Операционные системы

Евгений Власов

Алгоритмы планирования процессов

Решение о том, кому дать следующий квант времени процессора определяет планирование.

Планирование процессов в ОС это процесс выбора – кто будет исполняться следующим и как долго это будет исполняться.

ВАЖНО! Не путать с диспетчеризацией (переключением контекста), которая является просто механизмом передачи управления.

Классы планировщиков

- Пакетный
- Итерактивный
- Реального времени

Пакетный – ориентирован на длительные задачи, которые требуют больших вычислительных ресурсов, где не требуется частое прерывание. Т.е. подразумевают обработку больших задач большими пакетами, нет ограничения на время выполнения.

Интерактивный – ориентирован на снижение времени отклика, т.е. чтобы система казалась ”отзывчивой”. Обычные абонентские системы на ПК – это интерактивные системы, когда в ответ на действие пользователя (например перемещение мыши) ОС что-то делает. И всегда пользователю хочется, чтобы этот ответ происходил как можно быстрее.
Главное чтобы на поступающий в систему запрос был получен максимально быстро ответ. Запрос – это любое взаимодействие с компьютером.

Реального времени – специализированные класс, ориентированный на **дедлайн** – предельный срок завершения какой-либо работы. Главное, чтобы определенное действие завершалось к определенному сроку, это понятие называется **дедлайн**. Поступающий запрос должен быть обработан не более, чем в определенный промежуток времени. Классический пример СРВ – управление ядерным реактором, в котором превышение времени отклика приведет к аварийной ситуации.

Уровни планирования

- **Долговременное(долгосрочное)** – решает какие новые задачи будут добавлены (концептуальные вопросы).
- **Среднесрочное** – решает нужно ли временно выгружать программу во вторичную память (какую и вообще нужно ли это).
- **Краткосрочный** – решает, какому потоку дать следующий квант процессорного времени и какой длины. Координирует выполняющиеся потоки на разных ЦП.

Цели планирования

- Справедливость
- Эффективность
- Сокращение полного времени выполнения (turnaround time)
- Сокращение времени ожидания (waiting time)
- Сокращение времени отклика (response time)

Справедливость

гарантировать каждому заданию или процессу определенную часть времени использования процессора в компьютерной системе, при этом не допуская возникновения ситуации, когда процесс одного пользователя постоянно занимает процессор, в то время как процесс другого пользователя фактически не начинал выполняться.

Эффективность

постараться занять процессор на все 100% рабочего времени, не позволяя ему простаивать в ожидании процессов, готовых к исполнению. В реальных вычислительных системах загрузка процессора колеблется от 40 до 90%.

Сокращение полного времени выполнения (turnaround time)

обеспечить минимальное время между стартом процесса или постановкой задания в очередь для загрузки и его завершением.

Сокращение времени ожидания (***waiting time***)

сократить время, которое проводят процессы в состоянии готовность и задания в очереди для загрузки.

Сокращение времени отклика (***response time***)

минимизировать время, которое требуется процессу в интерактивных системах для ответа на запрос пользователя.

Желаемые свойства алгоритмов планирования

- Предсказуемость
- Минимизация накладных расходов.
- Равномерность загрузки вычислительной системы.
- Масштабируемость.

Предсказуемость

Одно и то же задание должно выполняться приблизительно за одно и то же время. Применение алгоритма *планирования* не должно приводить, к примеру, к извлечению квадратного корня из 4 за сотые доли секунды при одном запуске и за несколько суток – при втором запуске.

Минимизация накладных расходов.

Если на каждые 100 миллисекунд, выделенные процессу для использования процессора, будет приходиться 200 миллисекунд на определение того, какой именно процесс получит процессор в свое распоряжение, и на переключение контекста, то такой алгоритм, очевидно, применять не стоит.

Равномерность загрузки вычислительной системы

ОС обеспечивает доступ процессам ко всем ресурсам ВС, избегая по возможности ситуации когда нужные процессу ресурсы недоступны из-за некорректного планирования.

Масштабируемость

способность системы, сети или процесса справляться с увеличением рабочей нагрузки (увеличивать свою производительность) при добавлении ресурсов и/или увеличении нагрузки.

Например, рост количества процессов в системе в два раза не должен приводить к увеличению полного времени выполнения процессов на порядок

Статические параметры планирования

Статические параметры вычислительной системы – например, предельные значения ее ресурсов.

Статические параметры процесса – кем запущен, степень важности, запрошенное процессорное время, какие требуются ресурсы и т.д.

Динамические параметры планирования

Динамические параметры вычислительной системы – например, количество свободных ресурсов в данный момент.

Динамические параметры процесса – текущий приоритет, размер занимаемой оперативной памяти, использованное процессорное время и т.д.

Алгоритмы планирования

- **First-Come, First-Served (FCFS)**
- **Round Robin (RR)**
- **Shortest-Job-First (SJF)**
- **Гарантированное планирование**
- **Приоритетное планирование**
- **Многоуровневые очереди (Multilevel Queue)**
- **Многоуровневые очереди с обратной связью (Multilevel Feedback Queue)**

