Разработка алгоритмов анализа производительности фрагментированных программ

Выполнил:
Литвинов Василий Сергеевич

Научный руководитель:
Киреев Сергей Евгеньевич,
н.с. ИВМиМГ СО РАН

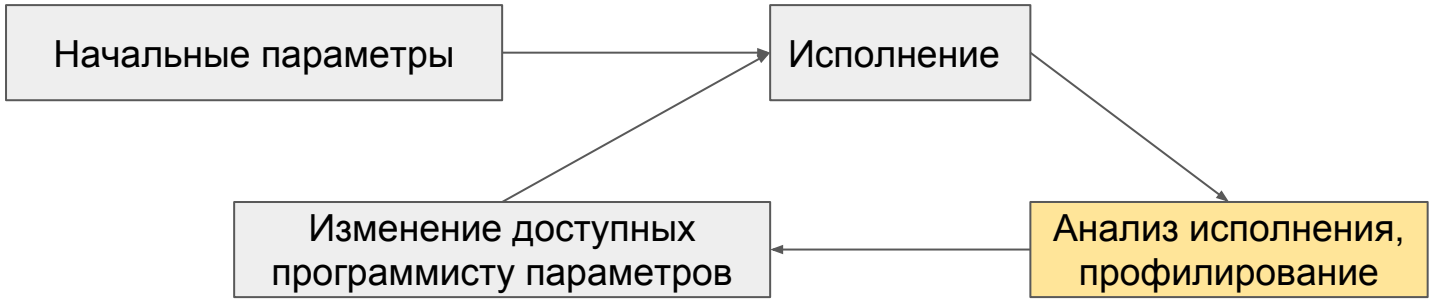
Новосибирск, 12 июля 2017

Технология фрагментированного программирования и система LuNA

- Решение системных задач параллельного программирования.
- Язык и исполнительная система, разработанные в ИВМиМГ СО РАН, реализующие технологию фрагментированного программирования. [1]

1. Malyshkin V.E., Perepelkin V.A. LuNA Fragmented Programming System, Main Functions and Peculiarities of Run-Time Subsystem // PaCT-2011 proceedings, Springer, LNCS 6873 (2011), pp. 53-61.

Производительность параллельных программ



<u>Проблема</u>: Обнаружение причин плохой производительности требует проверки многих гипотез.

Для упрощения решения этой задачи создаются профилировщики.

Это актуально и для технологии фрагментированного программирования.

Требования к средствам профилирования

Хороший профилировщик должен предоставлять такие характеристики исполнения, которые:

- Объясняют полученную производительность, указывают на возможные проблемы, ограничивающие производительность программы
- Дают информацию в терминах, которыми пользуется программист (в данном случае — терминах технологии фрагментированного программирования)

Обзор существующих средств профилирования параллельных программ

- Указывают на проблемы с производительностью
  - MPE + Jumpshot
  - Extrae / Paraver
  - Intel Trace Analyzer and Collector
  - Scalasca (KOJAK)
  - ...
- Предоставляют информацию в терминах ТФП
  - LuNA Evlog Analyze

Результат обзора: нет систем, удовлетворяющих обоим критериям.

Постановка задачи

Целью данной работы является разработка алгоритмов анализа исполнения фрагментированных программ и создание инструмента, позволяющего получить “высокоуровневые” характеристики исполнения LuNA-программ, определение которых позволит программисту сделать выводы о полученной производительности.

SLOW

Причины, ограничивающие масштабируемость параллельных программ (причины простоев вычислительных ресурсов) (SLOW) [1]:

- **S**tarvation -- низкая степень параллелизма на данном этапе исполнения.
- **L**atency -- задержки, обусловленные доступом к данным.
- **O**verhead -- работа по решению системных задач ПП, которой не возникает в последовательной программе.
- **W**aiting for contention resolution -- задержки синхронизации доступа к общим данным.

1. H. Kaiser, T. Heller, B. Adelstein-Lelbach, A. Serio, D. Fey, HPX: A Task Based Programming Model in a Global Address Space // Proc. of the 8th International Conference on PGAS Programming Models, 2014.

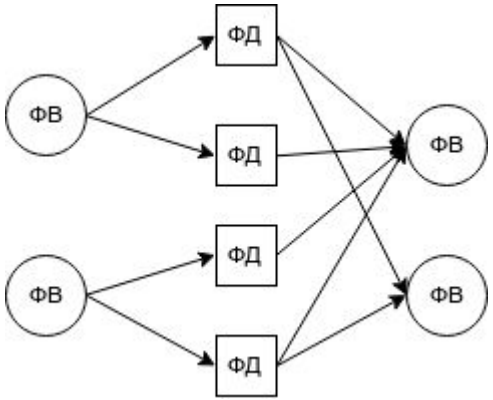
Постановка задачи

Были выделены следующие задачи:

- Оценить применимость факторов SLOW в качестве основы для характеристик исполнения фрагментированной программы
- Построить упрощенной модели исполнения фрагментированного алгоритма, достаточной для определения значений выбранных характеристик, независимой от реализации конкретной исполнительной системы.
- Построить алгоритм получения выбранных характеристик по заданному исполнению в соответствии с построенной моделью.
- Реализовать алгоритм применительно к LuNA-программам.
- Проверить корректность реализации с помощью тестирования.

Выбор объекта для анализа

- В основе языка LuNA лежит представление алгоритма в виде потенциально бесконечного графа зависимостей, дополненное необходимыми средствами для
  - представления любых вычислимых функций,
  - удобства записи численных алгоритмов.
- Исполнительная система реализует некоторый конечный граф зависимостей - “граф исполнения”.
- Для целей работы достаточно рассматривать исполнение в соответствии с “графом исполнения”.



ФД - фрагмент данных
ФВ - фрагмент вычислений

SLOW в ТФП

- **S**tarvation -- связано с отсутствием на узле готовых к исполнению ФВ из-за неудовлетворенных информационных зависимостей
- **L**atency -- задержки ожидания приема готовых ФД, являющимися входными для ФВ.
- **O**verhead -- возникает из-за накладных расходов на работу исполнительной системы.
- **W**aiting for contention resolution -- применительно к технологии фрагментированного программирования не имеет смысла

В качестве характеристик *TStarvation*,*TLatency*,*TOverhead* (SLO) было предложено взять вклады соответствующих факторов в формирование интервалов простоя.

Эти характеристики объясняют полученную производительность программы в терминах ТФП.

Модель исполнения фрагментированного алгоритма

Опишем упрощенную виртуальную машину, работа которой будет моделировать работу реальной исполнительной системы.

- Ресурсы: N вычислительных узлов на K ядер, на каждом узле работают K или меньше рабочих потоков и M потоков исполнительной системы.
- ФВ исполняются рабочими потоками. Известны времена получения входных ФД, начала и конца исполнения каждого ФВ.
- ФД вычисляются ФВ и потребляются ФВ. ФД могут быть переданы между узлами. Известны времена вычисления каждого ФВ.

