((double)tv0.tv_sec*1000000.0 +
(double)tv0.tv_usec))/1000000.0;
```

Exemples: http://docencia.ac.upc.edu/FIB/grau/CASO/labs2018-19-2q/exemple-rendiment.tar.xz

# Índex

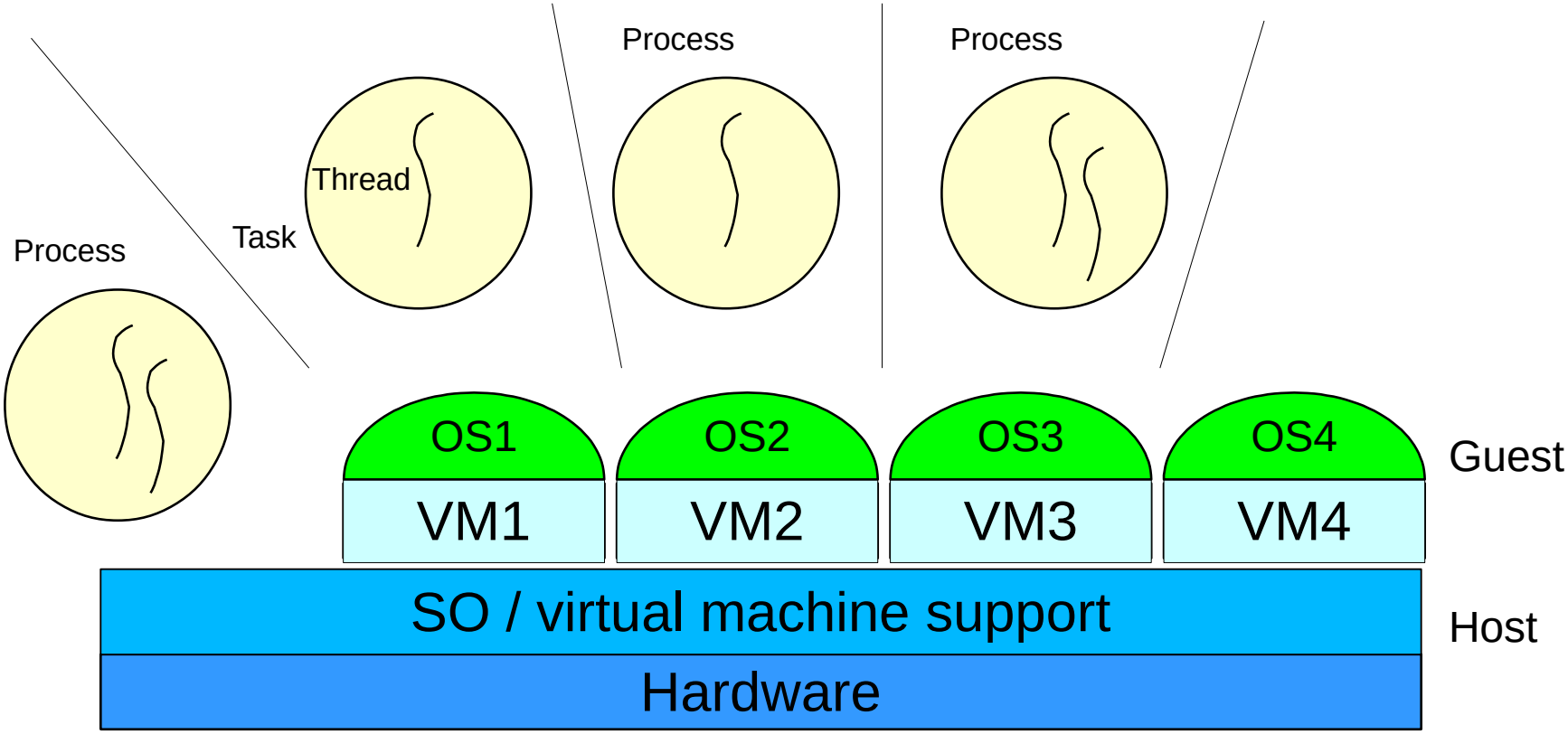
- Sincronització
  - Mutex / condition variables / futex / GCD
- Avaluació de rendiment
- Virtualització

# Virtualització

- Permet oferir un entorn virtual sencer a l'SO i les aplicacions
  - Diferent [o no] de la màquina física
- Podem oferir diverses màquines corrent sobre la mateixa
  - Diferents serveis
  - Diferents usuaris
  - ...

