Разработка эффективной структуры данных, поддерживающей инкрементальные изменения, для параллельной обработки деревьев

Автор: У Цзюньфэн, группа M4236
Научный руководитель: Станкевич А.С.
Консультант: Буздалов М.В

Кафедра компьютерных технологий
Факультет ИТиП

УНИВЕРСИТЕТ ИТМО

20 июня 2016 года

Мотивация: Структуры данных для динамических деревьев

- Структура данных поддерживает лес (укорененных) деревьев
  - на вершинах могут быть пометки
  - на ребрах могут быть пометки
- Операции модификации:
  - добавление вершины
  - подвешивание вершины в качестве ребенка другой вершины
  - отцепление вершины от ее родителя
- Запросы:
  - корень дерева, в котором находится данная вершина
  - правда ли, что две вершины в одной компоненте связности
  - сумма вершинных пометок в поддереве данной вершины
  - сумма реберных пометок на пути между двумя вершинами
- Зачем?
  - эффективные алгоритмы поиска максимального потока
  - ...

Цель и задачи

- Цель исследования
  - Разработка структуры данных на основе Rake-and-Compress деревьев для параллельной обработки деревьев, поддерживающей инкрементальные изменения: добавление нового узла, подвешивание узла в качестве ребенка другого узла, отцепление узла от его родителя, изменение меток на вершинах и ребрах.

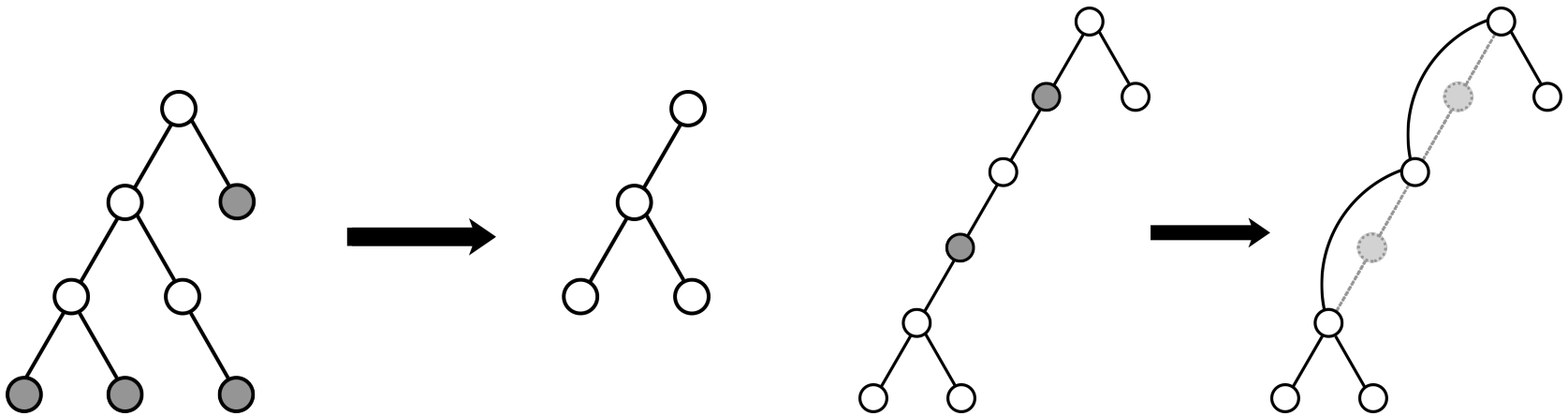
- Задачи
  1. снятие ограничения на число детей вершины, присущее параллельным реализациям Rake-and-Compress деревьев;
  2. уменьшение использования памяти с $\Theta(n \log n)$ до $\Theta(n)$, где n — общее число вершин в деревьях, при сохранении эффективности параллельных операций;
  3. обеспечение поддержки меток произвольных типов для вершин и ребер, при условии что метки на вершинах являются элементами коммутативного моноида, а метки на ребрах — элементами произвольного моноида.

Техническое задание и исходные данные к работе

- Требуется разработать структуру данных для параллельной обработки деревьев, поддерживающую инкрементальные изменения.
- В разработанной структуре данных требуется снять ограничение на максимальную степень вершины, имеющееся у существующих реализаций.
- Требуется поддержка эффективного выполнения запросов на определения корня дерева по представителю дерева, определение групповой суммы меток в поддереве данной вершины и на пути между двумя данными вершинами.
- Структуру данных требуется реализовать на языке C++ с использованием библиотеки PASL для реализации параллельных операций.