Жизненные циклы ФД и ФВ

- Жизненный цикл фрагмента можно представить в виде конечного автомата, для которого известны моменты смены состояния:

Диаграмма КА ФД:

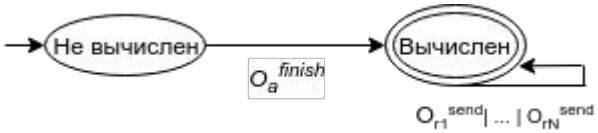


Диаграмма КА ФВ:

- Исполнение всей фрагментированной программы описывается работой конечного автомата, полученного путем объединения конечных автоматов всех ФД и ФВ данного графа исполнения.

Расчет характеристик по модели исполнения

- Анализируется каждый интервал простоя рабочего потока в порядке времени завершения простоя
- Затем результаты анализа суммируются для простоев в заданном временном промежутке для получения интегральных характеристик

Схема анализа одного интервала простоя

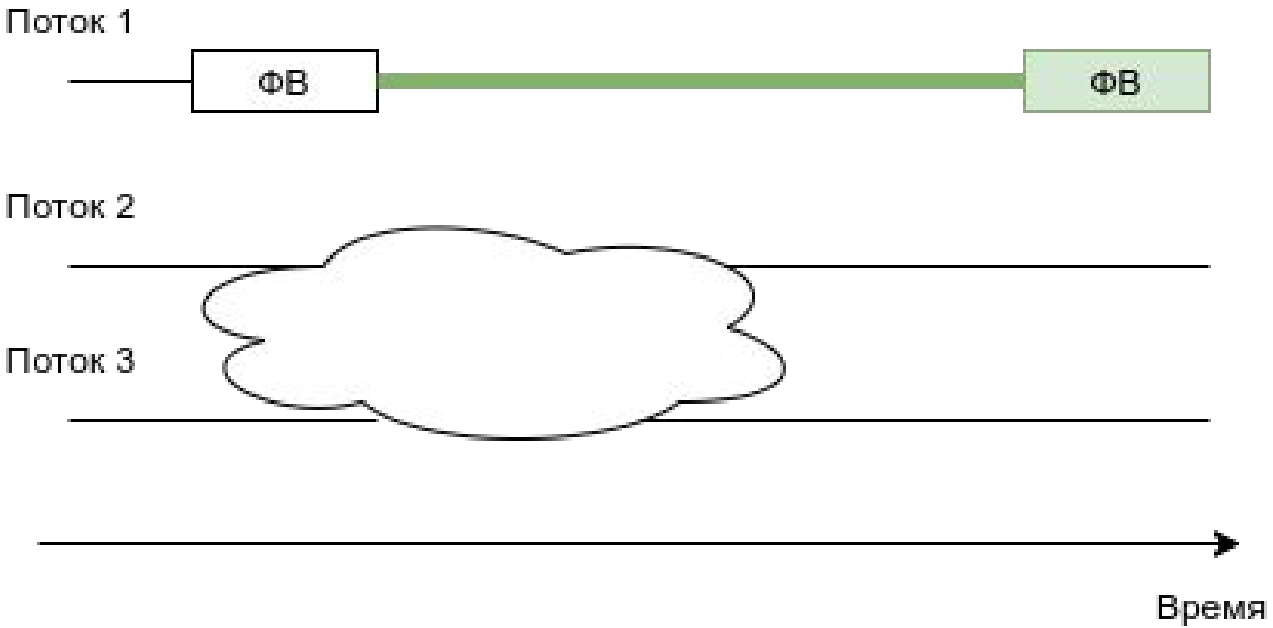


Схема анализа одного интервала простоя

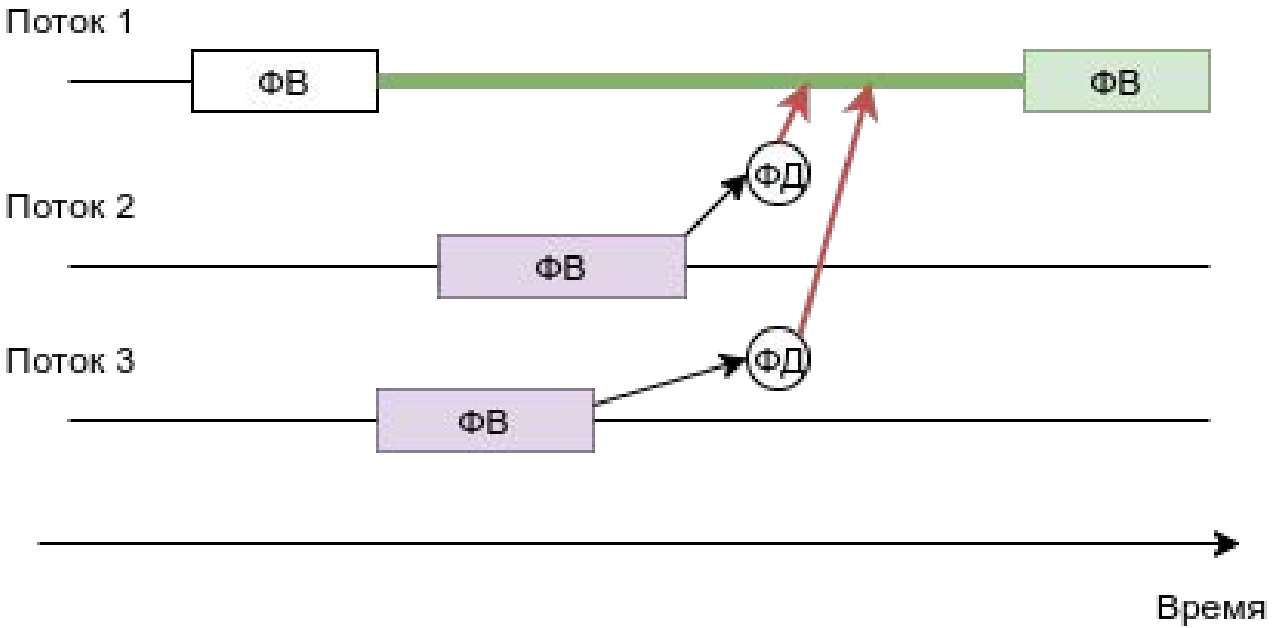


Схема анализа одного интервала простоя

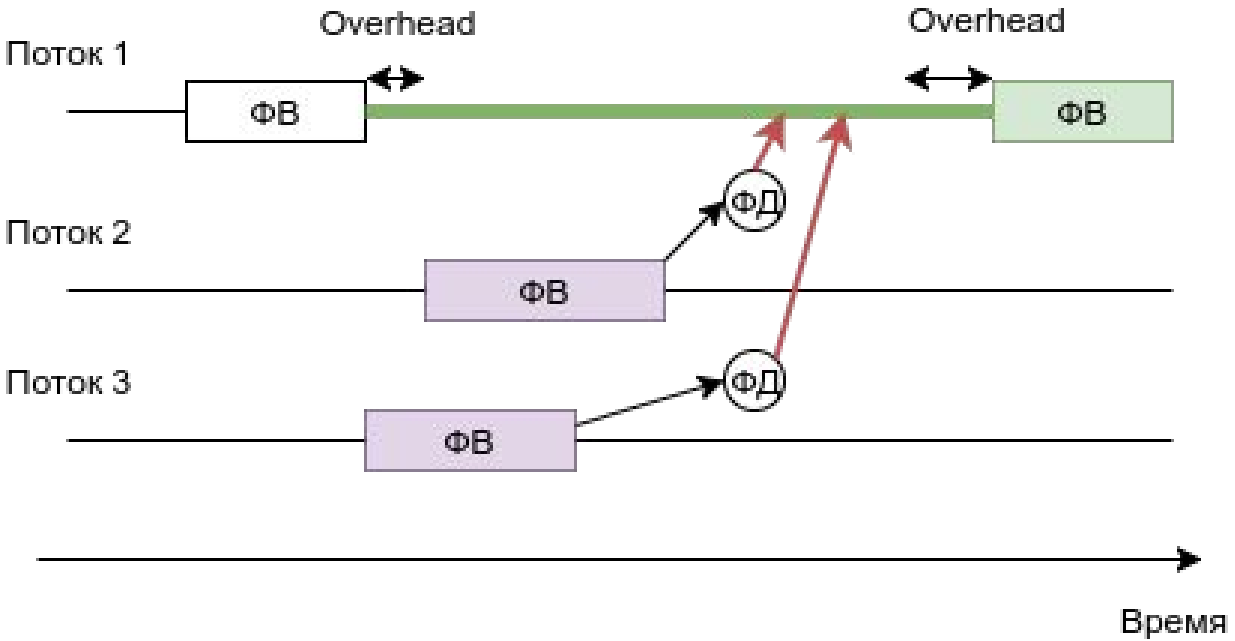


Схема анализа одного интервала простоя

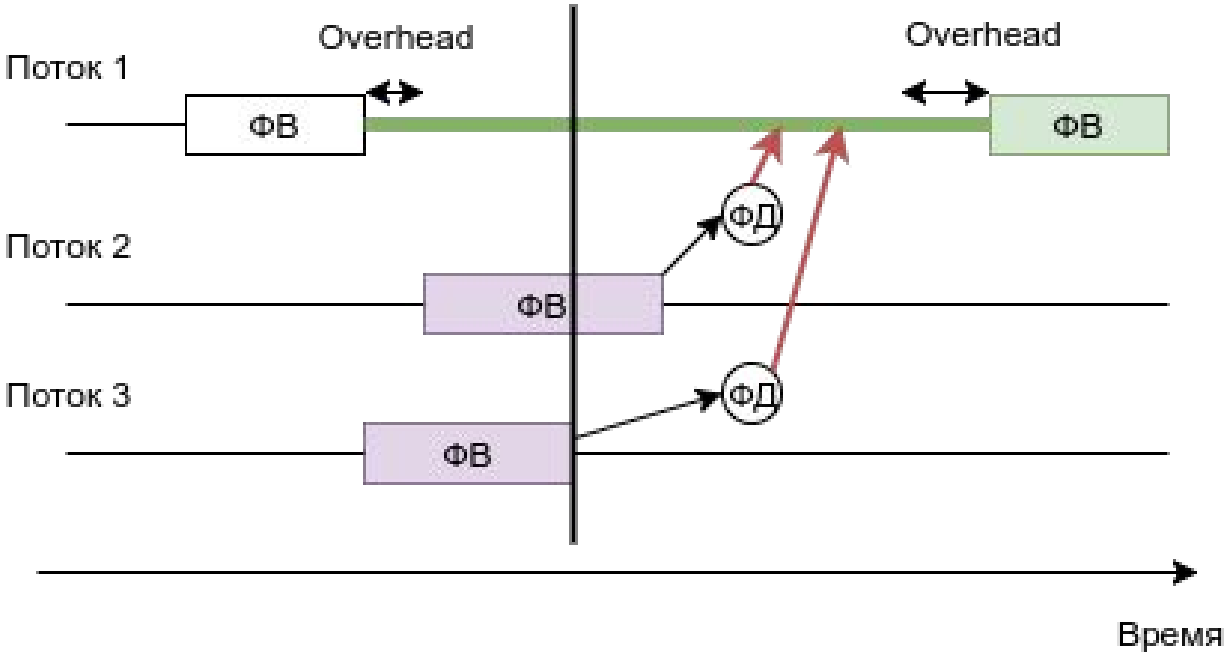


Схема анализа одного интервала простоя