# Virtualització

- Estructura
  - Una màquina virtual és un procés en el sistema host



# Virtualització

- Exemple: El Qemu del laboratori amb Hurd
  - El procés qemu-system-i386 conté tot Debian

```
top - 19:08:22 up 2 days, 12:07, 14 users,  load average: 0.62, 0.71, 0.60
Tasks: 195 total,   1 running, 194 sleeping,   0 stopped,   0 zombie
Cpu(s):  9.9%us,  3.3%sy,  0.0%ni, 86.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   1937056k total,  1463172k used,   473884k free,   112964k buffers
Swap:  5119996k total,   578552k used,  4541444k free,   577756k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
30542 xavim     20   0 1851m 172m  54m S   13  9.1   3:27.69 soffice.bin
15979 xavim     20   0 1895m  78m 2044 S    6  4.2  93:28.81 qemu-system-i386
 2068 root      20   0  186m  43m  20m S    4  2.3  49:20.04 X
 2475 xavim     20   0  399m  36m  18m S    0  1.9  14:50.55 konsole
 5149 root      20   0     0    0    0 S    0  0.0   0:00.05 kworker/0:0
 5156 root      20   0     0    0    0 S    0  0.0   0:00.03 kworker/1:2
 5162 xavim     20   0 19532 1340  944 R    0  0.1   0:00.04 top
    1 root      20   0  4304    0    0 S    0  0.0   0:01.82 init
```

# Virtualització

- Exemple: El VirtualBox del laboratori amb Hurd
  - El procés VirtualBox conté tot Debian

```
top - 11:46:39 up  5:03,  6 users,  load average: 0.87, 0.45, 0.24