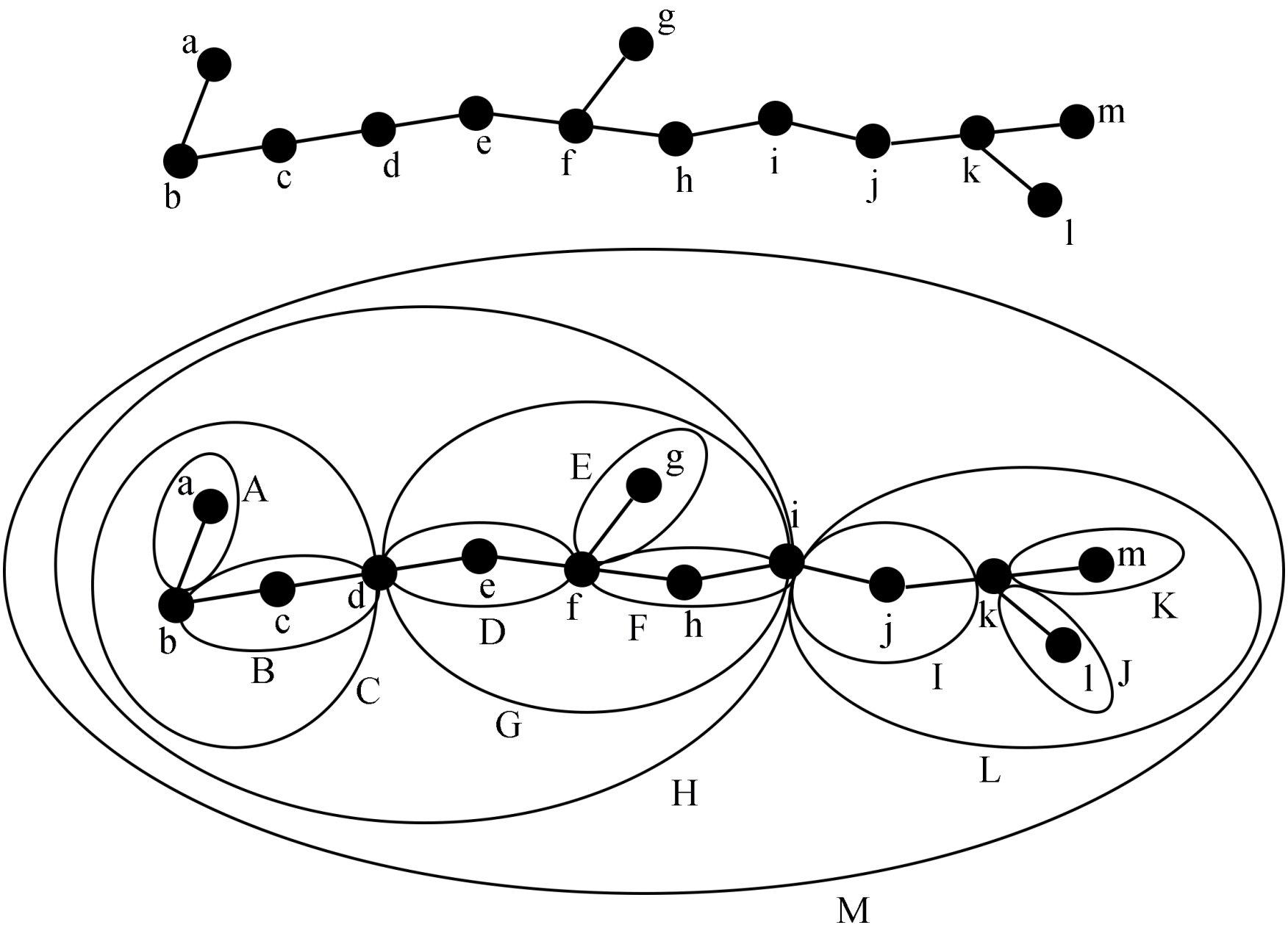
Rake-and-Compress Tree

- Исходное дерево подвергается одновременному применению операций Rake и Compress для всех возможных вершин



- Применяем эту процедуру к дереву, пока оно не сожмется в одну вершину
- $O(\log n)$ уровней, каждый уровень — различная степень подробности представления дерева

