Санкт-Петербургский государственный университет

Кафедра системного программирования

Реализация операций линейной алгебры на графическом процессоре с использованием F#

Артем Александрович Черников, группа 18.Б10-мм

16 ноября 2021 г.

Научный руководитель: к.ф.-м.н. С.В. Григорьев, доцент кафедры информатики

Санкт-Петербург
2020

Введение

- Граф может быть представлен в виде матрицы
- Существует двойственность между алгоритмами на графах и разреженной линейной алгеброй
- Стандарт GraphBLAS определяет операции над разреженными матрицами и векторами над расширенной алгеброй полуколец

Существующие решения

Критерии сравнения:

- Переносимость
- Гибкость с точки зрения создания пользователем собственных типов и операций над ними

Название библиотеки	Переносимость
SuiteSparse	Только CPU, в разработке GPU
GraphBLAST	Только GPU, поддержка CUDA
GBTL	Только CPU, в разработке GPU
pggraphblas	Только CPU

Таблица: Сравнение существующих решений по переносимости

Постановка задачи

Целью работы является реализация на языке F# GraphBLAS API для матриц, представленных в координатном формате, и разреженных векторов с поддержкой OpenCL

Задачи:

- Разработать и реализовать архитектуру некоторой части описанных в стандарте GraphBLAS примитивов линейной алгебры
- Реализовать набор базовых операций линейной алгебры над матрицами, представленными в координатном формате, и разреженными векторами с поддержкой OpenCL
- Реализовать некоторые классические задачи анализа графов
- Произвести сравнение с существующими аналогами

Используемые инструменты

- Язык F#
  - Предоставляет выразительную систему типов
  - Позволяет использовать созданную библиотеку на платформе .NET
- Библиотека Brahma.FSharp
  - Транслирует код на языке F# в OpenCL
  - Аналоги — FSCL, Alea.cuBase