Tasks: 197 total,   1 running, 196 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.7%us,  0.7%sy,  0.0%ni, 98.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.7%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  :  0.0%us,100.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.7%us,  0.7%sy,  0.0%ni, 98.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   3929444k total,  2332412k used,  1597032k free,    83896k buffers
Swap:        0k total,        0k used,        0k free,  1084788k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
11512 xavim     20   0 1760m 240m 194m S  101  6.3   2:04.73 VirtualBox
  745 messageb  20   0 20036 1540  860 S    0  0.0   0:01.91 dbus-daemon
  859 root      20   0  381m 111m  89m S    0  2.9   0:48.23 X
 1174 xavim     20   0  547m  49m  24m S    0  1.3   0:52.05 konsole
11474 xavim     20   0  549m  10m 7236 S    0  0.3   0:00.56 VBoxSVC
    1 root      20   0  4308  692  592 S    0  0.0   0:00.52 init
    2 root      20   0     0    0    0 S    0  0.0   0:00.00 kthreadd
```

# Virtualització

- Exemple: El VirtualBox del laboratori amb Hurd
  - Podem veure si VirtualBox té “threads” (clones): H

```
top - 11:54:47 up  5:11,  6 users,  load average: 1.03, 0.93, 0.57
Tasks: 483 total,   2 running, 481 sleeping,   0 stopped,   0 zombie
Cpu0  :  1.3%us,  0.3%sy,  0.0%ni, 98.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  1.3%us,  0.7%sy,  0.0%ni, 98.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  :  0.0%us,100.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.7%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   3929444k total,  2330820k used,  1598624k free,    84132k buffers
Swap:        0k total,        0k used,        0k free,  1081144k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
