27. ОС. Алгоритмы планирования. Планирование в системах пакетной обработки данных. Планирование в интерактивных системах. Память. Виртуальное адресное пространство. Политика замещения страниц.

Алгоритмы планирования в пакетных системах

1. Первый пришел – первый обслужен (неприоритетный, простейший алгоритм планирования).
2. Сначала самое короткое задание (неприоритетный, сроки выполнения заданий должны быть известны заранее).
3. Приоритет наименьшему времени выполнения (приоритетная версия предыдущего алгоритма, если новой задаче на выполнение требуется меньше времени, чем на завершение текущей, запускается новая задача).

Алгоритмы планирования в интерактивных системах

4. Циклическое планирование (процессу выделяется ресурс и квант времени, если по истечению выделенного времени процесс все еще выполняется, то выделенный ему ресурс отбирается и передается другому процессу).
5. Приоритетное планирование (каждому процессу присваивается уровень приоритетности, запускается процесс с наивысшим)
6. Лотерейное планирование (процессу выделяется квант времени, следующий процесс определяется путем лотереи, более важным процессам выделяется большее число «лотерейных билетов»).
7. Гарантированное планирование (каждому процессу выделяется доля процессорного времени *A,* при работе засекается сколько на самом деле ему досталось времени *B,* следующим выполняется процесс с наименьшим соотношением *B/A,* и будет работать пока его соотношение не превысит соотношение его ближайшего конкурента)
8. Справедливое планирование (всем пользователям выделяется равное количество процессорного времени).

Адресное пространство — это набор адресов, который может быть использован процессом для обращения к памяти. У каждого процесса имеется свое собственное адресное пространство, независимое от того адресного пространства, которое принадлежит другим процессам (за исключением тех особых обстоятельств, при которых процессам требуется совместное использование их адресных пространств).

Виртуальная память — метод управления памятью компьютера, позволяющий выполнять программы, требующие больше оперативной памяти, чем имеется в компьютере, путём автоматического перемещения частей программы между основной памятью и вторичным хранилищем (например, жёстким диском).

В основе виртуальной памяти лежит идея, что у каждой программы имеется свое собственное адресное пространство, которое разбивается на участки, называемые страницами. Каждая страница представляет собой непрерывный диапазон адресов. Эти страницы отображаются на физическую память, но для запуска программы присутствие в памяти всех страниц не обязательно. Когда программа ссылается на часть своего адресного пространства, находящегося в физической памяти, аппаратное обеспечение осуществляет необходимое отображение на лету. Когда программа ссылается на часть своего адресного пространства, которое не находится в физической памяти, операционная система предупреждается о том, что необходимо получить недостающую часть и повторно выполнить потерпевшую неудачу команду.

Алгоритмы замещения страниц

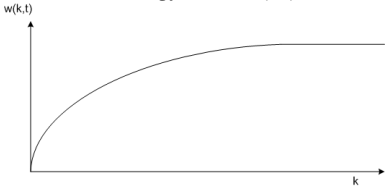
1. Идеальный (оптимальный) алгоритм заключается в том, чтобы выгружать ту страницу, которая будет запрошена позже всех.

Но этот алгоритм неосуществим, т.к. нельзя знать какую страницу когда запросят. Можно лишь набрать статистику использования.

2. Алгоритм NRU (Not Recently Used - не использовавшаяся в последнее время страница)
Используются биты обращения (R-Referenced) и изменения (M-Modified) в таблице страниц.
При обращении бит R выставляется в 1, через некоторое время ОС переведет его в 0.
M переводится в 0 только после записи на диск.
Благодаря этим битам можно получить 4 класса страниц:

Класс 0. не было обращений и изменений (R=0, M=0)
Класс 1. не было обращений, было изменение (R=0, M=1)
Класс 2. было обращение, не было изменений (R=1, M=0)
Класс 3. было обращение и изменение (R=1, M=1)

