Вариант 1

Модель алгоритма планирования потоков, основанного на квантовании

1. Исходные данные:

- фиксированная единая очередь потоков с заданным временем выполнения,
- фиксированная величина кванта процессорного времени,
- количество процессоров - 1,
- кратчайшая задача - первая.

2. Результаты работы модели должны включать:

- среднее время выполнения коротких и длинных потоков.

Оглавление

Введение

Модель алгоритма планирования потоков, основанного на квантовании, является одной из самых распространенных моделей в операционных системах. Эта модель используется для определения порядка выполнения процессов в многозадачной среде.

Цель реферата более подробно рассмотреть алгоритмы планирования потоков, основанных на квантовании.

Квант времени - это интервал времени, выделенный процессу для выполнения своих задач. В модели планирования потоков, основанной на квантовании, каждый процесс получает определенное количество квантов времени для выполнения своих задач. После истечения кванта времени процесс переходит в состояние готовности, и управление передается следующему процессу.

Одним из преимуществ модели планирования потоков, основанной на квантовании, является то, что она позволяет обеспечить справедливое распределение ресурсов между процессами. Каждый процесс получает равное количество времени для выполнения своих задач, что позволяет избежать ситуации, когда один процесс занимает все ресурсы системы.

Однако, модель планирования потоков, основанная на квантовании, также имеет свои недостатки. Например, если квант времени слишком

маленький, то система может тратить слишком много времени на переключение контекста между процессами. С другой стороны, если квант времени слишком большой, то процессы могут быть заблокированы на длительное время, что может привести к задержкам в выполнении задач.

1 Алгоритмы планирования, основанные на квантовании

В основе многих вытесняющих алгоритмов планирования лежит концепция квантования. В соответствии с этой концепцией каждому потоку поочередно для выполнения предоставляется ограниченный непрерывный период процессорного времени — квант. Смена активного потока происходит, если:

- поток завершился и покинул систему;
- произошла ошибка;
- поток перешел в состояние ожидания;
- исчерпан квант процессорного времени, отведенный данному потоку.

Поток, который исчерпал свой квант, переводится в состояние готовности и ожидает, когда ему будет предоставлен новый квант процессорного времени, а на выполнение в соответствии с определенным правилом выбирается новый поток из очереди готовых. Граф состояний потока, изображенный на рис. 1, соответствует алгоритму планирования, основанному на квантовании.

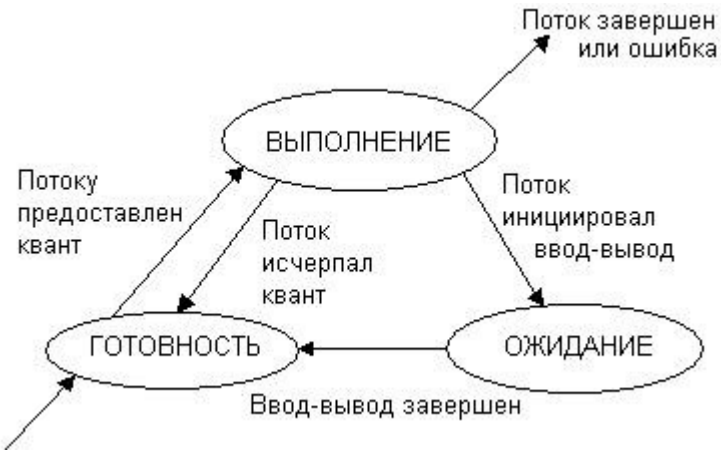

Рисунок 1 - Граф состояний потока в системе с квантованием

Кванты, выделяемые потокам, могут быть одинаковыми для всех потоков или различными. Рассмотрим, например, случай, когда всем потокам предоставляются кванты одинаковой длины q (рис. 2). Если в системе имеется п потоков, то время, которое поток проводит в ожидании следующего кванта, можно грубо оценить как q(n-l). Чем больше потоков в системе, тем больше время ожидания, тем меньше возможности вести одновременную интерактивную работу нескольким пользователям. Но если величина кванта выбрана очень небольшой, то значение произведения q(n-l) все равно будет достаточно мало для того, чтобы пользователь не ощущал дискомфорта от присутствия в системе других пользователей. Типичное значение кванта в системах разделения времени составляет десятки миллисекунд.