11552 xavim     20   0 1760m 240m 194m R  100  6.3  10:09.59 EMT
 1174 xavim     20   0  547m  49m  24m S    1  1.3   0:53.04 konsole
11512 xavim     20   0 1760m 240m 194m S    1  6.3   0:06.69 VirtualBox
  859 root      20   0  374m 106m  83m S    0  2.8   0:49.62 X
 1041 xavim     20   0 2808m  83m  40m S    0  2.2   0:51.85 kwin
11466 xavim     20   0  810m  43m  29m S    0  1.1   0:01.62 VirtualBox
11471 xavim     20   0  810m  43m  29m S    0  1.1   0:00.32 nspr-1
11525 xavim     20   0 1760m 240m 194m S    0  6.3   0:01.00 nspr-2
11559 xavim     20   0 1760m 240m 194m S    0  6.3   0:00.78 Timer
```

# Virtualització

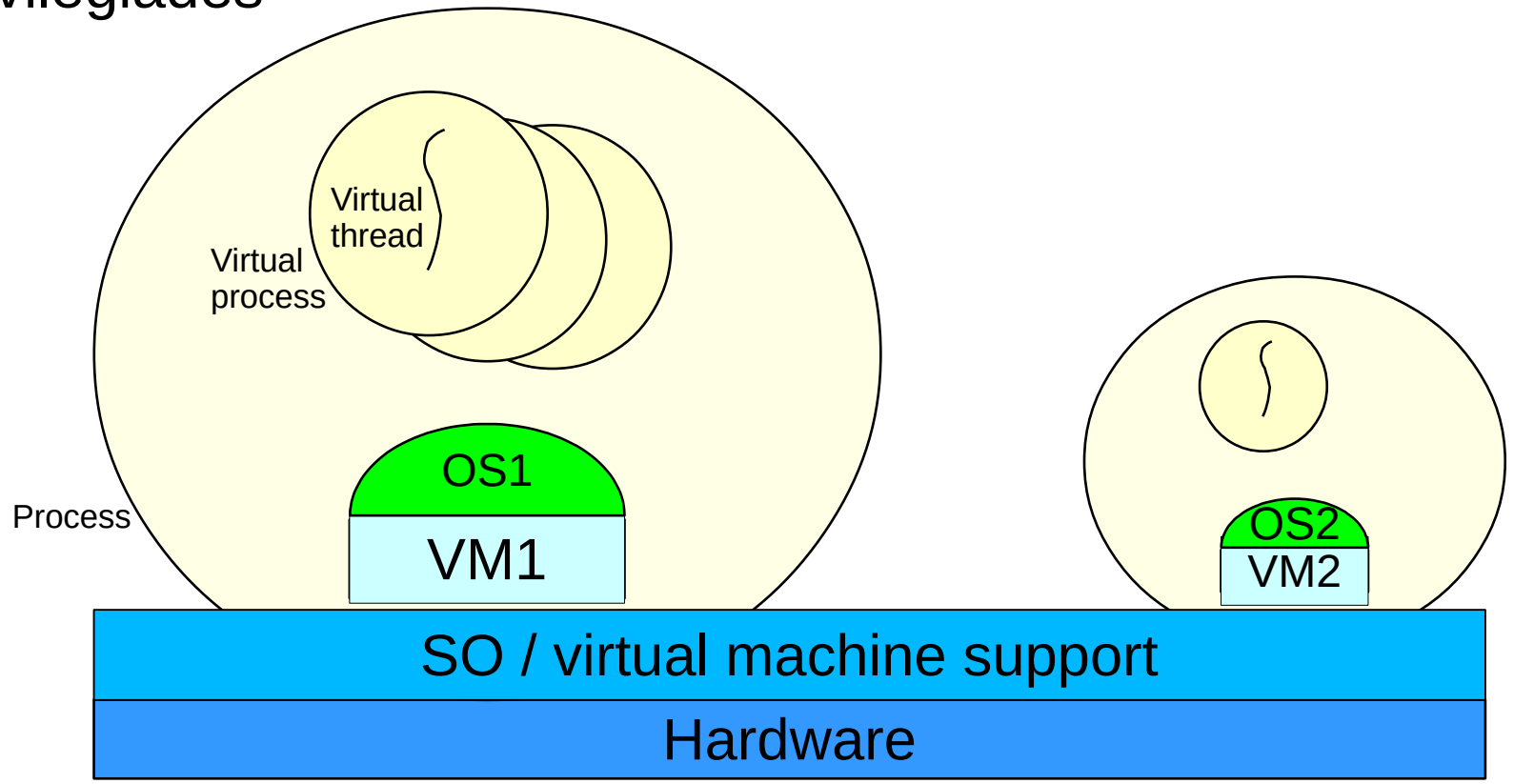
- Protecció
  - Cada màquina virtual està completament aïllada
    - de les altres
    - del host
- Compartició de recursos
  - Diverses màquines poden compartir recursos de forma segura
    - Processadors
    - Memòria
    - Disc
    - Xarxa
- Facilitat per engegar/parar i fer proves
  - Comprovació de nous serveis / del desenvolupament de l'empresa
  - Interessant per fer recerca en sistemes operatius

# Virtualització

- "System consolidation"
  - Ajuntar els serveis oferts per diverses màquines físiques en una de sola (física)
    - Usant una màquina virtual per a cadascuna de les originals
  - Els sistemes i serveis ja estan provats i se sap que funcionen bé
    - Es "consoliden" i s'estalvien recursos

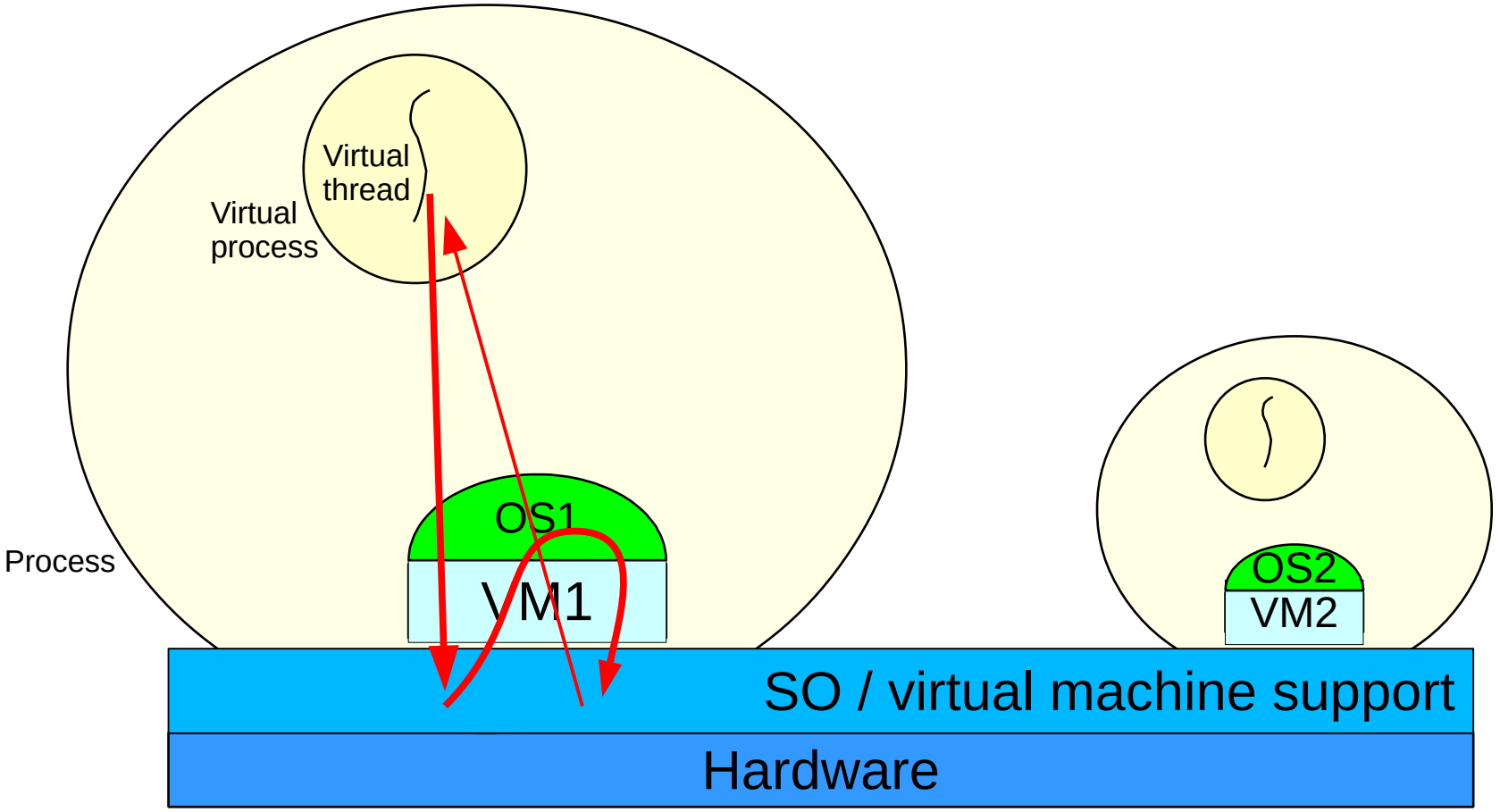
# Virtualització

- Dificultats d'implementació (veure OS1 a la fig.)