Rake-and-Compress Tree: Общая картина



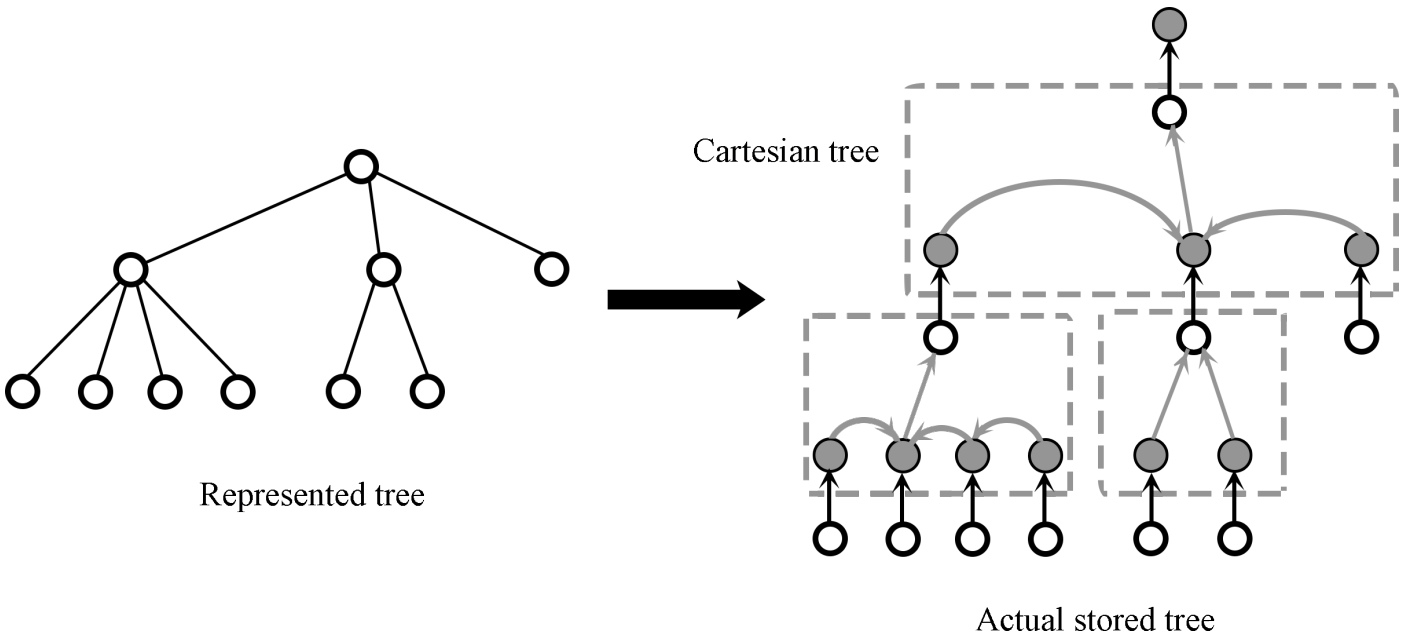
Преимущества и недостатки RC-деревьев

- Преимущества
  - Поддерживает запросы на поддеревьях и на путях
  - Может строиться параллельно
  - Поддерживает произвольные модификации
  - Может применять пакеты обновлений параллельно

- Недостатки
  - Эффективно работает, только если число детей вершины ограничено константой
  - Сложная реализация с разбором большого числа случаев
  - Параллелизм $\rightarrow$ проблемы с памятью $\rightarrow \Theta(n \log n)$ памяти в простой реализации
  - Переиспользуемых реализаций пока нет

Предлагаемая реализация (1/2)

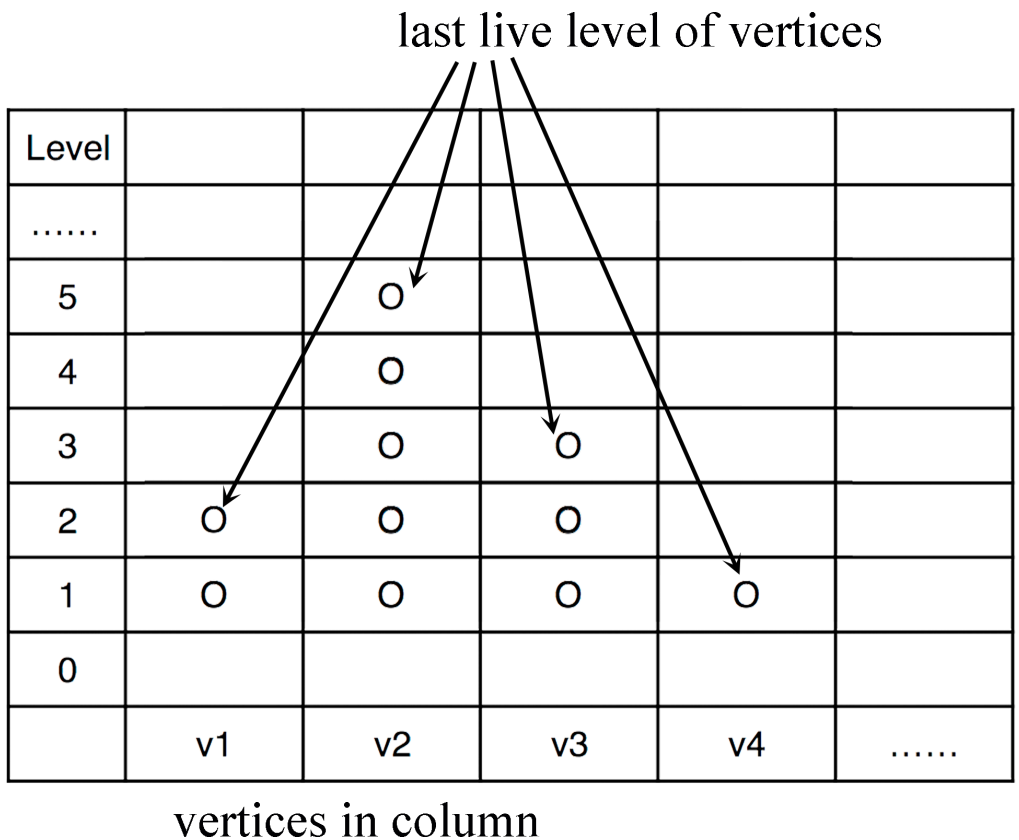
- Снятие ограничений на степень вершины — разделение каждой вершины на вершину данных и вершину связей, организация детей в декартово дерево