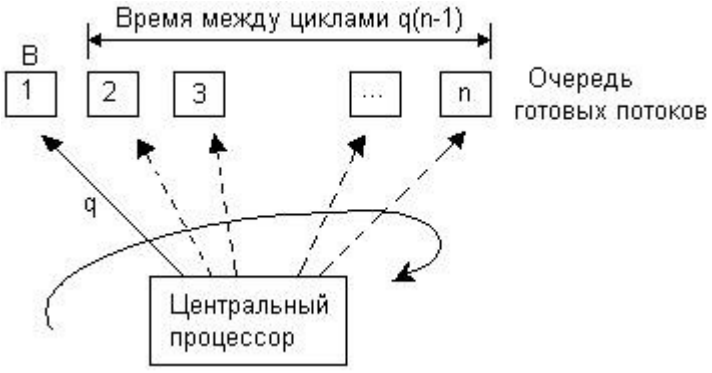

Рисунок 2 - Иллюстрация расчета времени ожидания в очереди

Если квант короткий, то суммарное время, которое проводит поток в ожидании процессора, прямо пропорционально времени, требуемому для его выполнения (то есть времени, которое потребовалось бы для выполнения этого потока при монопольном использовании вычислительной системы). Действительно, поскольку время ожидания между двумя циклами выполнения равно q(n-l), а количество циклов B/q, где B — требуемое время выполнения, то W*B(n-l). Заметим, что эти соотношения представляют собой весьма грубые оценки, основанные на предположении, что B значительно превышает q. При этом не учитывается, что потоки могут использовать кванты не полностью, что часть времени они могут тратить на ввод-вывод, что количество потоков в системе может динамически меняться и т. д.

Чем больше квант, тем выше вероятность того, что потоки завершатся в результате первого же цикла выполнения, и тем менее явной становится зависимость времени ожидания потоков от их времени выполнения. При достаточно большом кванте алгоритм квантования вырождается в алгоритм последовательной обработки, присущий однопрограммным системам, при котором время ожидания задачи в очереди вообще никак не зависит от ее длительности.

Кванты, выделяемые одному потоку, могут быть фиксированной величины, а могут и изменяться в разные периоды жизни потока. Пусть, например, первоначально каждому потоку назначается достаточно большой квант, а величина каждого следующего кванта уменьшается до некоторой заранее заданной величины. В таком случае преимущество получают короткие задачи, которые успевают выполняться в течение первого кванта, а длительные вычисления будут проводиться в фоновом режиме. Можно представить себе алгоритм планирования, в котором каждый следующий квант, выделяемый определенному потоку, больше предыдущего. Такой подход позволяет уменьшить накладные расходы на переключение задач в том случае, когда сразу несколько задач выполняют длительные вычисления.

Потоки получают для выполнения квант времени, но некоторые из них используют его не полностью, например из-за необходимости выполнить ввод или вывод данных. В результате возникает ситуация, когда потоки с интенсивными обращениями к вводу-выводу используют только небольшую часть выделенного им процессорного времени. Алгоритм планирования может исправить эту «несправедливость». В качестве компенсации за неиспользованные полностью кванты потоки получают привилегии при последующем обслуживании. Для этого планировщик создает две очереди готовых потоков (рис. 3). Очередь 1 образована потоками, которые пришли в состояние готовности в результате исчерпания кванта времени, а очередь 2 — потоками, у которых завершилась операция ввода-вывода. При выборе потока для выполнения прежде всего просматривается вторая очередь, и только если она пуста, квант выделяется потоку из первой очереди.

Многозадачные ОС теряют некоторое количество процессорного времени для выполнения вспомогательных работ во время переключения контекстов задач. При этом запоминаются и восстанавливаются регистры, флаги и указатели стека, а также проверяется статус задач для передачи управления. Затраты на эти вспомогательные действия не зависят от величины кванта времени, поэтому чем больше квант, тем меньше суммарные накладные расходы, связанные с переключением потоков.