  - Necessitat d'executar el sistema operatiu OS1
    - en mode "sistema", "deixant-li" executar instruccions privilegiades



# Virtualització

- Virtual user mode + virtual kernel mode
  - Executant-se en mode usuari "físic"!



# Virtualització

- Suport del processador (*virtualization technology*)
  - Intel VT-x (VMX – virtual machine extensions)
    - virtual machine monitor (VMM)
      - root operation
      - estructures de dades per representar el guest
    - ID de processador virtual
    - taula de pàgines extesa
    - reducció en el cost de les transicions entre VMM i el guest
  - AMD-V
    - Processor Guest Mode
    - Control Data Structure (VMCB)
    - ...

# Virtualització

- Instruccions detectades en *non-root mode*
  - Halt, In/Out, iret
  - Load/store de les taules de descriptors, interrupcions, task
    - LGDT, LIDT, LLDT, LTR, SGDT, SIDT, SLDT, STR
  - Moure dades a registres de control (CR0, CR3, …) i MSRs
- Altres causes de transferència de control de guest a host
  - Exceptions, interrupcions, crides a sistema

# Exemples d'entorns virtuals

- VMware
- Hyper-V (Microsoft)
- VirtualBox (Oracle)
- Qemu (GPL/LGPL)
- Bochs (LGPL)
- KVM (GPL v2)
- Xen (GPL)
- Linux Containers (GPL v2)
- ...

http://en.wikipedia.org/wiki/Comparison_of_platform_virtual_machines

# Magatzems de sistemes virtuals

- Tipus
  - vmk
  - qcow2
  - raw
- Eines
  - qemu-img

# Magatzems de sistemes virtuals

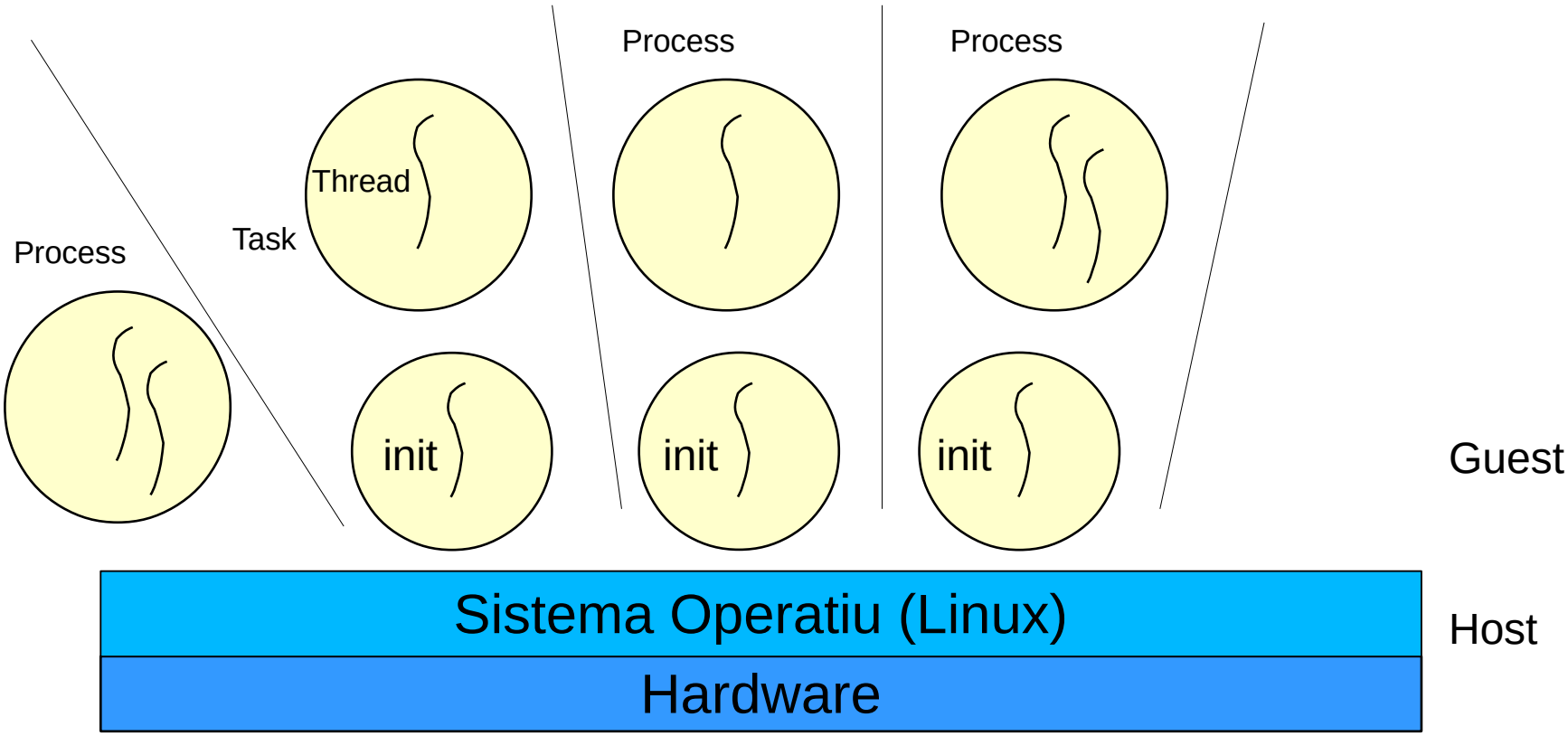
