УНИВЕРСИТЕТ ИТМО

MASTER OF SOFTWARE ENGINEERING

Реализация операций с разреженными булевыми матрицами на OpenCL

Студент: Карпенко Мария Владимировна
Научный руководитель: к. ф.-м. н. Москвин Денис Николаевич
Научный консультант: к. ф.-м. н. Григорьев Семён Вячеславович

Санкт - Петербург
ITMO / JetBrains
2021 г.

Графовые алгоритмы в терминах линейной алгебры

Базовые задачи, сводимые к матричным операциям:

- транзитивное замыкание
- поиск в ширину
- прямое произведение графов

Матричные алгоритмы CFPQ[1] опираются на операции с разреженными булевыми матрицами[2,3]

Преимущества специализации операций для булевых матриц:

✓ Булевы матрицы занимают меньше памяти в сжатых форматах

✓ Операции с элементами булевых матриц осуществляются быстрее:

- Определение ‘+’ : операция $\vee$
- Определение ‘*’ : операция $\wedge$

---

[1]Context-Free Path Queries — поиск путей в графе, описываемых контекстно-свободной грамматикой.
[2]Azimov R., Grigorev S. Context-free path querying by matrix multiplication /GRADES and NDA. – 2018.
[3]E. Shemetova. One Algorithm to Evaluate Them All: Unified Linear Algebra Based Approach to Evaluate Both Regular and Context-Free Path Queries // arXiv preprint arXiv:2103.14688. — 2021.

Выбор стандарта и разреженных форматов

- **CUDA** — платформа для параллельных вычислений и разработки приложений с использованием видеокарт от NVIDIA
- **OpenCL** — открытый стандарт кроссплатформенной разработки программ для разного рода ускорителей:
  - разные производители видеокарт: AMD, NVIDIA
  - другие устройства: FPGA

Универсальные матричные форматы

COO — coordinate format
CSR — compressed sparse row
DCSR — doubly compressed sparse row

COO, DCSR:

+ Хранят информацию только о ненулевых элементах
- Долгая индексация строк

Формат	Память	Доступ к ряду
COO	$2\ nnz = O(nnz)$	$O(\log nnz)$
CSR	$n + nnz = O(n, nnz)$	$O(1)$
DCSR	$2 \times nzr + nnz = O(nnz)$	$O(\log nzr)$

n - число строк матрицы
nnz - количество единиц
nzr - число непустых рядов

Обзор библиотек линейной алгебры

Необходимые операции для задач CFPQ:

(1) матричное сложение
(2) матричное умножение
(3) произведение Кронекера
(4) транспонирование
(5) извлечение подматрицы
(6) редуцирование строк матрицы

Библиотека	Технология	Реализации для булевых матриц	Операции
cuBool[4]	CUDA	+	1—6
SuiteSparse[5]	CPU, OpenMP	+	1—6
cuSPARSE[6]	CUDA	-	1, 2, 4
CUSP[7]	CUDA	-	1, 2, 4
clSPARSE[8]	OpenCL	-	2

[4] https://github.com/JetBrains-Research/cuBool
[5] https://people.engr.tamu.edu/davis/suitesparse.html
[6] http://clmathlibraries.github.io/clSPARSE/
[7] https://docs.nvidia.com/cuda/cusparse/index.html
[8] https://cusplibrary.github.io/

Цель и задачи

Цель — реализация OpenCL библиотеки с операциями для разреженных булевых матриц, необходимыми для реализации матричных алгоритмов CFPQ

Задачи:

- Реализовать операции с разреженными матрицами: матричное умножение, матричное сложение, транспонирование, извлечение подматрицы, редуцирование строк матрицы, произведение Кронекера
- Оформить результаты в библиотеку операций с разреженными булевыми матрицами
- Провести экспериментальное исследование библиотеки: оценить производительность на различных устройствах (NVIDIA, AMD, FPGA Arria 10) и сравнить реализации с существующими библиотеками линейной алгебры

Реализация операций: форматы

COO:

- транспонирование
- сложение
- произведение Кронекера

DCSR:

