Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
Высшего образования
Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа 2 по Операционным системам

Базовый трек

Вариант MacOS, LRU, io-thpt-read

Группа: P3316
Выполнил:
Сиразетдинов А.Н.
Проверил:
Гиниятуллин Арслан

Г. Санкт-Петербург
2024

Оглавление

Текст задания

Задание

Для оптимизации работы с блочными устройствами в ОС существует кэш страниц с данными, которыми мы производим операции чтения и записи на диск. Такой кэш позволяет избежать высоких задержек при повторном доступе к данным, так как операция будет выполнена с данными в RAM, а не на диске (вспомним пирамиду памяти).

В данной лабораторной работе необходимо реализовать блочный кэш в пространстве пользователя в виде динамической библиотеки (dll или so). Политику вытеснения страниц и другие элементы задания необходимо получить у преподавателя.

При выполнении работы необходимо реализовать простой API для работы с файлами, предоставляющий пользователю следующие возможности:

Открытие файла по заданному пути файла, доступного для чтения. Процедура возвращает некоторый хэндл на файл. Пример:
```
int lab2_open(const char *path).
```

Закрытие файла по хэндлу. Пример:
```
int lab2_close(int fd).
```

Чтение данных из файла. Пример:
```
ssize_t lab2_read(int fd, void buf[.count], size_t count).
```

Запись данных в файл. Пример:
```
ssize_t lab2_write(int fd, const void buf[.count], size_t count).
```

Перестановка позиции указателя на данные файла. Достаточно поддержать только абсолютные координаты. Пример:
```
off_t lab2_lseek(int fd, off_t offset, int whence).
```

Синхронизация данных из кэша с диском. Пример:
```
int lab2_fsync(int fd).
```

Операции с диском разработанного блочного кеша должны производиться в обход page cache используемой ОС.

В рамках проверки работоспособности разработанного блочного кэша необходимо адаптировать указанную преподавателем программу-загрузчик из ЛР 1, добавив использование кэша. Запустите программу и убедитесь, что она корректно работает. Сравните производительность до и после.

Ограничения

Программа (комплекс программ) должна быть реализован на языке C или C++.

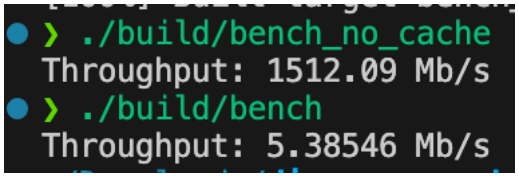
Если по выданному варианту задана политика вытеснения Optimal, то необходимо предоставить пользователю возможность подсказать page cache, когда будет совершен следующий доступ к данным. Это можно сделать либо добавив параметр в процедуры read и write (например, ssize_t lab2_read(int fd, void buf[.count], size_t count, access_hint_t hint)), либо добавив еще одну функцию в API (например, int lab2_advice(int fd, off_t offset, access_hint_t hint)). access_hint_t в данном случае – это абсолютное время или временной интервал, по которому разработанное API будет определять время последующего доступа к данным.

Запрещено использовать высокоуровневые абстракции над системными вызовами. Необходимо использовать, в случае Unix, процедуры libc.

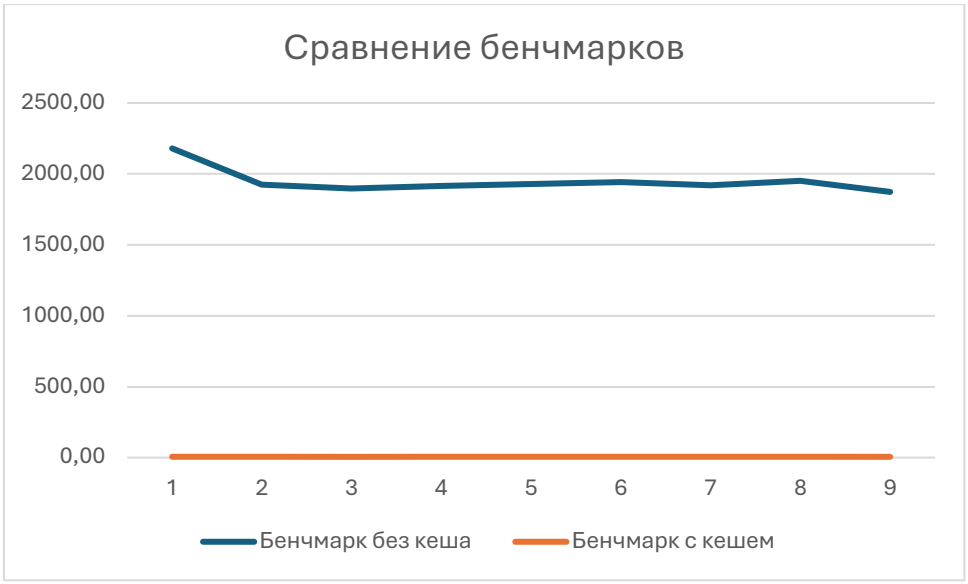
Код программы

https://github.com/secs-dev-os-course/diy-page-cache-Azat2202/pull/1

Сравнение нагрузчика с наличием и без кеша



Получена огромная разница



Вывод

В процессе выполнения лабораторной работы я узнал про работу кешей в операционной системе и написал собственный страничный кеш и сравнил работу системы с ним и без него. Я считаю выполнена гениальная задача по созданию кеша который замедляет систему

Вывод

В лабораторной работе я реализовал свой shell, который позволяет запускать программы и выводить время их выполнения.

Я разработал две программы нагрузчика и разными способами смотрел как они нагружают систему. С помощью элементов профилирования я построил FlameGraph для анализа времени исполнения программы. Так же анализировал комбинированный вариант нескольких потоков разных загрузчиков

В процессе работы я научился использовать утилиты для диагностики и профилирования