Разработка и реализация алгоритмов распределенной сборки мусора в системе LuNA

Выполнил: Плешков А. В.
Руководитель: Перепёлкин В. А.

Задачи

Основной целью школы стало: успешно и эффективно запустить модуль для реальной программы.

Идея решения

- Определить количество обращений к фрагменту данных во время компиляции программы
- Расставить рекомендации к удалению фрагментов данных

В код компилятора языка LuNA был встроен модуль, получающий информацию от компилятора, согласно реализованному API.

Проблемы, возникшие во время реализации

Первоначально, при реализации не были учтены зависимости по данным, например, в случае, когда существует потребление, происходящие по некоторому условию:

```
if (x > y) { do_smth(y); }
```

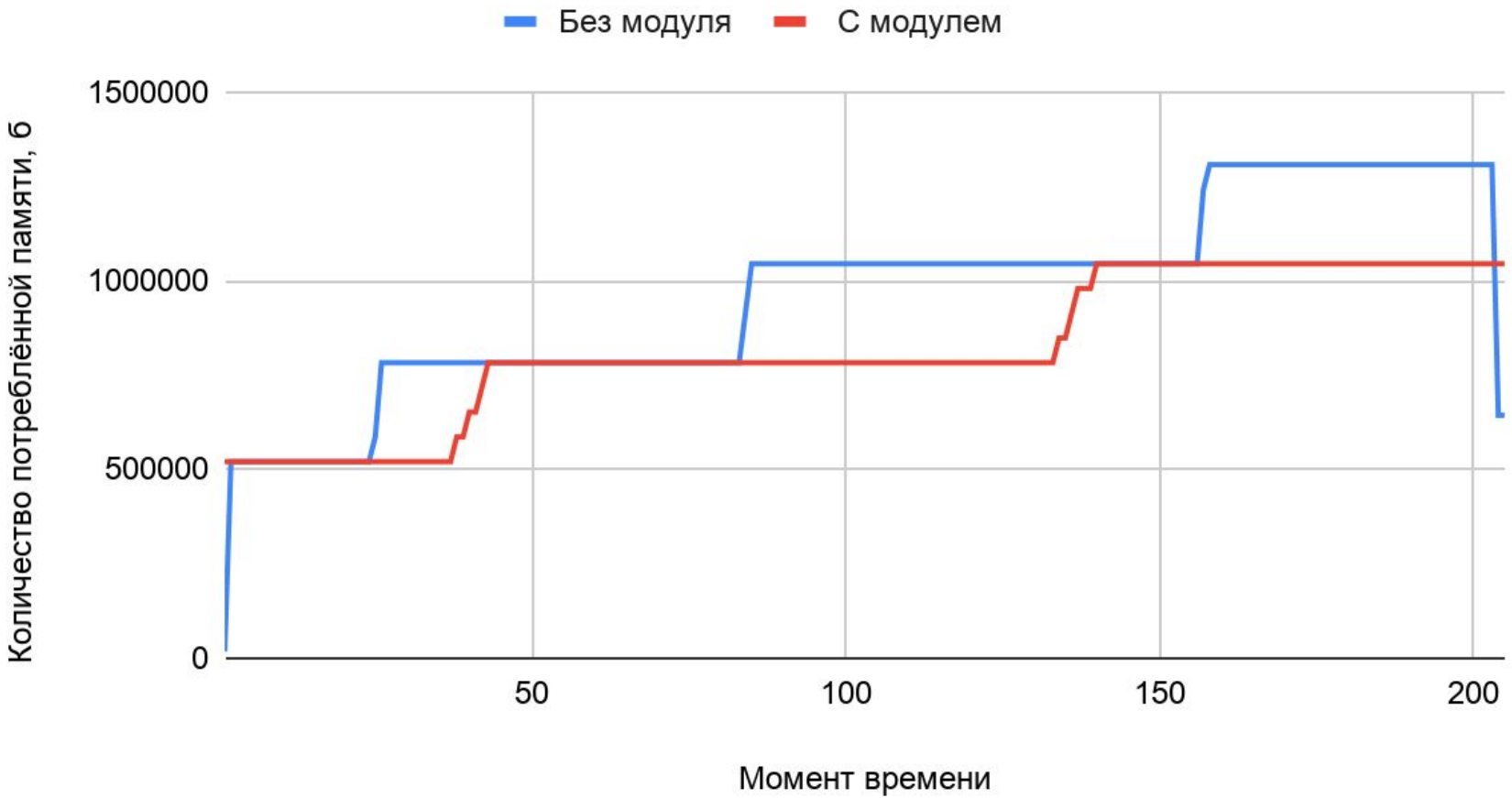
Фрагмент данных y будет употреблён дополнительно 1 раз, если условие выполнится. Рекомендация req_count поддерживает динамический подсчёт использований, однако для его вычисления необходимо предварительно запросить все зависимые фрагменты данных, то есть x в данном случае.

Частой операцией при работе с массивами является обращение по индексу, так что поддержка подобных операций необходима. Сделать это можно следующим образом - увеличивать число использований элемента массива, если к нему было совершенно прямое обращение. Однако, такие операции требуют особой структуры данных, так что поддержка индексации ещё не была реализована.

Тестирование

Было запланировано протестировать модуль на перемножении матриц, но в связи с тем, что в перемножении матриц используется индексация элементов массива, для тестирования программа была немного урезана, всё таки в контексте данной работы нас интересует потребление памяти, а не корректность программы.

Запуск программы



Заключение

- Расширены возможности модуля расставления рекомендаций
- Исправлены проблемы, неучтённые в ходе практики
- Было проведено тестирование модуля