- Пометки на вершинах — элементы коммутативного моноида, на ребрах — элементы произвольного моноида
  - Вершины связей имеют нулевые пометки на вершинах и ребрах — результаты запросов не зависят от структуры

Предлагаемая реализация (2/2)

- Решение проблем с памятью — вектор «живых» инкарнаций вершин для каждой вершины
- Для четных и нечетных слоев — разные векторы для разделения чтения слоя i и записи слоя $i + 1$



Распараллеливание

- В алгоритме используется две распараллеливаемые операции
  - Цикл FOR
  - Вычисление префиксных сумм:

    даны числа $x_1, x_2, ..., x_n$

    вычислить $s_1 = x_1, s_2 = x_1 + x_2, ..., s_n = x_1 + x_2 + ... + x_n$
- Реализации операций
  - Последовательная (Sequential)
  - Параллельная с использованием OpenMP
  - Параллельная с использованием PASL

Результаты эксперимента

- Тесты:
  1. Дерево-«палка»: n вершин, при этом i-тая вершина является ребенком $(i-1)$-ой вершины при $i > 0$
  2. Дерево-«пучок»: n вершин, при этом i-тая вершина является ребенком вершины 0 при $i > 0$
  3. Дерево-«два пучка»: n вершин при $n = 2m$, m целое, при этом вершина 0 является родителем вершин с 1 по $(m-1)$, вершина m является родителем вершин с $(m+1)$ по $(n-1)$, а вершины 0 и m дополнительно соединены
  4. Построение дерева-«палки» в десять этапов, каждый из которых состоит в присоединении $n/10$ вершин
- Тестовый сервер: восемь ядер, Intel Xeon E5606, 2.13 ГГц
- Следующие слайды: время работы на тестах в секундах

