1 Тупики в распределенных системах

[...]

Транзакции в системе распределенной БД выполняются на нескольких сайтах и используют данные на этих сайтах. При этом объем данных распределяется между сайтами неравномерно. Кроме того, варьируется время выполнения на каждом сайте. В результате одна и та же транзакция может быть активной на одних сайтах и неактивной на других.

Если на сайте находятся две конфликтующие транзакции, возможна ситуация, при которой одна из транзакций находится не в активном состоянии. То есть важно местоположение транзакций (**проблема местоположения транзакций, transaction location**).

Для решения данной проблемы используется модель Daisy Chain. Она предполагает хранение доп информации о транзакции. При перемещении транзакции с одного сайта на другой предлагается хранить список необходимых таблиц и список сайтов [там еще что-то], а также список блокировок с типами.

Вся дополнительная информация о транзакциях должна быть отправлена на все заинтересованные сайты для анализа.

Пример: Chundy-Misza-Haas algorithm

Процессы запрашивают сразу несколько необходимых им ресурсов, что уменьшает количество транзакций. Если при очередном запросе ресурс занят, то процесс генерирует специальное сообщение и посылает его другим взаимодействующим с ним процессам. В сообщении указывается три поля: номер процесса, отправляющего сообщение; номер процесса, отправляющего сообщение (да-да, то же самое); номер процесса, которому посылается сообщение.

Процесс, получивший сообщение, проверяет, ждет ли он сам ресурс, запрашиваемый другим процессом. Если ждет, то во второе поле он записывает свой номер, а в третье поле записывает номер процесса, от которого ждет освобождение ресурса. Затем посылает сообщение дальше.

В результате, если процесс получит сообщение обнаружит свой номер

в первом и третьем поле, то тем самым он обнаружит, что система находится в тупике. Такое сообщение называется зондом.

[картинка]

А как блокированный процесс может что-то делать (анализировать, принимать, посылать)? Дополнительный поток, который может каким-то чудом выполнять работу.

2 Управление памятью

Речь идет об оперативной памяти.

Память как ресурс неоднородна. В системе может рассматриваться иерархия памяти, в зависимости от близости к процессору.

[что-то про программируемые процессоры и микрокоманды]

!!! Процессор собственной памяти не имеет и постоянно обменивается информацией с оперативной памяти (считывает команды и данные из ОП и записывает данные в ОП).

По архитектуре фон Неймана, процессор может выполнять только те программы, что находятся в ОП. [про работу программ].

Вторичная память (флеш-память, SSD и тд). Несмотря на то, что ее физические принципы изменились, вторичная память остается блочной. Вторичная память является энергонезависимой и предназначена для долговременного хранения данных.

У вторичной памяти есть еще одна задача: поддержка в современных системах пэйджинга. То есть компьютер имеет память, аналогичную дисковой, и дисковое адресное пространство делится на две неравные части. Большая часть используется для хранения данных, при этом имеются в виду т.н. обычные файлы (regular).

Файл - поименованная совокупность данных (имеет имя). Эта совокупность может быть бессмысленной. Но чтобы можно было обратиться к этим данным, то эти данные должны быть идентифицироваться. В UNIX идентификатором файла является inode, но в системе файлы идентифицируются именем.

Очевидно, что эта иерархия памяти, строящаяся по принципу близости к процессору, имеет определенные характеристики. В частности, память должна иметь то же быстродействие, что и процессор. Если это требование не будет выполняться, то отпадает смысл повышать производительность процессора.

В современных системах имеется такая ситуация, при которой быстродействие ОП отстает на порядок от производительности процессора ввиду

разных техник производства. Поэтому в процессорах есть кеши трех уровней. Кеш дает возможность решить проблему с отставанием.

В современных системах есть одноуровневая память (файлы, отображаемые в память) — для пэйджинга используется память файла.

2.1 Управление память в старых системах

Принято при управлении памятью рассматривать связанное (программа в памяти занимает непрерывное адресное пространство) и несвязанное (программа может занимать в память непрерывные участки) распределение памяти.

Классификация:

1) Одиночное непрерывное распределение

   Речь идет об однопрограммных системах, в которых в каждый момент времени в памяти находится только одно приложение. При таком распределении находятся две программы: ОС и приложение. Оставшаяся часть — свободная неиспользуемая память.

   [картинки]

   [что-то про DOS]