- умножение
- взятие подматрицы
- редуцирование строк матрицы
- произведение Кронекера

Промежуточные операции:

- сортировка
- префиксная сумма
- слияние

Реализация операций: выбор алгоритма умножения

Алгоритм W. Liu[9]

- \+ отсутствует сортировка строк с большим числом элементов
- \- требуется промежуточная матрица
- \- дополнительная глобальная память для больших рядов

Алгоритм Y. Nagasaka[10]

- \+ не требует промежуточной матрицы
- \+ единый подход к обработке всех строк
- \- требуется двойной проход для предварительного вычисления размера матриц

Вывод:

Алгоритм Y. Nagasaka выбран в качестве основного, так как он:

- примерно в 2 раза быстрее по сравнению с алгоритмом W. Liu на матрицах без плотных строк
- менее требователен по памяти на более плотных матрицах

[9]Liu W. Vinter B. An efficient GPU general sparse matrix-matrix multiplication for irregular data // 2014 IEEE

[10]Nagasaka Y. et al . High-performance and memory-saving sparse general matrix-matrix multiplication for nvidia pascal gpu // 2017 ICPP

Реализация операций: произведение Кронекера

Разные алгоритмы для форматов:

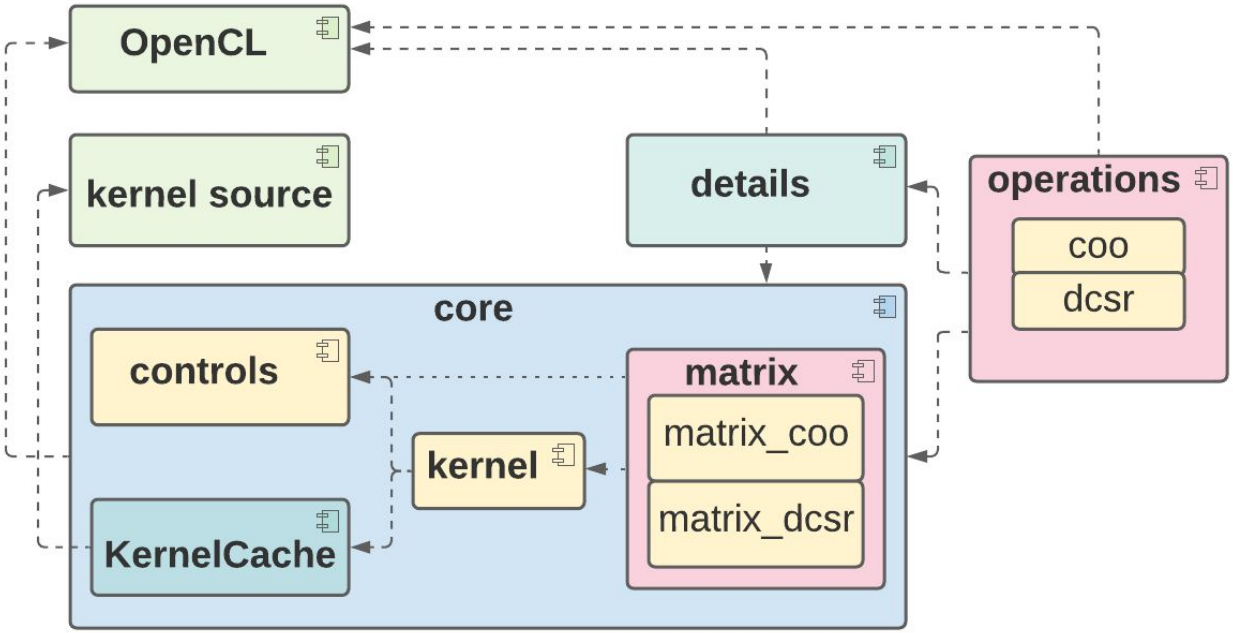
- DCSR:
  - постоянная индексация строк (*log nzr, log nzc*)
  - префиксная сумма
- COO:
  - сортировка результата

Вывод:
Реализация произведения Кронекера на основе формата DCSR оказалась эффективнее.