Описание решения

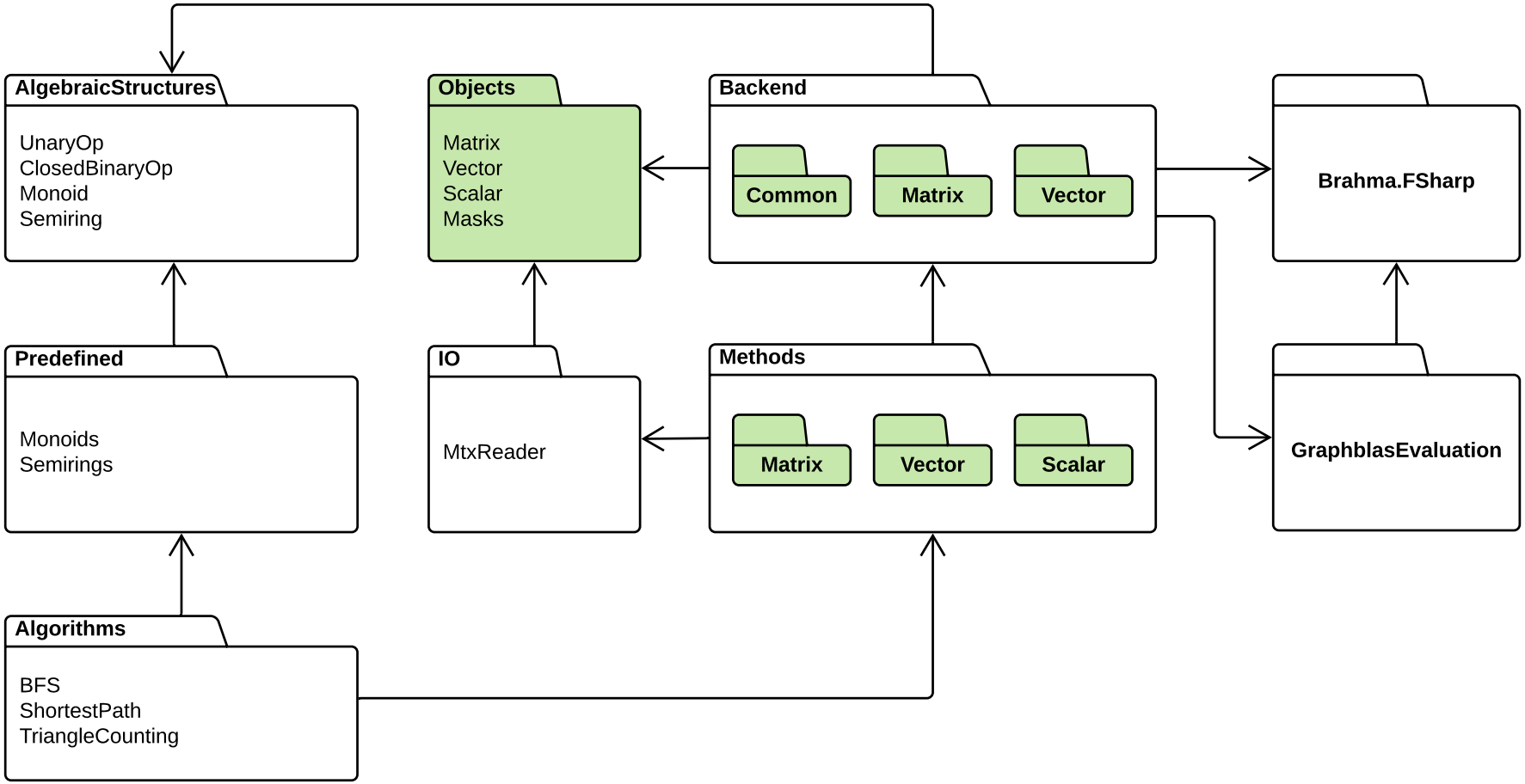

Рис.: Общая архитектура решения

Описание решения

Реализованные операции:

- Поэлементное сложение матриц, векторов
  - Слить массивы индексов в один с сохранением упорядоченности элементов
  - Отметить повторяющиеся индексы в получившемся массиве
  - Посчитать итоговую позицию каждого неповторяющегося индекса в получившемся массиве
  - Перенести неповторяющиеся данные в результирующий массив индексов
- Сокращение вектора при помощи аддитивной бинарной операции
- Запись значений в вектор через маску
  - Профильтровать аргумент через маску
  - Слить полученный вектор с вектором-целью

Сравнение с аналогом

Название матрицы	Размер	Количество ненулевых элементов
luxembourg_osm	114599	119666
belgium_osm	1441295	1549970
wiki-Talk	2394385	5021410
cit-Patents	3774768	16518948

Таблица: Матрицы, на которых производилось сравнение

Название матрицы	Среднее время выполнения, мс	
	SuiteSparse	GraphBLAS-sharp
luxembourg_osm	9.81	37.2
belgium_osm	46.59	80.98
wiki-Talk	53.51	125.85
cit-Patents	263.47	504.51

Таблица: Результаты замеров сложения для библиотек GraphBLAS-sharp и SuiteSparse

Сравнение с аналогом

Название матрицы	Размер	Количество ненулевых элементов
luxembourg_osm	114599	119666
belgium_osm	1441295	1549970
wiki-Talk	2394385	5021410
cit-Patents	3774768	16518948

Таблица: Матрицы, на которых производилось сравнение

Название матрицы	Среднее время копирования данных, мс
luxembourg_osm	6.57
belgium_osm	36.04
wiki-Talk	85.52
cit-Patents	353.38

Таблица: Результаты замеров времени копирования вспомогательных данных

Выводы

- Требуется снизить накладные расходы путём модифицирования библиотеки Brahma.FSharp
- С учётом переносимости и высокоуровневого интерфейса предлагаемое решение жизнеспособно

Результаты

- Разработана и реализована архитектура одномерной и двумерной маски, скаляра, а также разреженного вектора и матрицы, представленной в координатном формате
- Реализованы операции линейной алгебры с поддержкой OpenCL:
  - Поэлементное сложение разреженных векторов и матриц, представленных в координатном формате
  - Сокращение разреженного вектора при помощи аддитивной операции
  - Запись в разреженный вектор через маску
- Произведено сравнение с аналогом на примере поэлементного сложения