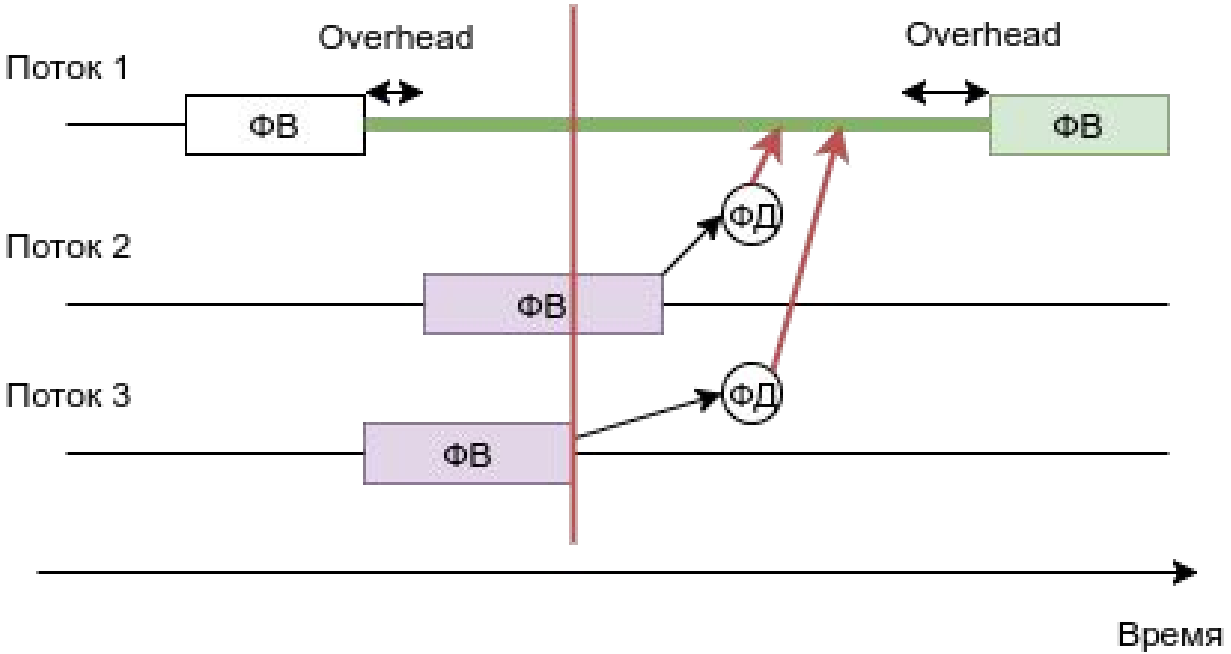


Схема анализа одного интервала простоя

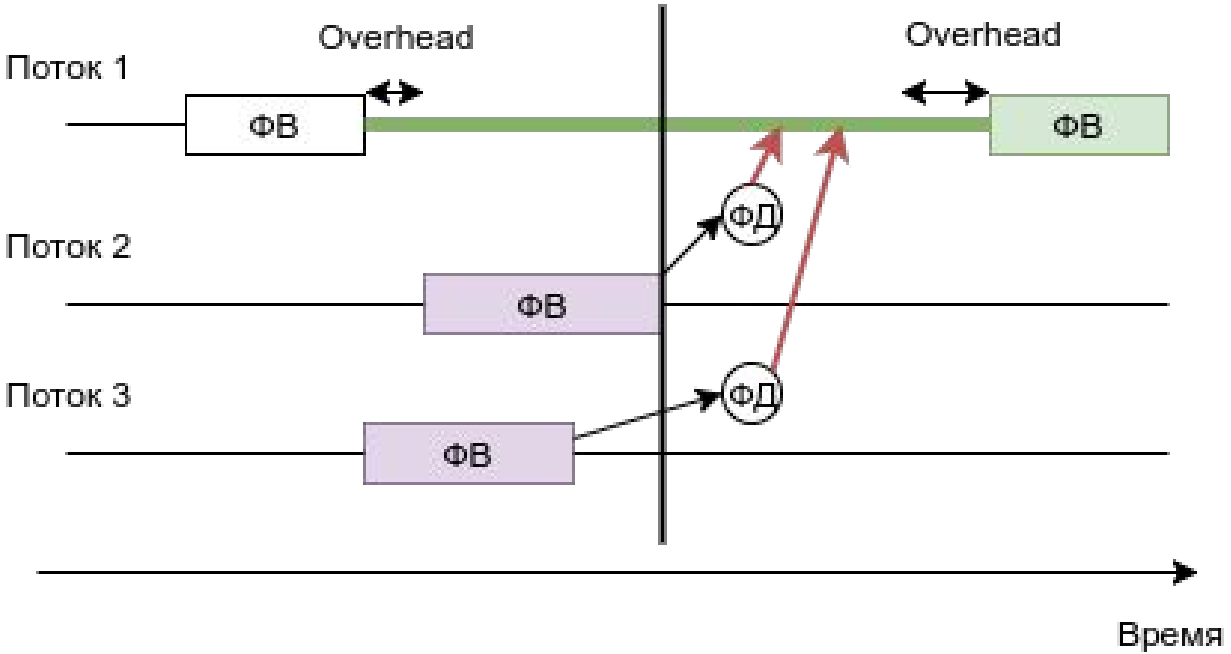


Схема анализа одного интервала простоя

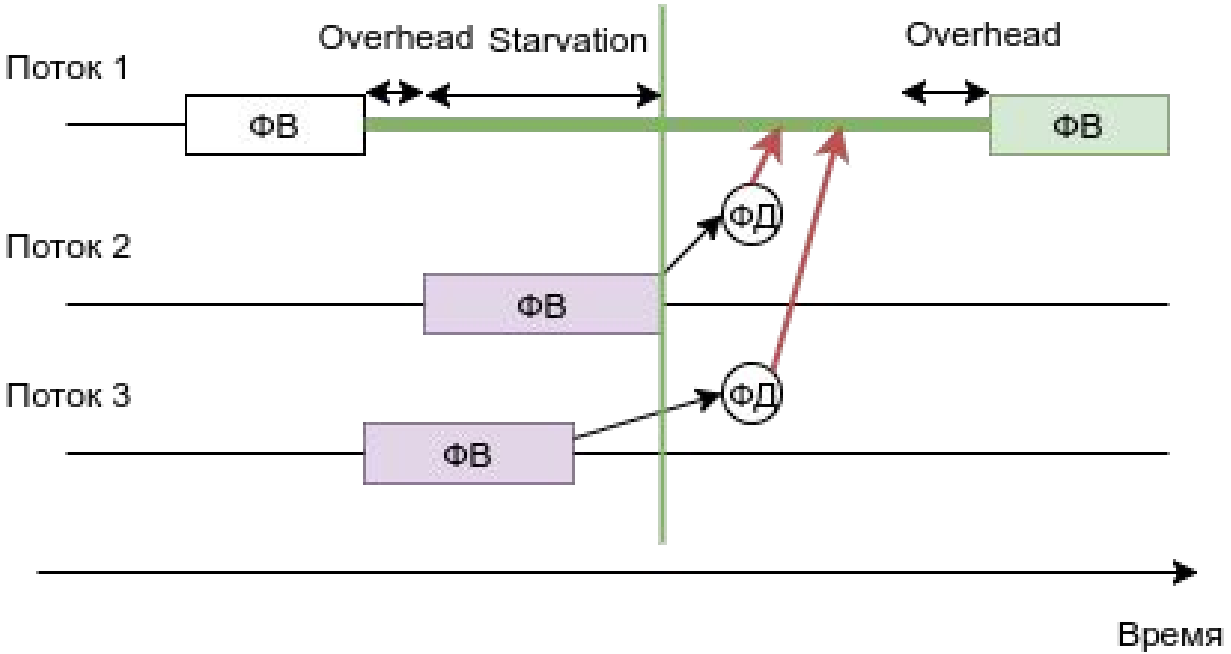


Схема анализа одного интервала простоя

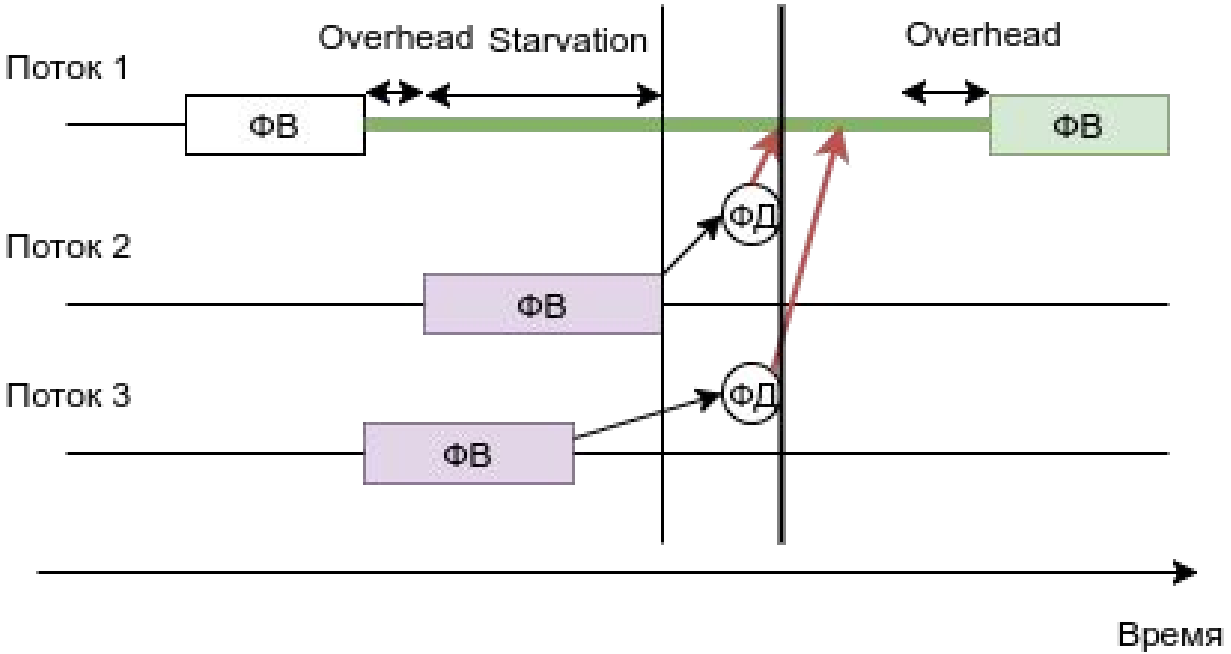


Схема анализа одного интервала простоя