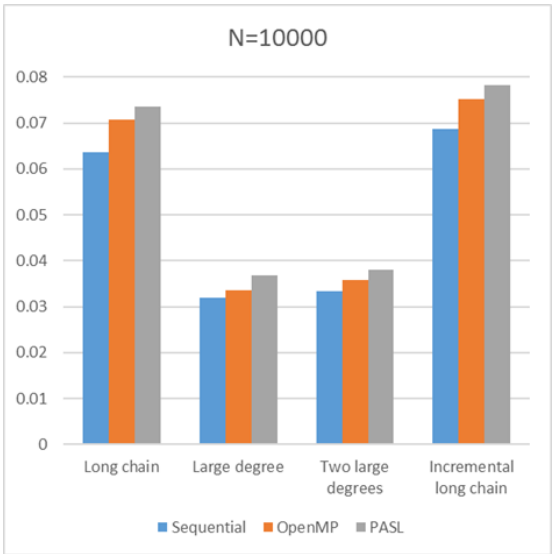
Результаты экспериментов: одно ядро

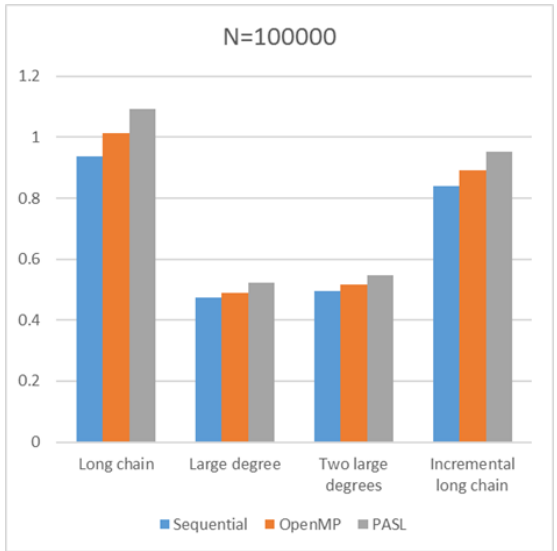
Sequential	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.063533	0.936087	9.8946	31.3906
2 large degree	0.031963	0.473844	5.04014	16.8301
3 two large degrees	0.03333	0.495401	5.19158	16.8656
4 incremental long chain	0.068615	0.840358	11.1828	36.3289

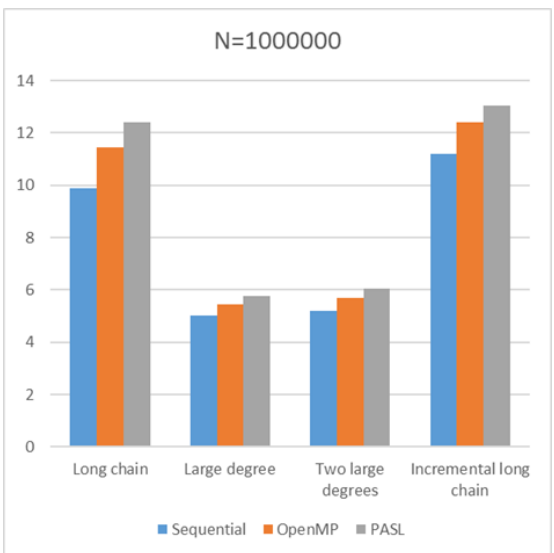
OpenMP	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.07072	1.01285	11.4329	35.1043
2 large degree	0.033578	0.48801	5.46487	17.5146
3 two large degrees	0.035731	0.515872	5.68942	18.0354
4 incremental long chain	0.075208	0.891592	12.389	40.6775

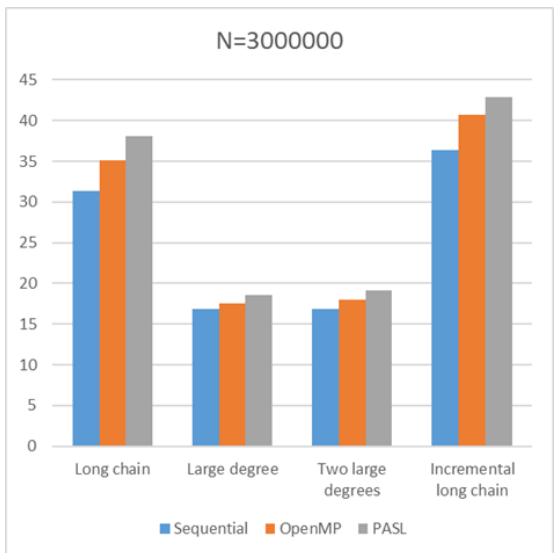
PASL	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.073684	1.09323	12.3956	38.1296
2 large degree	0.036832	0.522955	5.75152	18.5603
3 two large degrees	0.038104	0.548474	6.03994	19.1851
4 incremental long chain	0.078299	0.953348	13.0366	42.883

