Кооперация процессов и основные аспекты ее логической организации

Кооперация процессов и основные аспекты ее логической организации

- Для нормального функционирования процессов операционная система старается максимально обособить их друг от друга. Каждый процесс имеет собственное адресное пространство, нарушение которого, как правило, приводит к аварийной остановке процесса. Каждому процессу по возможности предоставляются свои дополнительные ресурсы. Тем не менее для решения некоторых задач процессы могут объединять свои усилия.
- Для достижения поставленной цели различные процессы могут исполняться псевдопараллельно на одной вычислительной системе или параллельно на разных вычислительных системах, взаимодействуя между собой.

Причины кооперации

- • *Повышение скорости работы*. Пока один процесс ожидает наступления некоторого события, другие могут заниматься полезной работой, направленной на решение общей задачи.
- • *Совместное использование данных*. Различные процессы могут работать с одной и той же динамической базой данных или с разделяемым файлом, совместно изменяя их содержимое.

Кооперация процессов и основные аспекты ее логической организации

- Для удобства работы пользователя, желающего, например, редактировать и отлаживать программу одновременно. В этой ситуации процессы редактора и отладчика должны уметь взаимодействовать друг с другом. «Общение» процессов обычно приводит к изменению их поведения в зависимости от полученной информации. Если деятельность процессов остается неизменной при любой принятой ими информации, то это означает, что они на самом деле в «общении» не нуждаются.
- Процессы, которые влияют на поведение друг друга путем обмена информацией, принято называть *кооперативными* или *взаимодействующими* процессами, в отличие от *независимых* процессов, не оказывающих друг на друга никакого воздействия

Категории средств обмена информацией

- По объему передаваемой информации и степени возможного воздействия на поведение другого процесса все средства такого обмена можно разделить на три категории.
- Сигнальные.
- Канальные.
- Разделяемая память.

Категории средств обмена информацией. Сигнальные

- Передается минимальное количество информации — один бит, «да» или «нет». Используются, как правило, для извещения процесса о наступлении какого-либо события. Степень воздействия на поведение процесса, получившего информацию, минимальна. Все зависит от того, знает ли он, что означает полученный сигнал, надо ли на него реагировать и каким образом.

Категории средств обмена информацией. Канальные

- «Общение» процессов происходит через линии связи, предоставленные операционной системой. Объем передаваемой информации в единицу времени ограничен пропускной способностью линий связи. С увеличением количества информации возрастает и возможность влияния на поведение другого процесса.

Категории средств обмена информацией. Разделяемая память

- Два или более процессов могут совместно использовать некоторую область адресного пространства. Созданием разделяемой памяти занимается операционная система. Возможность обмена информацией максимальна, как, впрочем, и влияние на поведение другого процесса, но требует повышенной осторожности.
- Использование разделяемой памяти для передачи/получения информации осуществляется с помощью средств обычных языков программирования, в то время как сигнальным и канальным средствам коммуникации для этого необходимы специальные системные вызовы. Разделяемая память представляет собой наиболее быстрый способ взаимодействия процессов в одной вычислительной системе.

Логическая организация механизма передачи информации

- Прежде всего необходимо определиться со способом адресации при использовании средства связи.
- Различают два способа адресации: *прямую* и *непрямую*.
- В случае прямой адресации взаимодействующие процессы непосредственно общаются друг с другом, при каждой операции обмена данными явно указывая имя или номер процесса, которому информация предназначена или от которого она должна быть получена. Если и процесс, от которого данные исходят, и процесс, принимающий данные, указывают имена своих партнеров по взаимодействию, то такая схема адресации называется *симметричной прямой адресацией*. Ни один другой процесс не может вмешаться в процедуру симметричного прямого общения двух процессов, перехватить посланные или подменить ожидаемые данные. Если только один из взаимодействующих процессов, например передающий, указывает имя своего партнера по кооперации, а второй процесс в качестве возможного партнера рассматривает любой процесс в системе, например ожидает получения информации от произвольного источника, то такая схема адресации называется *асимметричной прямой адресацией*.

Логическая организация механизма передачи информации

- При непрямой адресации данные помещаются передающим процессом в некоторый промежуточный объект для хранения данных, имеющий свой адрес, откуда они могут быть затем изъяты каким-либо другим процессом. При этом передающий процесс не знает, как именно идентифицируется процесс, который получит информацию, а принимающий процесс не имеет представления об идентификаторе процесса, от которого он должен ее получить.

Буферизация

- Для хранения информации при передаче между процессами используется *буфер*. Можно выделить следующие виды буферов:
- Буфер нулевой емкости или отсутствует. Никакая информация не может сохраняться на линии связи. В этом случае процесс, посылающий информацию, должен ожидать, пока процесс, принимающий информацию, не будет готов ее получить (в реальности этот случай никогда не реализуется).
- Буфер ограниченной емкости. Размер буфера равен n то есть линия связи не может хранить до момента получения более чем n единиц информации. Если в момент передачи данных в буфере хватает места, то передающий процесс не должен ничего ожидать. Информация просто копируется в буфер. Если же в момент передачи данных буфер заполнен или места недостаточно, то необходимо задержать работу процесса отправителя до появления в буфере свободного пространства.
- Буфер неограниченной емкости. Теоретически это возможно, но практически плохо не реализуемо. Процесс, посылающий информацию, ни когда не ждет окончания ее передачи и приема другим процессом. При использовании канального средства связи с непрямой адресацией под емкостью буфера обычно понимается количество информации, которое может быть помещено в промежуточный объект для хранения данных.

