Лабораторная работа №3 по теме «Алгоритмы кэширования»

Введение

Кэширование – сохранение данных в буфере для ускорения доступа к ним. Ускорение достигается за счёт того, что не происходит «дорогих» операций – каких-либо вычислений, дискового ввода-вывода, передачи данных по сети, повторной генерации html-страницы для пользователя, чтение инструкций из памяти в процессор, и т.д.

Таким образом, *кэш – это память с большой скоростью доступа*, в которой сохраняются данные из памяти с меньшей скоростью доступа. Кэш состоит из набора записей, связанных с элементом (или блоком элементов) в памяти, из которой данные были получены.

Простым примером применения кэширования информации может послужить любой web-браузер. При посещении веб-сайтов он может сохранять на компьютере пользователя данные, повторно использующиеся на страницах (изображения, аудио, и др.) или же страницы целиком, чтобы не загружать их повторно. Так достигается эффект более быстрой загрузки страниц. Данный пример можно назвать программной реализации кэш-памяти.

Алгоритмы кэширования

Ознакомившись с понятием кэш-памяти, следует затронуть вопрос, по какому принципу в кэш-память помещаются данные, ведь размер данных в основной памяти всегда велик по сравнению с размером кэша? Существует несколько алгоритмов кэширования (замещения записей в кэше), рассмотрим их ниже.

MRU (Most Recently Used)

Алгоритм подразумевает вытеснение элементов в порядке времени их использования «от нового к старому». Когда при добавлении элемента в кэше заканчивается место, происходит удаление из него той записи, которая использовалась самой последней. Авторы работы «Semantic Data Caching and Replacement» утверждают, что для схем случайного доступа и циклического сканирования больших наборов данных (иногда называемых схемами циклического доступа) алгоритмы кэширования MRU имеют больше попаданий по сравнению с LRU за счет их стремления к сохранению старых данных. Алгоритмы MRU наиболее полезны в случаях, когда чем старше элемент, тем больше обращений к нему происходит.

LRU (Least Recently Used)

Алгоритм подразумевает вытеснение элементов, которые не запрашивались дольше всего. Как следствие, появляется необходимость хранить время последнего использования элементов в кэше.

LFU (Least Frequency Used)

Идея применения данного алгоритма основывается на том, что в кэше будут находиться данные наиболее «популярные», вероятность же использования той или иной записи оценивается исходя из частоты запросов к записи (необходимо подсчитывать частоту использования каждой записи в кэше).

Выбор того или иного алгоритма кэширования, как правило, обусловлен статистикой обращения к данным. Алгоритмы кэширования, в том числе и описанные выше, могут не применяться в «чистом» виде, например, существует алгоритм ARC (Adaptive Replacement Cache), разработанный компанией IBM, который использует

концепции алгоритмов LRU и LFU. Технически это выглядит как разделение кэша на два списка, в одном записи замещаются в соответствии с алгоритмом LRU, а в другом – в соответствии с LFU.

Задание

В данной лабораторной работе необходимо реализовать контейнер, использующийся в качестве кэша, поведение которого при добавлении элементов соответствует одному из перечисленных выше алгоритмов.

Варианты

1. MRU
2. LRU
3. LFU

Номер варианта студента соответствует порядковому номеру этого студента в списке группы. Нумерация циклическая.