Результаты экспериментов: одно ядро









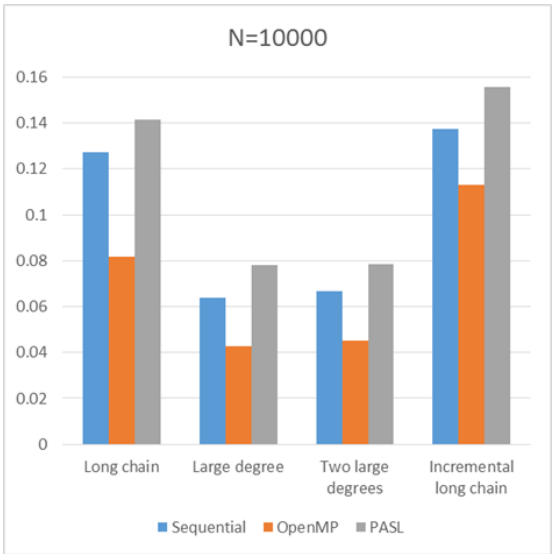
Результаты экспериментов: два ядра

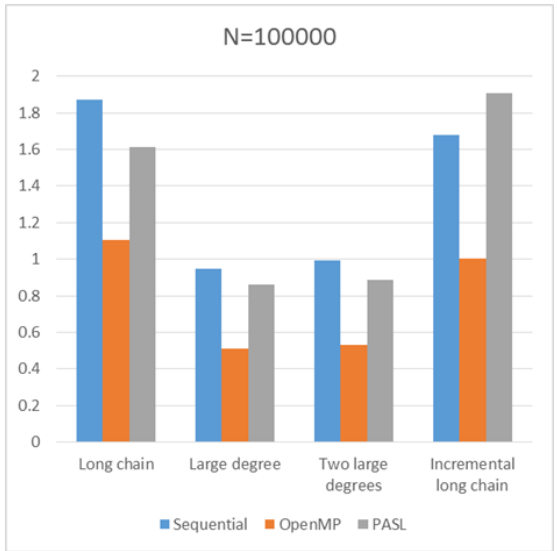
Sequential	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.063533	0.936087	9.8946	31.3906
2 large degree	0.031963	0.473844	5.04014	16.8301
3 two large degrees	0.03333	0.495401	5.19158	16.8656
4 incremental long chain	0.068615	0.840358	11.1828	36.3289

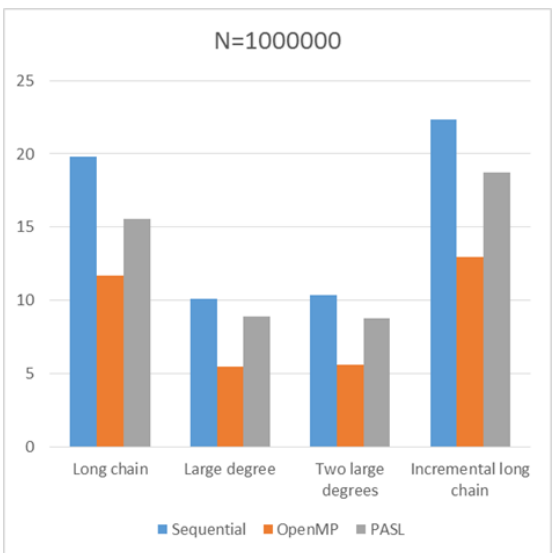
OpenMP	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.081845	1.10473	11.6696	35.7506
2 large degree	0.04275	0.510112	5.47687	18.0637
3 two large degrees	0.044991	0.529194	5.59677	17.8966
4 incremental long chain	0.11298	1.00143	12.9886	41.9446

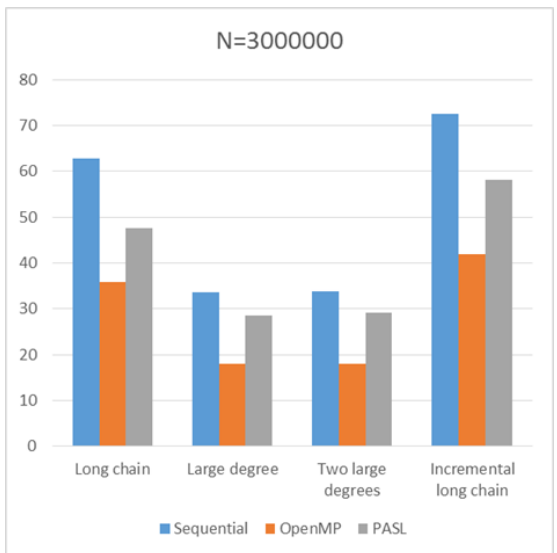
PASL	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.141362	1.61106	15.5824	47.6541
2 large degree	0.077958	0.859135	8.91695	28.6172
3 two large degrees	0.078529	0.88774	8.77827	29.134
4 incremental long chain	0.155524	1.90973	18.7348	58.2101