№	nnz(M), млн	nnz(M ⊗ M)[11] млн	DCSR, мс	COO, мс
1	0,001	1	0,77 ± 0,03	3,15 ± 0,02
2	0,002	4	2,50 ± 0,09	14,56 ± 0,72
3	0,003	9	5,23 ± 0,04	52,25 ± 0,95
4	0,004	16	9,00 ± 0,11	66,06 ± 0,49
5	0,005	25	13,37 ± 0,11	128,76 ± 0,55
6	0,006	36	19,06 ± 0,34	246,19 ± 1,09
7	0,007	49	25,19 ± 0,34	278,84 ± 0,50
8	0,008	64	31,97 ± 0,38	319,79 ± 1,22
9	0,009	81	36,32 ± 0,05	562,48 ± 0,52
10	0,01	100	45,38 ± 0,28	615,95 ± 0,58

[11]⊗ — операция произведения Кронекера

Библиотека clBool



Controls — состояние библиотеки и выбор устройства

kernel — создание и вызов GPU-ядер

kernel source — бинарные файлы с исходным кодом ядер

KernelCache — скомпилированные ядра

Исполнение кода на NVIDIA, AMD, FPGA: устройства

Характеристики устройств, участвующих в экспериментах

Вендор	NVIDIA	AMD	Intel FPGA
Имя	GeForce GTX 1070	Radeon Vega Frontier Edition	10AX115S2F45E2LG
Глобальная память, gb	7,926	15,98	8
Локальная память, kb	48	64	16
Макс. размер блока	1024	256	2147483647
Частота, MHz	1746	1600	1000
Число АЛУ	1920	4096	427200

Исполнение кода на NVIDIA, AMD, FPGA: сложение

$M + M^2$, мс

№	Число строк, млн	nnz(M), млн	nnz(M + M^2), млн	NVIDIA	AMD	Intel FPGA
1	0,06	0,24	0,92	1,73 ± 3,97	1,71 ± 0,30	86,51
2	0,11	0,24	0,39	1,21 ± 0,53	1,32 ± 0,12	58,54
3	0,40	3,20	**14,97**	19,84 ± 0,10	11,43 ± 3,18	1572,83
4	0,74	5,16	**26,40**	35,24 ± 3,28	17,19 ± 4,59	2714,14
5	0,92	5,11	**30,81**	38,10 ± 0,53	19,34 ± 4,68	3077,81
6	1,09	3,08	9,93	13,24 ± 0,70	7,92 ± 2,58	944,677
7	1,39	3,84	12,26	15,48 ± 0,52	9,01 ± 2,68	1167,12
8	1,44	3,10	8,41	9,86 ± 0,42	6,78 ± 2,59	771,219
9	1,97	5,53	17,74	21,01 ± 0,45	11,89 ± 3,23	1686,66
10	2,22	4,88	13,63	15,51 ± 0,48	9,27 ± 2,81	1248,45

Исполнение кода на NVIDIA, AMD, FPGA: умножение

M^2, мс

№	Число строк, млн	nnz, млн	nnz(M^2), млн	NVIDIA (1)	NVIDIA (2)	AMD (2)	Intel FPGA (2)
1	0,06	0,24	0,71	2,71 ± 0,11	1,88 ± 0,34	2,78 ± 1,46	5590,79
2	0,11	0,24	0,39	4,33 ± 0,11	2,06 ± 0,36	2,43 ± 0,25	9824,65
3	0,40	3,20	**14,39**	63,81 ± 3,92	52,02 ± 0,04	38,60 ± 3,92	40527,5
4	0,74	5,16	**25,37**	93,24 ± 0,39	81,43 ± 0,29	55,65 ± 6,07	67595,9
5	0,92	5,11	**29,71**	184,06 ± 0,53	126,36 ± 0,34	85,97 ± 8,78	77601,9
6	1,09	3,08	7,24	31,46 ± 0,22	14,02 ± 0,15	19,75 ± 1,27	96978,8
7	1,39	3,84	8,90	38,72 ± 0,17	16,64 ± 0,12	23,29 ± 1,68	122992
8	1,44	3,10	5,32	37,70 ± 0,88	16,62 ± 0,23	23,08 ± 1,84	126465
9	1,97	5,53	12,91	54,06 ± 0,43	23,10 ± 0,06	29,35 ± 1,65	175612
10	2,22	4,88	8,76	57,27 ± 1,71	24,57 ± 0,11	29,66 ± 0,85	-

