Sistemas Operativos

Administración de Procesos

Diego Madariaga

1.
Estrategias de Scheduling

Algunos conceptos

La ejecución de un proceso consiste en una sucesión alternada de:

- Ráfagas de CPU (tiempo que necesita para ejecutar instrucciones)
- Periodos de espera (proceso espera por la E/S)

La ejecución de un proceso puede ser determinada por las duraciones de sus ráfagas de CPU y sus esperas (en paréntesis):

- Ej: 3 (5) 2 (1) 3 (4)
- Dichas duraciones son independientes de la estrategia de scheduling utilizada

Estrategias de scheduling

- First come first served
- Shortest Job First
- Prioridades
- Round Robin

Pueden ser de tipo preemptive o non-preemptive.

Diagrama de scheduling

El diagrama muestra las decisiones de scheduling para 3 procesos. A la izquierda se indica para cada proceso las duraciones de sus ráfagas de CPU y entre paréntesis las duraciones de sus estados de espera.

La línea punteada indica que el estado del proceso es READY. Si el proceso está en estado de espera el espacio aparece en blanco

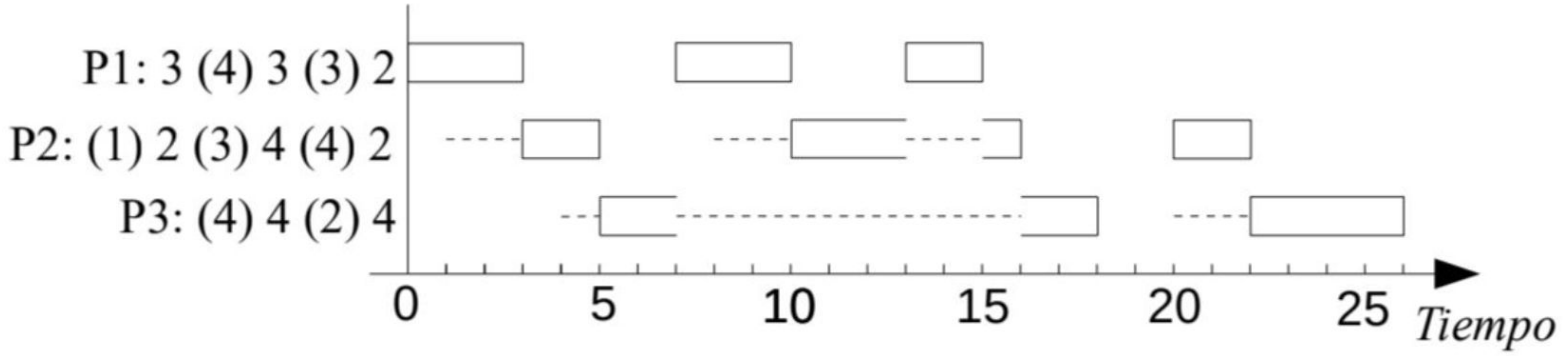
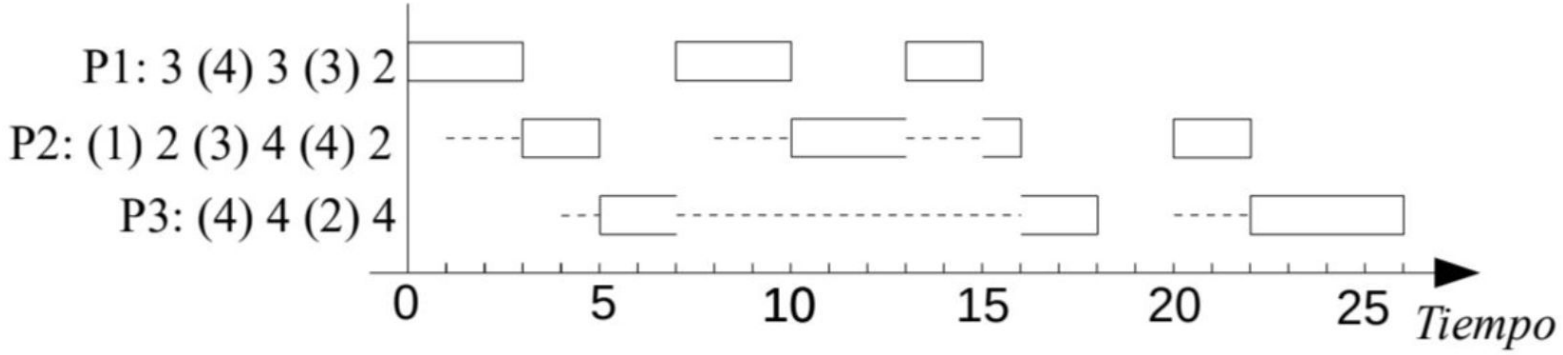


