Алгоритм блокировки (Choked algorithm)

Алгоритм блокировки используется для изменения списка активных пиров, то есть пиров, которым пир раздает данные. Это делает BitTorrent справедливым, потому что пиры «делятся» данными с другими пирами, которые предоставляют им наибольшую скорость загрузки. С точки зрения справедливости, пиры не должны загружать больше, чем они отдают. Так как пиры отдают данные другим пирам, которые позволяют им выполнять загрузку файлов, этот алгоритм отдает предпочтение тем пирам, которые раздают данные, чем тем, которые этого не делают. Этот алгоритм пытается максимально использовать раздачу, например, если два пира имеют низкую скорость загрузки, они могут начать раздавать друг другу данные и оба будут иметь гораздо более высокую скорость загрузки. Экспериментальным путем пир определяет, может ли он получить более высокую скорость передачи данных за счет неиспользованных соединений.

Алгоритм блокировки работает по-разному в состоянии сидера и личера. Когда пир находится в состоянии личера, он должен поддерживать текущую скорость загрузки для пиров, подключенных к нему. Пиры становятся разблокированными в соответствии со следующим алгоритмом:

1. Каждые 10 секунд заинтересованные пиры упорядочиваются по скорости загрузки, и три пира с самой высокой скоростью загрузки становятся разблокированными.
2. Каждые 30 секунд один случайный заинтересованный пир становится разблокированным

Пиры, разблокированные на шаге 1, называются очередными разблокированными пирами (regular unchoked peers – RU). Пиры, разблокированные на шаге 2, называются оптимистически разблокированными пирами (optimistic unchoked peer – OU). Если OU-пир входит в тройку самых быстрых пиров (т.е. одновременно является RU-пиром), то случайным образом выбирается другой заинтересованный пир.

Оптимистическая разблокировка необходима, потому что без нее у пиров нет другого способа обнаружения лучших соединений, чем те, которые установлены у них на текущий момент. Кроме того, без нее новые пиры, у которых нет фрагментов, никогда не смогут получить свой первый фрагмент.

Алгоритм выбора фрагмента (Piece selection algorithm)

Типичное использование BitTorrent -клиента - загрузка статического, не потокового контента. В этом контексте упорядочение фрагментов не имеет значения, и они загружаются в соответствии с алгоритмом «сначала самый редкий» (*rarest first*).

Целью этого алгоритма является увеличение разнообразия доступных фрагментов, что делает количество копий каждого фрагмента одинаковым насколько это возможно. В итоге маловероятно, что у пиров возникнут проблемы с загрузкой фрагментов, содержащих редкие блоки, которые трудно найти. Кроме того, с помощью этого алгоритма у пира всегда будет что предложить другим пирам.

Поскольку каждый пир хранит информацию обо всех фрагментах у пиров, находящихся в его списке, он может определить, сколько копий этих фрагментов доступно для загрузки. Он использует эту информацию, чтобы определить какой фрагмент загрузить первым,

отслеживая самый редкий фрагмент в наборе. Этот набор обновляется каждый раз, когда он получает сообщение *have* или всякий раз, когда пир покидает список.

В случае, когда пир только что присоединился к торренту и не имеет ни одного фрагмента, он использует алгоритм «сначала случайный» (*random first*). Этот метод используется до тех пор, пока пир не скачает 4 фрагмента, а затем он переходит к использованию алгоритма «сначала самый редкий».

Кроме того, при использовании алгоритма «сначала самый редкий» применяется политика строго приоритета, в соответствии с которой остальные блоки фрагмента, соответствующего загружаемому в настоящий момент блоку, имеют приоритет. Таким образом, после того, как пир скачивает какой-либо блок фрагмента, он будет выбирать для загрузки другие блоки того же фрагмента. Это позволит пиру скачивать полные фрагменты вместо того, чтобы иметь отдельные блоки редких фрагментов.

Когда личер близок к завершению загрузки, он переходит в режим завершения (end game mode). Этот режим запускается тогда, когда пир запросил все блоки. В этом режиме пир запрашивает все блоки, которые он не получил, у всех подключенных пиров. Как только он получает блок, он посылает сообщение отмены (*cancel)* другим подключенным пирам. Эти сообщения отмены необходимы для снижения объема трафика, затрачиваемого на пересылку избыточных блоков. Было доказано, что время режима завершения не требует много трафика, так как он очень короткий и конец файла всегда быстро загружается.

Задачи:

1. Реализовать алгоритм блокировки
2. Реализовать алгоритм выбора фрагмента