Результаты экспериментов: два ядра









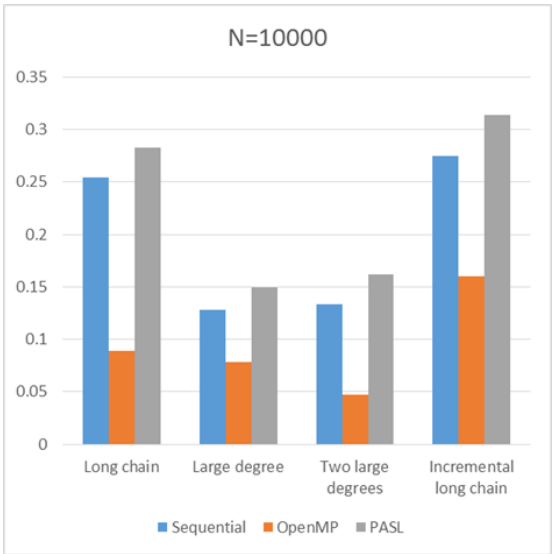
Результаты экспериментов: четыре ядра

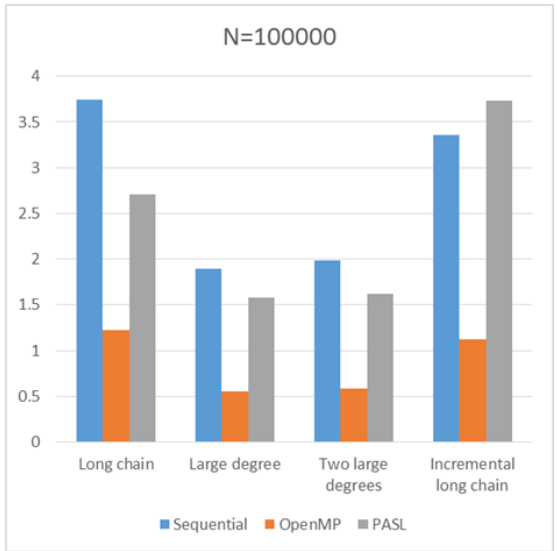
Sequential	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.063533	0.936087	9.8946	31.3906
2 large degree	0.031963	0.473844	5.04014	16.8301
3 two large degrees	0.03333	0.495401	5.19158	16.8656
4 incremental long chain	0.068615	0.840358	11.1828	36.3289

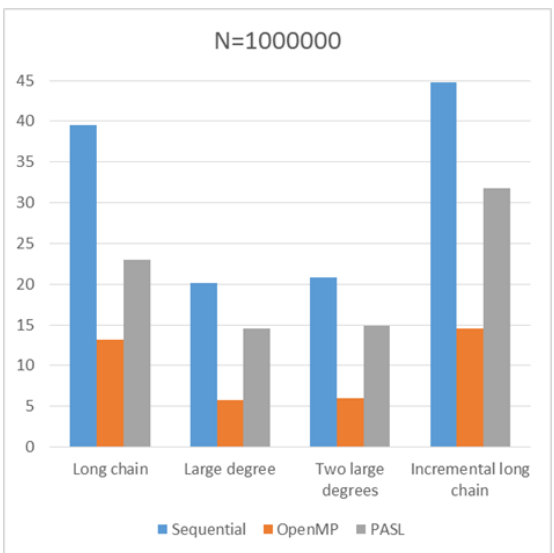
OpenMP	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.089218	1.22818	13.169	40.9343
2 large degree	0.07859	0.553611	5.78005	18.871
3 two large degrees	0.047683	0.583647	5.94402	18.8319
4 incremental long chain	0.159757	1.12068	14.4973	47.2622

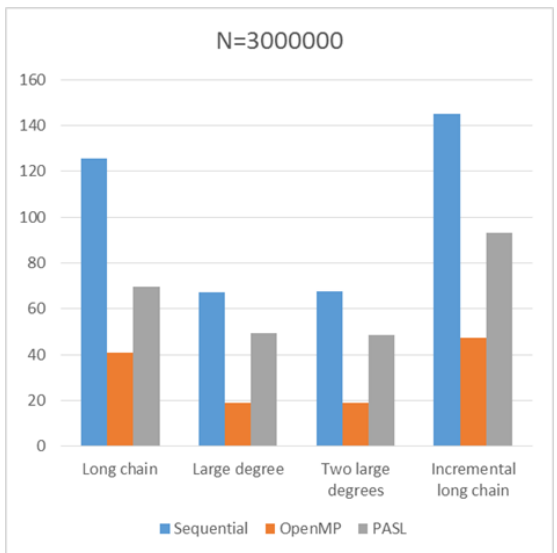
PASL	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.28281	2.71137	22.9835	69.6526
2 large degree	0.149832	1.5798	14.5509	49.3806
3 two large degrees	0.162006	1.61922	14.9153	48.653
4 incremental long chain	0.313789	3.73799	31.7739	93.2229