- Mount -o loop, offset=N …
  - Permet muntar una partició del disc en fitxer

Estructura del disc? Depèn de cada cas

```
$ /sbin/fdisk -l Fedora-Minimal-armhfp-21-5-sda.raw

Disk Fedora-Minimal-armhfp-21-5-sda.raw: 2139 MB, 2139095040 bytes
255 heads, 63 sectors/track, 260 cylinders, total 4177920 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xa99d2bd5

                                  Device  Boot      Start         End      Blocks   Id  System
Fedora-Minimal-armhfp-21-5-sda.raw1            2048     1001471      499712   83  Linux
Fedora-Minimal-armhfp-21-5-sda.raw2         1001472     1251327      124928   83  Linux
Fedora-Minimal-armhfp-21-5-sda.raw3         1251328     3985407     1367040   83  Linux
```

# Linux Containers

- Basat en compartir un sol kernel entre diferents jerarquies de processos



# Linux Containers

- Extensions per crear jerarquies de processos
  - Clone!!
    - CLONE_NEWUTS, nou espai de noms
      - Uname, domainname, hostname
    - CLONE_NEWIPC, shmem, sem, msg
    - CLONE_NEWNET, IPv4 & IPv6
    - CLONE_NEWNS, mount namespace
    - CLONE_NEWPID, pid = 1 i nova jerarquia

# Exercici

- **Per entregar com a pràctica al Racó**
  - Fer un programa que escrigui al disc 500 MBytes, mesurant el temps que triga a fer-ho amb gettimeofday
  - I que imprimeixi els temps invertit i el bandwidth que ha aconseguit la transferència d'informació