Diagrama de scheduling

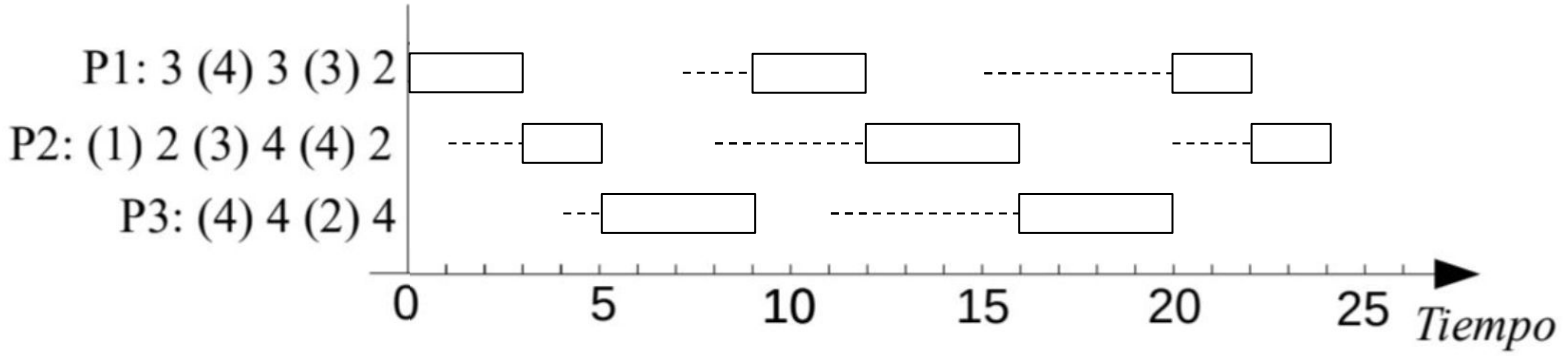
¿De qué estrategia de scheduling se trata?



- Preemptive
- Estrategia de prioridades

Diagrama de scheduling

Rehacer el diagrama considerando la estrategia *First Come First Served*.



2.
Scheduling con Prioridades para 1 core

Scheduler con prioridades para single core: nKernel/sched-pri-1core.c

- La cola ready

```
static NthQueue *nth_pri1ReadyQueue[MAXPRI];
```

- Setear prioridad a un thread

```
void nSetPriority(nThread th, int pri) {
  START_CRITICAL
  th->pri= pri;
  schedule(); // The calling thread may loose the CPU
  END_CRITICAL
}
```

Scheduler con prioridades para single core: nKernel/sched-pri-1core.c

- Pasar un thread a estado READY

```
static void nth_pri1SetReady(nThread th) {
  CHECK_CRITICAL("nth_fcfsSetReady")
  if (th->status==READY || th->status==RUN)
    nFatalError("nth_fcfsReady", "Already in READY status\n");
  th->status= READY;
  nth_putBack(nth_pri1ReadyQueue[th->pri], th);
}
```

Scheduler con prioridades para single core: nKernel/sched-pri-1core.c

- Pasar un thread a estado de espera

```
static void nth_pri1Suspend(State waitState) {
  CHECK_CRITICAL("nth_fcfsSuspend")
  nThread th= nSelf();
  if (th->status!=RUN && th->status!=READY)
    nFatalError("nth_fcfsSuspend", "Thread not ready/run\n");
  th->status= waitState;
}
```

Scheduler con prioridades para single core: nKernel/sched-pri-1core.c

- Scheduler como tal:

```
static void nth_pri1Schedule(void) {
  ...
}
```

Función que se llama cada vez que hay cambios en la cola ready (ej: se agrega un nuevo thread). Acá se determina cuál es el siguiente thread a ejecutarse.

Sistemas Operativos

Administración de Procesos

Diego Madariaga