Модели передачи данных по каналам связи

- Существует две модели— поток *ввода -вывода* и *сообщения*.
- При передаче данных с помощью потоковой модели, операции передачи/приема информации не интересуются содержимым данных. Процесс, прочитавший 100 байт из линии связи, не знает и не может знать, были ли они переданы одновременно, или порциями по 20 байт, пришли они от одного процесса или от разных. Данные представляют собой простой поток байтов, без какой-либо их интерпретации со стороны системы. Примерами потоковых каналов связи могут служить pipe и FIFO.

Модели передачи данных по каналам связи

- Одним из наиболее простых способов передачи информации между процессами по линиям связи является передача данных через pipe (канал, труба или конвейер). Такой способ реализует потоковую модель ввода/вывода. Информацией о расположении трубы в операционной системе обладает только процесс, создавший ее. Этой информацией он может поделиться исключительно со своими наследниками — процессами-детьми и их потомками. Поэтому использовать pipe для связи между собой могут только родственные процессы, имеющие общего предка, создавшего данный канал связи.
- Если разрешить процессу, создавшему трубу, сообщать о ее местонахождении в системе другим процессам, сделав вход и выход трубы каким-либо образом видимыми для всех остальных, например, зарегистрировав ее в операционной системе под определенным именем, мы получим объект, который принято называть FIFO или именованный pipe. Именованный pipe может использоваться для организации связи между любыми процессами в системе.

Модели передачи данных по каналам связи

- В модели сообщений процессы налагают на передаваемые данные некоторую структуру. Весь поток информации они разделяют на отдельные сообщения, вводя между данными, по крайней мере, границы сообщений. Кроме того, к передаваемой информации могут быть присоединены указания на то, кем конкретное сообщение было послано и для кого оно предназначено. Все сообщения могут иметь одинаковый фиксированный размер или могут быть переменной длины.
- И потоковые линии связи, и каналы сообщений всегда имеют буфер конечной длины.

Надежность средств связи

- Способ коммуникации считается надежным, если при обмене данными выполняются четыре условия:

1. Не происходит потери информации.
2. Не происходит повреждения информации.
3. Не появляется лишней информации.
4. Не нарушается порядок данных в процессе обмена

Каким образки достигается надежность в вычислительной системе?

- Например, при передаче сообщениями. Для обнаружения повреждения информации каждое передаваемое сообщение снабжается некоторой контрольной суммой, вычисленной по посланной информации. При приеме сообщения контрольную сумму вычисляется заново и проверятся ее соответствие пришедшему значению. Если данные не повреждены, то контрольные суммы совпадают и подтверждается правильность их получения. Если данные повреждены (контрольные суммы не совпадают), и считается, что сообщение не поступило. Вместо контрольной суммы также используется специальное кодирование передаваемых данных с помощью кодов, исправляющих ошибки. Такое кодирование позволяет при числе искажений информации, не превышающем некоторого значения, восстановить первоначальные неискаженные данные. Для гарантии правильного порядка получения сообщений их нумеруют. При приеме сообщения с номером, не соответствующим ожидаемому, с ним поступают, как с утерянным.
- Подобные действия могут быть возложены: на операционную систему; на процессы, обменивающиеся данными; совместно на систему и процессы, разделяя их ответственность.

Прекращения обмена

- Здесь нужно выделить два аспекта: требуются ли от процесса какие-либо специальные действия по прекращению использования средства коммуникации и влияет ли такое прекращение на поведение других процессов. Для способов связи, которые не подразумевали никаких инициализирующих действий, обычно ничего специального для окончания взаимодействия предпринимать не надо.
- Если же установление связи требовало некоторой инициализации, то, как правило, при ее завершении бывает необходимо выполнить ряд операций, например сообщить операционной системе об освобождении выделенного связного ресурса.

Нити

- Любой отдельно взятый процесс в мультипрограммной системе никогда не может быть выполнен быстрее, чем при работе в однопрограммном режиме на том же вычислительном комплексе. Тем не менее, если алгоритм решения задачи обладает определенным внутренним параллелизмом, то можно ускорить его работу, организовав взаимодействие нескольких процессов. Параллелизм одного процесса достигается за счет нитей.
- *Нить исполнения* или просто *нить* (в англоязычной литературе используется термин *thread)*. Нити процесса разделяют его программный код, глобальные переменные и системные ресурсы, но каждая нить имеет собственный программный счетчик, свое содержимое регистров и свой стек. Процесс можно представить как совокупность взаимодействующих нитей и выделенных ему ресурсов.

Нити

- Процесс, содержащий всего одну нить исполнения считается «традиционным процессом».
- Иногда нити называют облегченными процессами или мини-процессами, так как во многих отношениях они подобны традиционным процессам. Нити, как и процессы, могут порождать нити-потомки, правда, только внутри своего процесса, и переходить из одного состояния в другое.
- Поскольку нити одного процесса разделяют существенно больше ресурсов, чем различные процессы, то операции создания новой нити и переключения контекста между нитями одного процесса занимают значительно меньше времени, чем аналогичные операции для процессов в целом.
- Различают операционные системы, поддерживающие нити на уровне ядра и на уровне библиотек.

Нити

- Для операционных систем, поддерживающих нити на уровне ядра планирование использования процессора происходит в терминах нитей, а управление памятью и другими системными ресурсами остается в терминах процессов.
- В операционных системах, поддерживающих нити на уровне библиотек пользователей, и планирование процессора, и управление системными ресурсами осуществляются в терминах процессов. Распределение использования процессора по нитям в рамках выделенного процессу временного интервала осуществляется средствами библиотеки. В подобных системах блокирование одной нити приводит к блокированию всего процесса, ибо ядро операционной системы не имеет представления о существовании нитей. По сути дела, в таких вычислительных системах просто имитируется наличие нитей исполнения.