(1) — алгоритм W. Liu
(2) — алгоритм Y. Nagasaka

Сравнение с библиотеками: сложение

$M + M^2$, мс

№	nnz(M + M^2), млн	cuBool	clBool	Cusp	cuSPARSE	SuiteSparse
1	0,92	1,12 ± 0,02	1,73 ± 3,97	1,54 ± 0,20	2,40 ± 0,04	3,91 ± 2,06
2	0,39	1,84 ± 0,10	1,21 ± 0,53	1,11 ± 0,21	0,86 ± 0,04	1,86 ± 0,29
3	**14,97**	12,14 ± 0,46	19,84 ± 0,10	16,26 ± 0,40	23,74 ± 0,04	37,12 ± 0,13
4	**26,40**	20,06 ± 2,67	35,24 ± 3,28	29,86 ± 1,65	27,75 ± 1,65	65,02 ± 1,35
5	**30,81**	24,20 ± 0,90	38,10 ± 0,53	32,08 ± 1,34	88,48 ± 0,28	77,68 ± 1,66
6	9,93	16,69 ± 0,59	13,24 ± 0,70	11,07 ± 0,28	11,62 ± 0,03	36,50 ± 1,01
7	12,26	19,86 ± 0,65	15,48 ± 0,52	14,42 ± 0,98	17,43 ± 0,03	46,33 ± 3,07
8	8,41	19,55 ± 0,12	9,86 ± 0,42	9,83 ± 0,24	11,57 ± 0,99	28,44 ± 1,00
9	17,74	31,61 ± 1,01	21,01 ± 0,45	18,90 ± 0,40	20,50 ± 0,82	65,74 ± 1,39
10	13,63	30,65 ± 1,89	15,51 ± 0,48	14,75 ± 0,31	18,15 ± 0,03	50,29 ± 0,93

Сравнение с библиотеками: умножение

M^2, мс

№	nnz(M^2), млн	cuBool	clBool-hash	clSPARSE	cuSPARSE	Cusp	SuiteSparse
1	0,71	1,78 ± 0,06	1,88 ± 0,34	2,02 ± 0,34	2,04 ± 0,04	5,39 ± 0,12	7,92 ± 0,27
2	0,39	2,50 ± 0,04	2,06 ± 0,36	2,64 ± 0,34	1,72 ± 0,05	3,79 ± 0,07	3,72 ± 1,57
3	**14,39**	24,19 ± 0,82	52,02 ± 0,04	32,73 ± 1,57	408,71 ± 0,54	108,98 ± 0,45	258,04 ± 8,39
4	**25,37**	34,63 ± 1,53	81,43 ± 0,29	51,19 ± 2,01	184,64 ± 10,23	172,74 ± 0,36	378,94 ± 15,28
5	**29,71**	42,19 ± 1,26	126,36 ± 0,34	165,03 ± 8,30	4726,33 ± 0,52	246,232 ± 9,90	712,54 ± 20,07
6	7,24	18,26 ± 0,19	14,02 ± 0,15	23,03 ± 1,60	37,24 ± 0,13	42,38 ± 0,17	67,36 ± 1,57
7	8,90	22,73 ± 0,18	16,64 ± 0,12	28,43 ± 1,88	46,40 ± 0,15	51,77 ± 0,30	81,27 ± 1,80
8	5,32	23,37 ± 0,17	16,62 ± 0,23	25,31 ± 1,47	26,54 ± 0,07	33,09 ± 0,20	58,46 ± 1,76
9	12,91	32,09 ± 0,34	23,10 ± 0,06	39,26 ± 2,52	65,14 ± 4,47	76,72 ± 3,23	116,88 ± 3,99
10	8,76	35,30 ± 0,22	24,57 ± 0,11	38,51 ± 2,12	50,68 ± 0,14	51,28 ± 0,35	93,47 ± 2,09