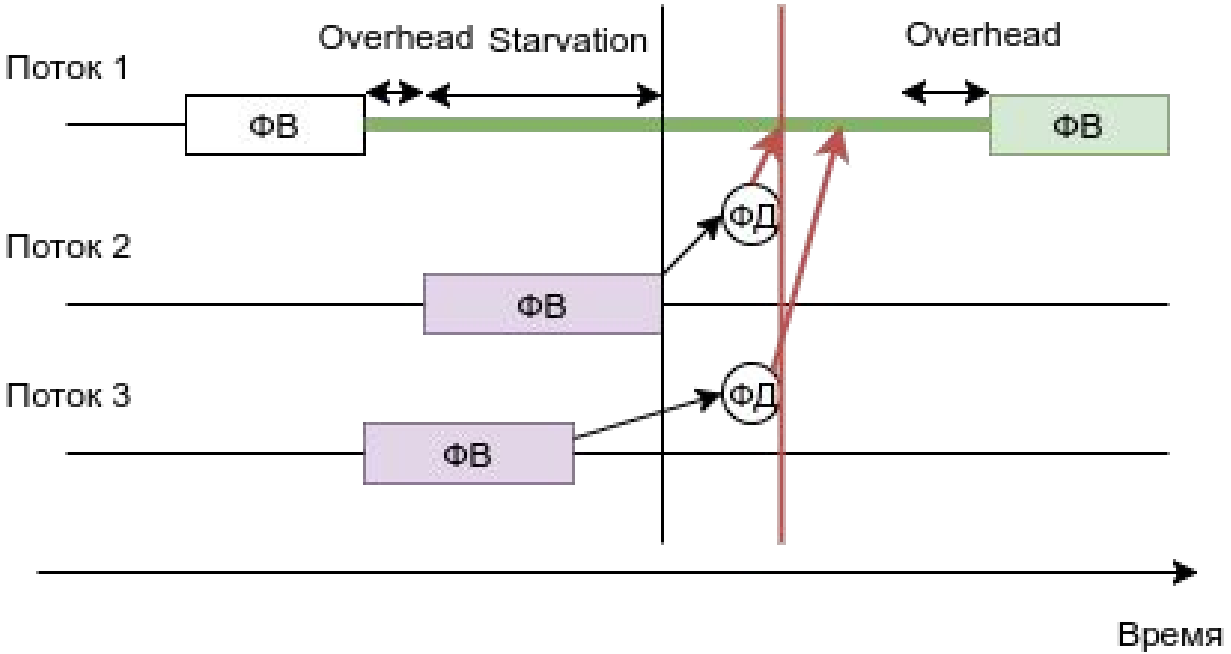


Схема анализа одного интервала простоя

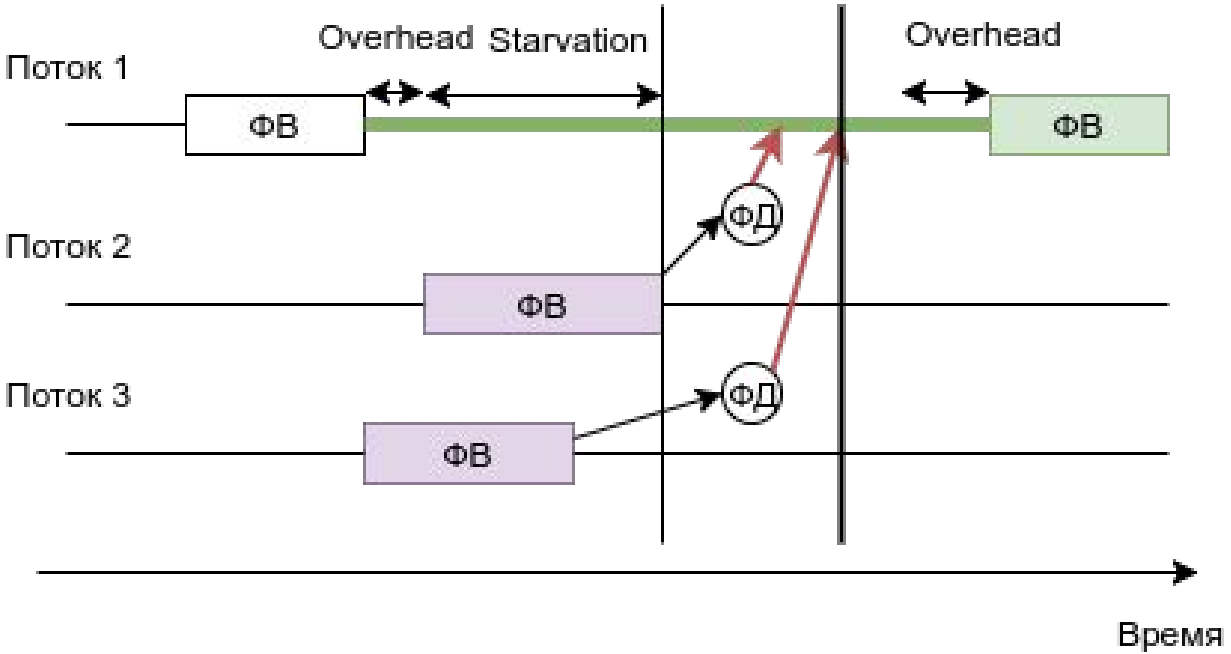


Схема анализа одного интервала простоя

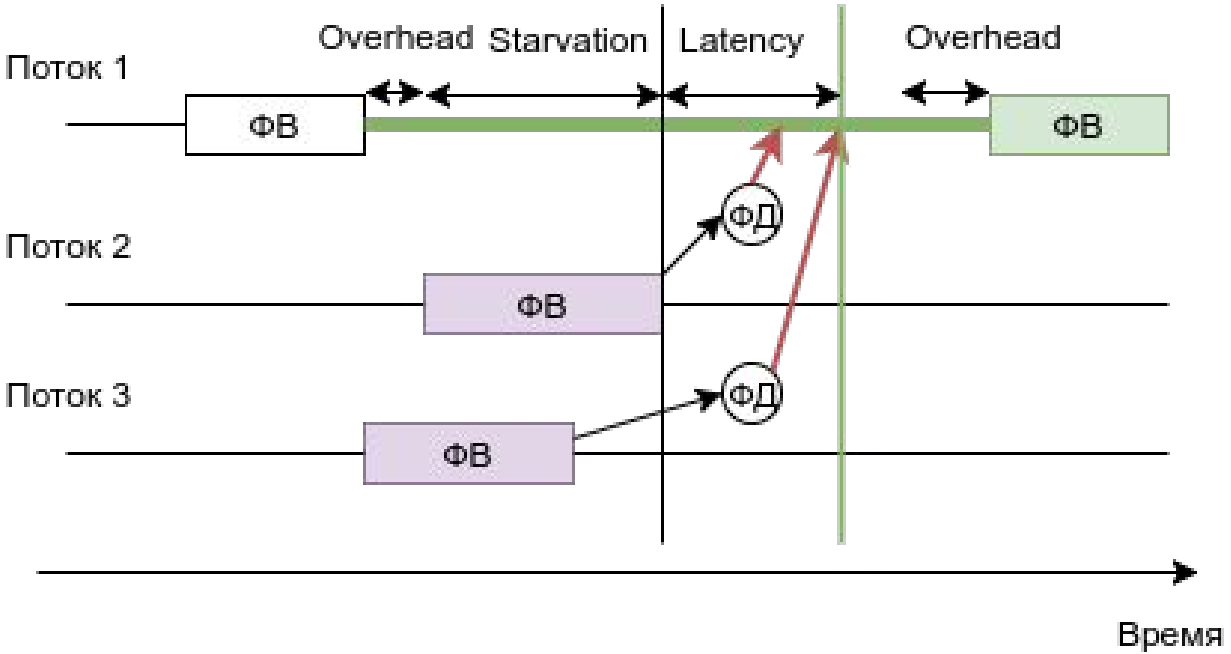


Схема анализа одного интервала простоя

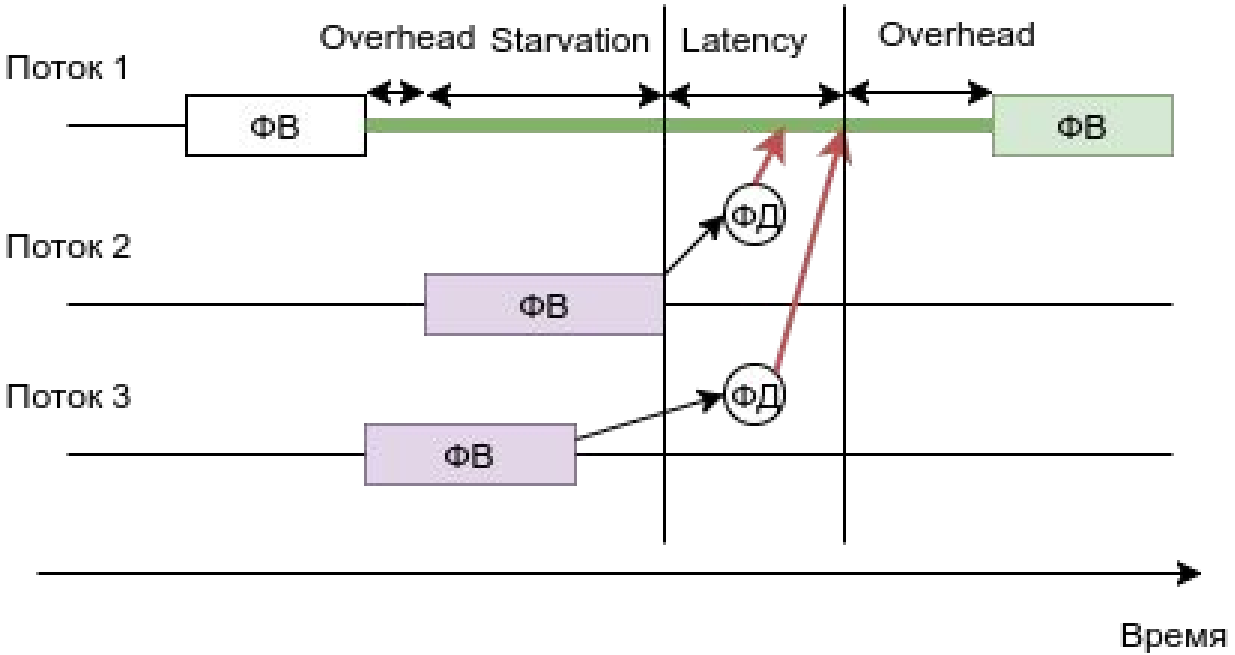
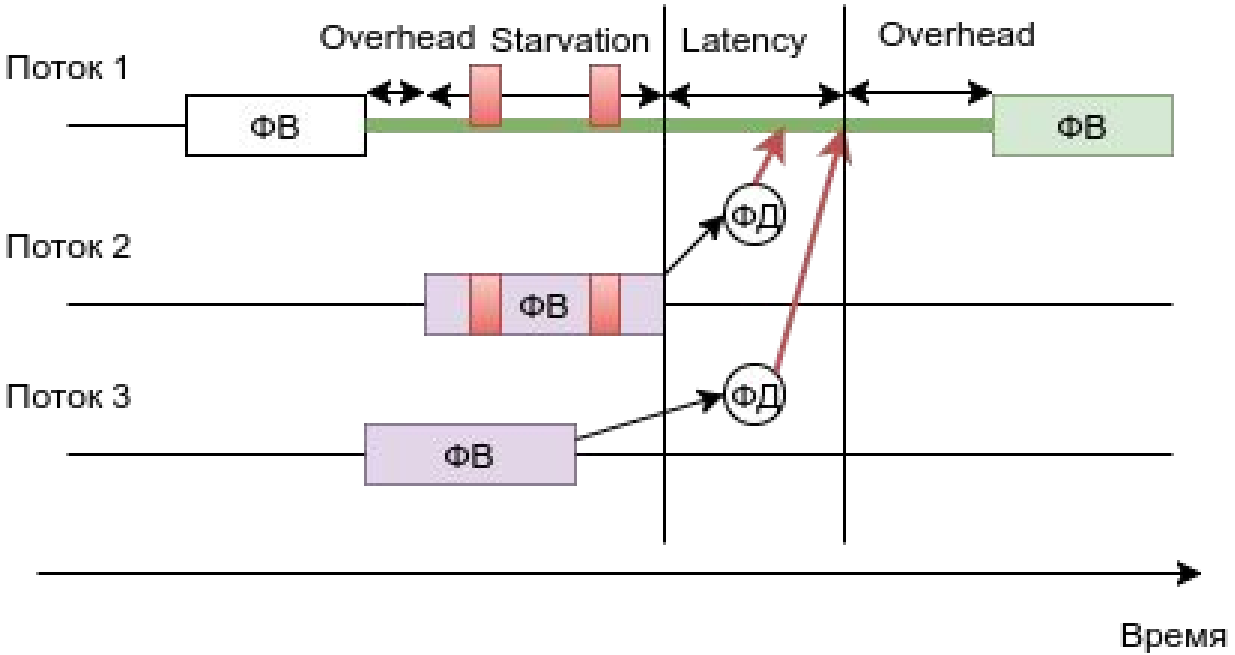