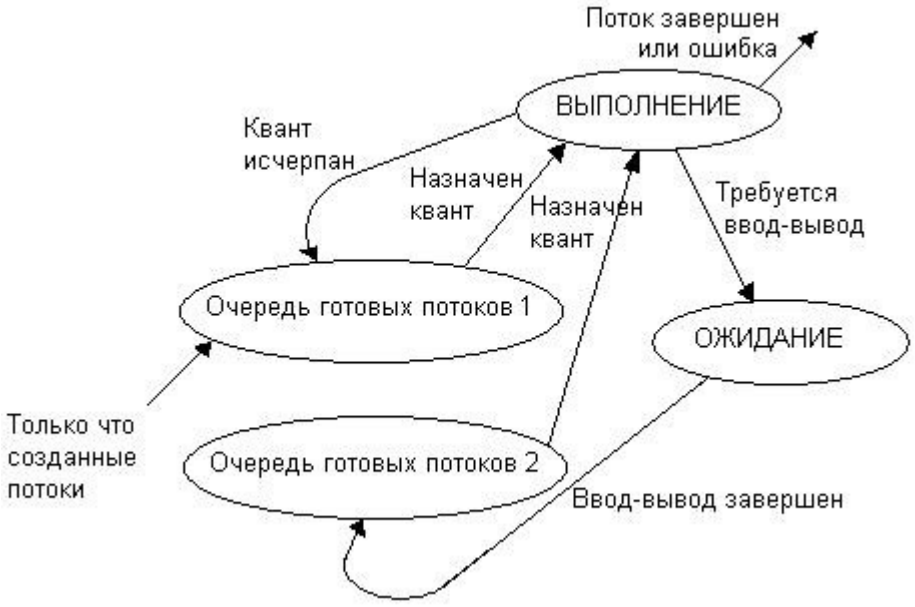

Рисунок 3 - Квантование с предпочтением потоков, интенсивно обращающихся к вводу-выводу

2 Практическая часть

Вариант 1

Модель алгоритма планирования потоков, основанного на квантовании

1. Исходные данные:

- фиксированная единая очередь потоков с заданным временем выполнения,
- фиксированная величина кванта процессорного времени,
- количество процессоров - 1,
- кратчайшая задача - первая.

2. Результаты работы модели должны включать:

- среднее время выполнения коротких и длинных потоков.
- количество потоков, которые были выполнены за определенный период времени.
- среднее время ожидания для каждого потока.
- процент использования процессорного времени.

Решение:

3. Алгоритм планирования:

o Каждый поток получает квант процессорного времени для выполнения.

o Если поток завершен, то он удаляется из очереди.

o Если поток не завершен, но истекло его время выполнения, то он перемещается в конец очереди и ждет своей очереди на выполнение.

o Если процессор свободен, то выбирается следующий поток из очереди для выполнения.

o При появлении нового потока он добавляется в конец очереди.

4. Пример работы алгоритма:

o Единая очередь содержит 5 потоков с временем выполнения 4, 6, 2, 8, 10.

o Величина кванта процессорного времени составляет 3 единицы.

o Количество процессоров - 1.

o Кратчайшая задача - первая.

o Поток с временем выполнения 4 получает первый квант процессорного времени и завершается.

o Поток с временем выполнения 6 получает следующий квант процессорного времени и остается незавершенным.

o Поток с временем выполнения 2 получает следующий квант процессорного времени и завершается.

o Поток с временем выполнения 8 получает следующий квант процессорного времени и остается незавершенным.

o Поток с временем выполнения 10 получает следующие два кванта процессорного времени и завершается.

o Среднее время выполнения коротких потоков составляет (4+2)/2 = 3 единицы времени.

o Среднее время выполнения длинных потоков составляет (6+8+10)/3 = 8 единиц времени.

o Количество выполненных потоков за определенный период времени - 5.

о Среднее время ожидания для каждого потока:

- Поток с временем выполнения 4: 0 единиц времени.
- Поток с временем выполнения 6: 1 единица времени.
- Поток с временем выполнения 2: 0 единиц времени.
- Поток с временем выполнения 8: 2 единицы времени.
- Поток с временем выполнения 10: 4 единицы времени.

о Процент использования процессорного времени - (4+3+2+8+6)/(5*3) = 66.67%.

Заключение

В алгоритмах, основанных на квантовании, какую бы цель они не преследовали (предпочтение коротких или длинных задач, компенсация недоиспользованного кванта или минимизация накладных расходов, связанных с переключениями), не используется никакой предварительной информации о задачах. При поступлении задачи на обработку ОС не имеет никаких сведений о том, является ли она короткой или длинной, насколько интенсивными будут ее запросы к устройствам ввода-вывода, насколько важно ее быстрое выполнение и т. д. Дифференциация обслуживания при квантовании базируется на «истории существования» потока в системе.

В целом, модель планирования потоков, основанная на квантовании, является эффективным инструментом для управления процессами в многозадачной среде. Она позволяет обеспечить справедливое распределение ресурсов между процессами и уменьшить вероятность возникновения блокировок. Однако, для достижения наилучших результатов необходимо

тщательно настраивать параметры модели в зависимости от конкретных условий использования.