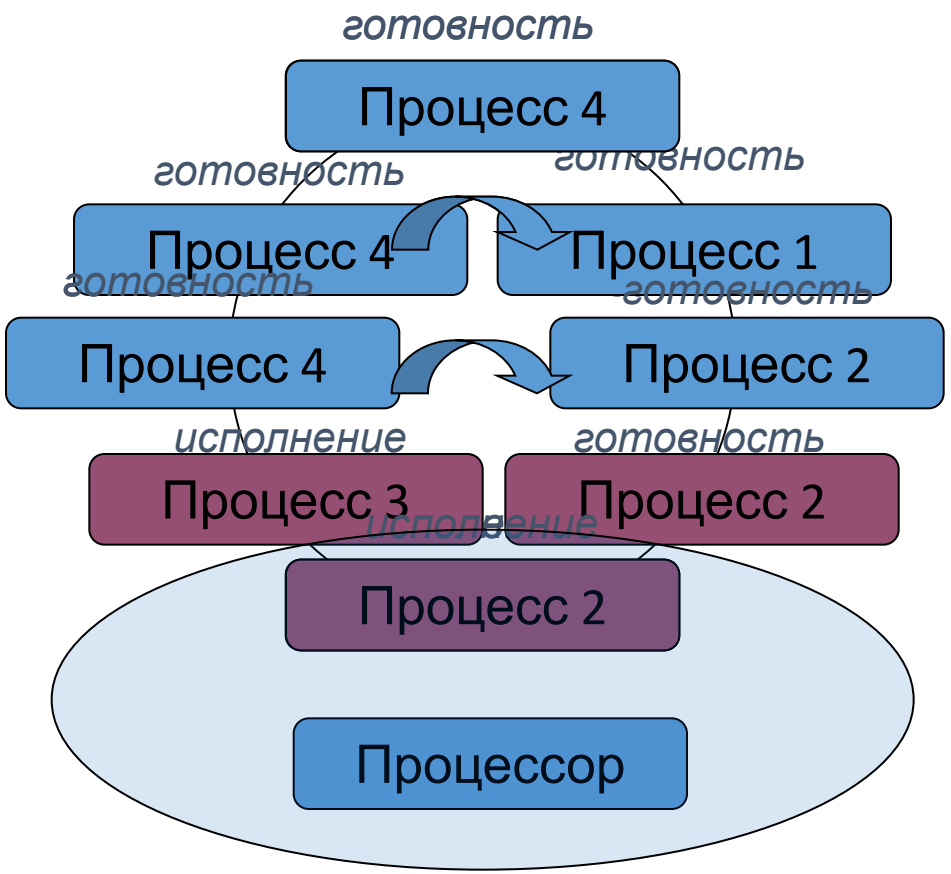
Алгоритмы планирования

FCFS (First Come – First Served)

Процессы	P_0	P_1	P_0
Продолжительность CPU burst	13	4	13

Алгоритмы планирования

RR (Round Robin)



Алгоритмы планирования

RR (Round Robin)

- Остаток времени CPU burst <= кванта времени:
  - процесс освобождает процессор до истечения кванта;
  - на исполнение выбираем новый процесс из начала очереди готовых;
- Остаток времени CPU burst >= кванта времени:
  - По окончании кванта процесс помещается в конец очереди готовых к исполнению процессов;
  - на исполнение выбираем новый процесс из начала очереди готовых.

Алгоритмы планирования
RR (Round Robin)

Процессы	P_0	P_1	P_2
Продолжительность CPU burst	13	4	1

Величина кванта времени – 4

время	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_0	И	И	И	И	Г	Г	Г	Г	Г	И	И	И	И	И	И	И	И	И
P_1	Г	Г	Г	Г	И	И	И	И										
P_2	Г	Г	Г	Г	Г	Г	Г	Г	И									

исполнение

P_0

Очередь готовых

P_0	P_0	P_0

Алгоритмы планирования
RR (Round Robin)

Процессы	P_0	P_1	P_2
Продолжительность CPU burst	13	4	1

Величина кванта времени – 1

время	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_0	И	Г	Г	И	Г	И	Г	И	Г	И	И	И	И	И	И	И	И	И
P_1	Г	И	Г	Г	И	Г	И	Г	И									
P_2	Г	Г	И															

исполнение

P_0

Очередь готовых

P_0	P_0	P_0

Алгоритмы планирования

SJF (Shortest Job First) невытесняющий

Процессы	P_0	P_1	P_2	P_3
Продолжительность CPU burst	5	3	7	1

время	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
P_0	Г	Г	Г	Г	И	И	И	И	И							
P_1	Г	И	И	И												
P_2	Г	Г	Г	Г	Г	Г	Г	Г	Г	И	И	И	И	И	И	И
P_3	И															

исполнение

P_0

готовность

P_0	P_1	P_2	P_3

Алгоритмы планирования

SJF (Shortest Job First) вытесняющий

Процессы	P_0	P_1	P_2	P_3
Продолжительность CPU burst	6	2	5	5
Момент появления в очереди	0	2	6	0

время	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_0	Г	Г	Г	Г	Г	Г	Г	Г	Г	Г	Г	Г	И	И	И	И	И	И
P_1			И	И														
P_2							Г	И	И	И	И	И						
P_3	И	И	Г	Г	И	И	И											

исполнение

P_0

готовность

P_0	P_1	P_2	P_3

Приоритетное планирование

каждому процессу присваивается определенное числовое значение – приоритет, в соответствии с которым ему выделяется процессор. Процессы с одинаковыми приоритетами планируются в порядке FCFS.

Планирование с использованием приоритетов может быть как вытесняющим, так и невытесняющим. При вытесняющем планировании процесс с более высоким приоритетом, появившийся в очереди готовых процессов, вытесняет исполняющийся процесс с более низким приоритетом. В случае невытесняющего планирования он просто становится в начало очереди готовых процессов. Давайте рассмотрим примеры использования различных режимов приоритетного

Многоуровневые очереди (Multilevel Queue)