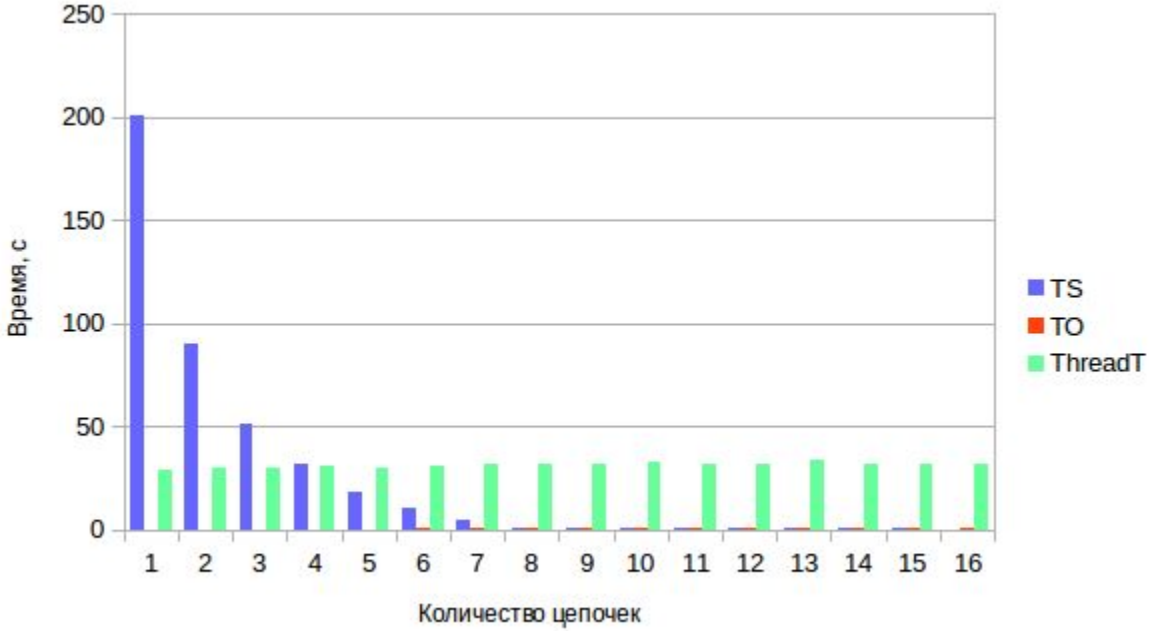
Схема анализа одного интервала простоя



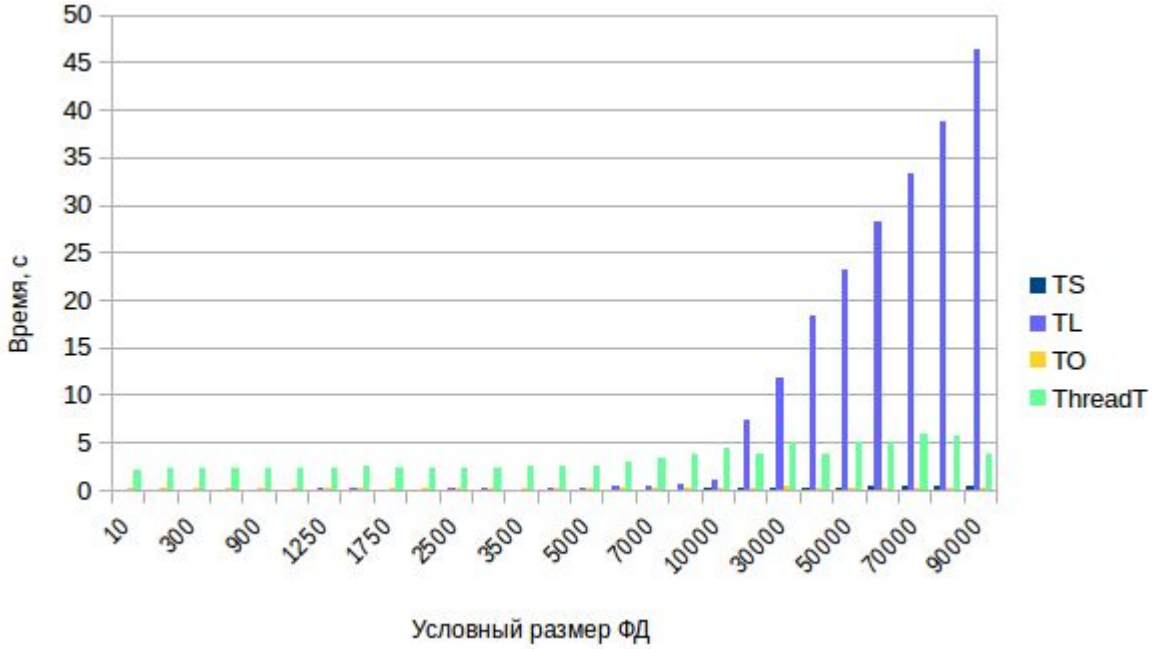
Реализация анализатора

- Спроектирован и реализован модуль для системы исполнения LuNA, позволяющий собирать необходимые для анализа задержек данные в файлы. Предусмотрена возможность анализировать некоторые задержки “на лету” и записывать в файлы результаты.
- Разработана утилита для расчета информации о задержках из этих файлов.
- Использованные языки и технологии: C++, SQLite