Результаты

- Реализованы операции с разреженными булевыми матрицами:
  - ✓ матричное умножение
  - ✓ матричное сложение
  - ✓ транспонирование матрицы
  - ✓ извлечение подматрицы
  - ✓ редуцирование строк матрицы
  - ✓ произведение Кронекера
- Создана библиотека clBool, которая может быть использована в качестве бэкенда для алгоритмов анализа графов, опирающихся на эти операции, в том числе для алгоритмов CFPQ
  github: https://github.com/mkarpenkospb/clBool
- Экспериментальное исследование реализаций показало следующие результаты:
  - Операции могут быть исполнены на видеокартах AMD и NVIDIA
  - Реализация умножения для булевых матриц до 2-х раз быстрее по сравнению с другими библиотеками и требует в среднем на треть меньше памяти
- Результаты исследования были представлены на конференции GrAPL 2021: Workshop on Graphs, Architectures, Programming, and Learning

Характеристики матриц

№	Имя	Число строк, млн	Nnz, млн	Max nnz/row, млн	Nnz(M^2), млн	Nnz($M+M^2$), млн
1	wing	0,06	0,24	4	0,71	0,92
2	luxembourg_osm	0,11	0,24	6	0,39	0,39
3	amazon0312	0,40	3,20	10	14,39	14,97
4	amazon-2008	0,74	5,16	10	25,37	26,40
5	web-Google	0,92	5,11	456	29,71	30,81
6	roadNet-PA	1,09	3,08	9	7,24	9,93
7	roadNet-TX	1,39	3,84	12	8,90	12,26
8	belgium_osm	1,44	3,10	10	5,32	8,41
9	roadNet-CA	1,97	5,53	12	12,91	17,74
10	netherlands_osm	2,22	4,88	7	8,76	13,63

SuiteSparse Matrix Collection https://sparse.tamu.edu/

Сравнение с библиотеками: сложение

$M + M^2$, МБ

№	nnz(M + M^2), млн	cuBool	clBool	CUSP	cuSPRS	SuiteSparce
1	0,92	95	101	105	163	176
2	0,39	95	101	97	151	174
3	**14,97**	221	477	455	405	297
4	**26,40**	323	761	723	595	319
5	**30,81**	355	857	815	659	318
6	9,93	189	283	329	317	287
7	12,26	209	381	385	357	319
8	8,41	179	255	303	297	302
9	17,74	259	509	513	447	331
10	13,63	233	405	423	385	311

Сравнение с библиотеками: умножение

M^2, МБ

№	nnz(M^2), млн	cuBool	clBool-hash	clBool-merge	clSPRS	CUSP	cuSPRS	SuiteSparse
1	0,71	93	89	95	105	125	155	22
2	0,39	91	89	91	97	111	151	169
3	**14,39**	165	163	277	459	897	301	283
4	**25,37**	225	221	405	701	1409	407	319
5	**29,71**	241	239	491	1085	1717	439	318
6	7,24	157	153	199	283	481	247	294
7	8,90	167	165	229	329	581	271	328
8	5,32	151	159	181	259	397	235	302
9	12,91	199	211	287	433	771	325	344
10	8,76	191	189	261	361	585	291	311

AMD: два алгоритма умножения

M^2, мс

№	nnz(M^2), млн	Алгоритм W. Liu	Алгоритм Y. Nagasaka
1	0,71	4,93 ± 3,53	2,78 ± 1,46
2	0,39	5,24 ± 0,28	2,43 ± 0,25
3	**14,39**	38,65 ± 2,54	38,60 ± 3,92
4	**25,37**	55,91 ± 5,08	55,65 ± 6,07
5	**29,71**	94,01 ± 9,35	85,97 ± 8,78
6	7,24	38,72 ± 6,37	19,75 ± 1,27
7	8,90	44,08 ± 4,67	23,29 ± 1,68
8	5,32	37,50 ± 3,43	23,08 ± 1,84
9	12,91	51,94 ± 3,83	29,35 ± 1,65
10	8,76	52,90 ± 5,79	29,66 ± 0,85