Лекция 21
Планирование процессов

Планирование процессов

- Планировщик — компонента ядра операционной системы
- Планировщик определяет, какой процесс из числа готовых к выполнению назначается на выполнение на ЦП
- Типы планировщиков:
  - Пакетный
  - Разделения времени
  - Реального времени

Цели планирования

- Максимизация пропускной способности (throughput)
- Минимизация времени отклика (response time)
- Минимизация задержки (latency)
- Максимизация честности (fairness)
- Соблюдение сроков (deadline)
- Равномерная загрузка всех процессоров
-
- **Цели противоречивы — требуется компромисс!**

Пакетное планирование (batch)

- Цель — обеспечить максимальную пропускную способность ВС (то есть максимальное число выполненных задач)
- Ядро переключается с одного на другой процесс при следующих условиях:
  - Выполнявшийся процесс завершил работу
  - При выполнении возникла фатальная ошибка или процесс исчерпал отведенные ему ресурсы
  - Выполнявший процесс инициировал операцию, которая не может быть выполнена немедленно
  - Процесс запросил добровольное переключение

Планирование разделения времени (time sharing)

- Цель: разделить процессорное время между процессами, готовыми к выполнению
- Ядро переключается с одного процесса на другой при следующих условиях
  - Процесс завершил работу
  - При выполнении возникла ошибка
  - Процесс инициировал операцию, которая не может быть выполнена немедленно
  - Истек квант времени выполнения процесса
  - Процесс запросил добровольное переключение

Классификация процессов

- По поведению
  - «I/O-bound» - процесс выполняет активный обмен с внешними устройствами и проводит много времени в ожидании ввода-вывода (пример: веб-сервер, редактор текста)
  - «CPU-bound» - процессы, интенсивно занимающие процессорное время (пример: компиляция программ, вычислительные задачи, рендеринг изображений и т. п.)

Классификация процессов

- По назначению:
  - Интерактивные — основное время проводят в ожидании пользовательского ввода, при поступлении ввода должны быстро активироваться, чтобы не было ощущения «торможения»
  - Пакетные — не ожидают ввода пользователя (компиляторы, численные приложения...)

Параметры планирования процессов разделения времени

- Значение nice: [-20, 19] — чем меньше значение, тем выше приоритет. 0 — приоритет по умолчанию
- Приоритет группы процессов: grpnice
- Приоритет пользователя: usrnice
- Полный приоритет: nice + grpnice + usrnice отсеченное по интервалу [-20; 19]
-

Планирование в Linux

- Linux поддерживает различные алгоритмы планирования, которые можно выбирать «на лету»
- Основной планировщик для процессов разделения времени: CFS — completely fair scheduler

CFS

- Идеальный процессор — время процессора делится поровну между всеми выполняющимися задачами
- Доля каждого процесса изменяется динамически с добавлением или удалением процессов из очереди процессов, готовых к выполнению
- Target Scheduling Latency — время, за которое выполняются все готовые к выполнению процессы
  - Если TSL = 20ms и есть два готовых процесса, каждый процесс будет работать по 10ms
- Minimum Granularity — минимальное время выполнения одного процесса

Virtual Runtime

- У каждого процесса ведется учет virtual runtime (в рамках TSL)
- Как только virtual runtime процесса становится больше virtual runtime какого-либо другого процесса, текущий процесс снимается с выполнения, и на выполнение ставится процесс с минимальным virtual runtime
- У высокоприоритетных процессов virtual runtime растет медленее, чем у низкоприоритетных

Virtual runtime

delta_exec = now – exec_start;

- Для процессов с nice == 0 (приоритет по умолчанию) virtual runtime == phys runtime
  - delta_exec_weighted = delta_exec
- Для произвольных процессов:
  - delta_exec_weighed = delta_exec * (NICE_0_LOAD / weight);
  - NICE_0_LOAD = 2^10 (1024)
  - Weight = NICE_0_LOAD / (1.25 ^ nice); // ^ - степень
  - Вычисления целочисленные, берутся табличные значения
- Vruntime += delta_exec_weighted

Вызов планировщика

- Регулярно при прерываниях от таймера
- При переводе процесса в режим ожидания (если результат системного вызова не доступен немедленно)
- При переводе процесса из режима ожидания в режим готовности к выполнению

Планирование реального времени (real time)

- Цель: обеспечить минимальное время отклика, то есть время от наступления события до постановки на выполнение процесса, ожидающего этого события
- Виды планирования реального времени
  - На основе фиксированного расписания
  - На основе статических приоритетов

Планирование реального времени

- Ядро переключается с одного процесса на другой при следующих условиях
  - Процесс завершил работу
  - При выполнении возникла ошибка
  - Процесс инициировал операцию, которая не может быть выполнена немедленно
  - Готов к выполнению процесс с большим приоритетом
  - Истек квант времени выполнения процесса
  - Процесс запросил добровольное переключение

Статический приоритет

- Каждый процесс реального времени имеет статический приоритет [1;99]
- Процессы разделения времени имеют статический приоритет 0, то есть назначаются на выполнение, только если нет готовых к выполнению процессов реального времени

Типы планирования р.в.

- SCHED_FIFO — нет квантования времени, процесс выполняется, пока не появится более высокоприоритетный процесс, либо процесс не начнет ввод-вывод, либо не будет снят
- SCHED_RR (round-robin) — выполнение квантуется, процессы одного приоритета выполняются по очереди

Инверсия приоритета (priority inversion)

- Предположим, что низкоприоритетный процесс P1 захватил некоторый ресурс R
- В это время стал готов к выполнению высокоприоритетный процесс P2, которому требуется ресурс R. Процесс P2 ожидает освобождения ресурса R процессом P1
- В это время может быть назначен на выполнение среднеприоритетный процесс P3, который еще более отсрочит время освобождения ресурса R процессом P1

Инверсия приоритета

- Проблема возникает, потому что процессы, ожидающие освобождения ресурса неявно получают приоритет процесса, захватившего ресурс.
- Отсрочка выполнения высокоприоритетного процесса может иметь катастрофические последствия
- Однозначного решения проблемы не существует
- Возможный вариант: назначать процессу, захватившему ресурс, максимальный приоритет ожидающего процесса (наследование приоритета)