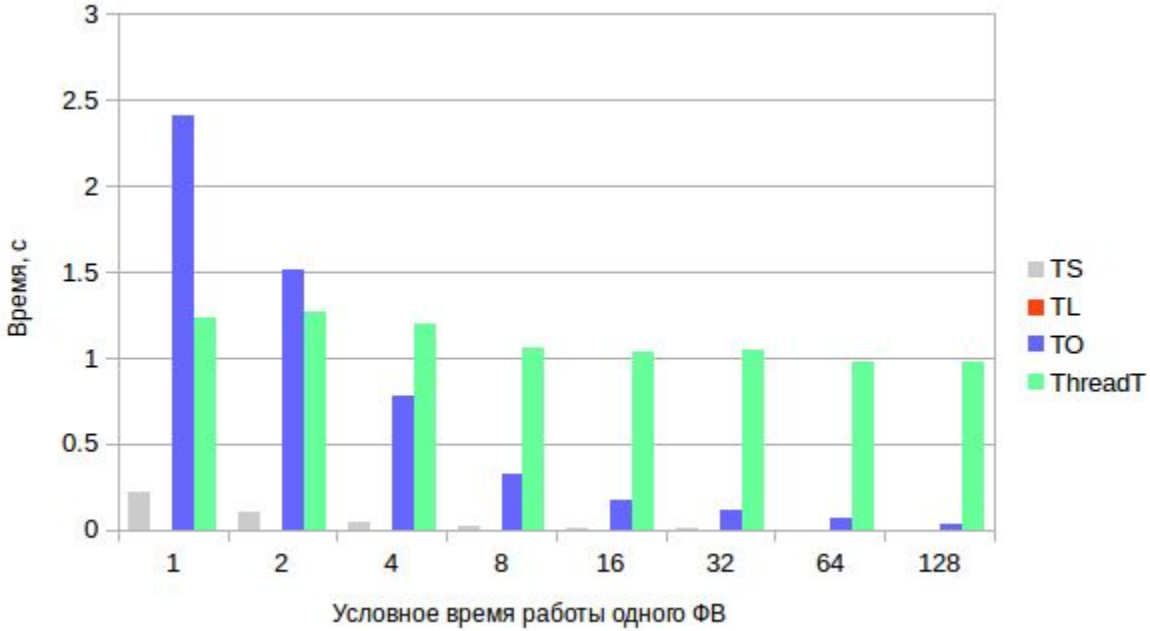
Синтетические тесты (Starvation)



Синтетические тесты (Latency)



Синтетические тесты (Overhead)



Анализ LuNA-реализации метода частиц-в-ячейках

Параметрами задачи являются:

- размеры пространственной сетки: *NX* × *NY* × *NZ*,
- количество модельных частиц: *NP*,
- число шагов по времени: *NT*.

Параметрами реализации являются:

- степень фрагментации пространства моделирования - количество подобластей, на которые оно разбивается: *FX* × *FY* × *FZ*,
- ресурсы, использованные для расчета - количество узлов кластера, ядер и рабочих потоков в одном узле.

NX = *NY* = *NZ* = 100, *NP* = 1 000 000, NT = 10, *FX* = *FY* = *FZ* = 4, на 1 узле с 16 рабочими потоками	● *TS* = 30,1 с ● *TL* = 0 с ● *TO* = 110,6 с (~69 % от WallT) ● *ThreadT* = 13,1 с ● *WallT* = 159,8 с
NX = *NY* = *NZ* = 100, *NP* = 1 000 000, *NT* = 10, *FX* = *FY* = *FZ* = 4, на 4 узлах с 16 рабочими потоками на каждом	● *TS* = 7669 с (~73% от WallT) ● *TL* = 1556 с ● *TO* = 1234 с ● *ThreadT* =12 с ● *WallT* = 10499 с

Результаты

- Сформулированы требования к профилировщику фрагментированных программ
- Выбраны характеристики исполнения, удовлетворяющие требованиям
- Разработан алгоритм получения выбранных характеристик по заданному исполнению, для чего была построена модель исполнения фрагментированной программы
- Алгоритм реализован для системы фрагментированного программирования LuNA
- Тесты показали адекватность подхода

Спасибо за внимание!

Направления дальнейшей работы

- Анализ частей LuNA-программ.
- Увеличение точности вычисления характеристик.
- Поиск других показательных характеристик.

Оценка применимости характеристик SLO

- Преобладает *TS*: низкая степень параллелизма в алгоритме, крупная фрагментация алгоритма, плохое распределение ФВ по процессам.
- Преобладает *TL*: медленная коммуникационная среда, плохое распределение ФД по процессам, из-за которого нарушается принцип локальности данных и ФД доходят до получателя не по оптимальному пути.
- Преобладает *TO*: мелкая фрагментация алгоритма, неэффективная работа системы исполнения.

Расчет характеристик по модели исполнения

- Анализируется каждый интервал простоя рабочего потока в порядке времени завершения простоя
- Затем результаты анализа суммируются для простоев в заданном временном промежутке для получения интегральных характеристик

Анализ одного интервала простоя Delay

- Каждый интервал простоя заканчивается началом исполнения какого-то ФВ *m*
- В $Starvation_1$ — время от начала простоя до завершения вычисления последнего входного ФД *n* на рабочем потоке *wt(n)*
- В Latency — время на передачу входных для *m* ФД по сети сверх времени, ушедшего на $Starvation_1$
- Из $Starvation_1$ выделяется $Overhead^{concrete}$, $Overhead^{uniform}$ для всех ФД, исполнявшихся в wt(n) в течение $Starvation_1$
  - $Overhead^{concrete}$ — известны границы интервалов, в течение которых работала runtime-система
  - $Overhead^{uniform}$ — примерно известно суммарное время работы runtime-системы
- $\Sigma|Starvation_1 \setminus Overhead^{concrete}| - Overhead^{uniform} = Starvation(m)$
- $\Sigma|Overhead^{concrete}| + Overhead^{uniform} + \Sigma|Delay \setminus Starvation_1 \setminus Latency(m)| = Overhead(m)$

Получение характеристик

Для получения выбранных характеристик было сделано следующее:

- Построена упрощенная модель вычислителя,
- Построена упрощенная модель исполнения ФА на вычислителе в виде конечного автомата,
- Построен алгоритм расчета выбранных характеристик исполнения ФА на основе модели ФА и модели исполнения ФА