Алгоритм удаляет произвольную страницу, относящуюся к самому низкому непустому классу.
3. Алгоритм FIFO (First In, First Out «первой пришла – первой ушла»)
Редко используемый алгоритм, существенный недостаток которого заключается в том, что наиболее часто запрашиваемая страница может быть выгружена.
4. Алгоритм «второй шанс»
Модификация предыдущего алгоритма FIFO, исключающая выгрузку часто используемой страницы. Если R=1, то страница переводится в конец очереди, если R=0, то страница выгружается.
Недостаток заключается в частом перемещении страниц в очереди.
5. Алгоритм «часы»
Страницы содержатся в циклическом списке в виде часов, где стрелка указывает на самую старую страницу. Если R=0, страница выгружается. Если R=1, бит R сбрасывается, стрелка движется вперед.
6. Алгоритм LRU (Least Recently Used - использовавшаяся реже всего)
Первый метод:
Чтобы реализовать этот алгоритм, можно поддерживать список, в котором выстраивать страницы по количеству использования. Эта реализация очень дорога.
Второй метод:
В таблице страниц добавляется запись - счетчик обращений к странице. Чем меньше значение счетчика, тем реже она использовалась.
7. Алгоритм «рабочий набор»
Замещение страниц по запросу - когда страницы загружаются по требованию, а не заранее, т.е. процесс прерывается и ждет загрузки страницы.
Буксование - когда каждую следующую страницу приходится процессу загружать в память.
Чтобы не происходило частых прерываний, желательно чтобы часто запрашиваемые страницы загружались заранее, а остальные подгружались по необходимости.
Рабочий набор - множество страниц (к), которое процесс использовал до момента времени (t). Т.е. можно записать функцию w(k,t).

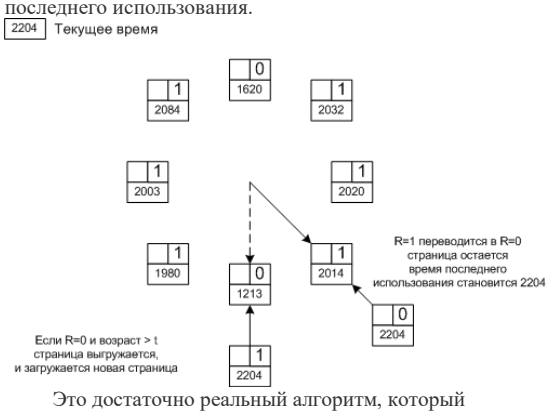


Т.е. рабочий набор выходит в насыщение, значение w(k,t) в режиме насыщения может служить для рабочего набора, который необходимо загружать до запуска процесса.
Алгоритм заключается в том, чтобы определить рабочий набор, найти и выгрузить страницу, которая не входит в рабочий набор.
Этот алгоритм можно реализовать, записывая, при каждом обращении к памяти, номер страницы в специальный сдвигающийся регистр, затем удалялись бы дублирующие страницы. Но это дорого.
В принципе можно использовать множество страниц, к которым обращался процесс за последние **t** секунд.
Текущее виртуальное время (Tv) - время работы процессора, которое реально использовал процесс.
Время последнего использования (Told) - текущее время при R=1, т.е. все страницы проверяются на

R=1, и если да то текущее время записывается в это поле.
Теперь можно вычислить возраст страницы (не обновления) **Tv-Told,** и сравнить с**t**, если больше, то страница не входит в рабочий набор, и страницу можно выгружать.
Получается три варианта:

- если R=1, то текущее время запоминается в поле время последнего использования
- если R=0 и возраст > t, то страница удаляется
- если R=0 и возраст =< t, то эта страница входит в рабочий набор

8. Алгоритм WSClock
Алгоритм основан на алгоритме "часы", но использует рабочий набор.
Используются битов R и M, а также время последнего использования.



Это достаточно реальный алгоритм, который используется на практике.