Результаты экспериментов: четыре ядра









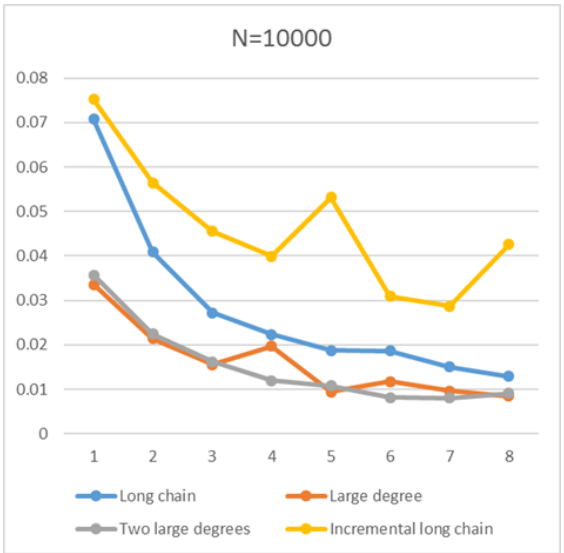
Результаты экспериментов: восемь ядер

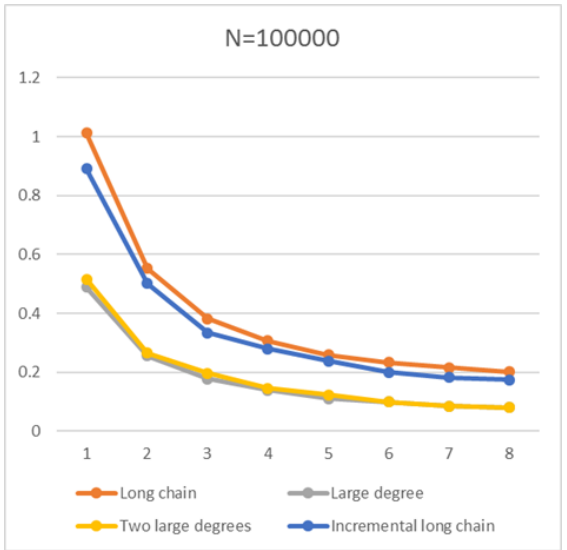
Sequential	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.063533	0.936087	9.8946	31.3906
2 large degree	0.031963	0.473844	5.04014	16.8301
3 two large degrees	0.03333	0.495401	5.19158	16.8656
4 incremental long chain	0.068615	0.840358	11.1828	36.3289

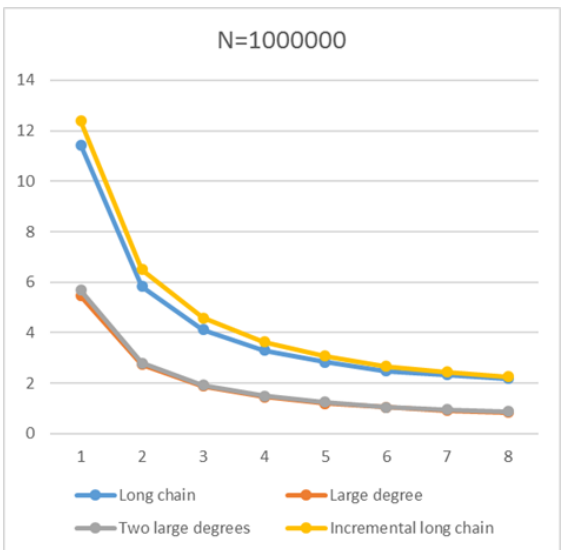
OpenMP	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.103436	1.61389	17.3962	54.0412
2 large degree	0.067749	0.638183	6.74281	21.2547
3 two large degrees	0.072668	0.638638	7.00279	21.7108
4 incremental long chain	0.340943	1.39135	17.9712	58.4357

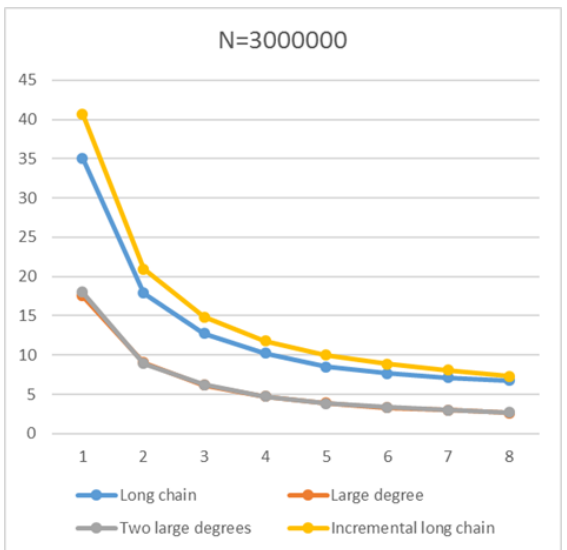
PASL	$n = 10000$	$n = 100000$	$n = 1000000$	$n = 3000000$
1 long chain	0.563715	5.00583	39.7913	121.166
2 large degree	0.306949	2.98072	27.0531	89.4415
3 two large degrees	0.322895	3.15966	28.327	87.608
4 incremental long chain	0.644581	7.4977	57.6276	167.604

Сводные результаты экспериментов: OpenMP

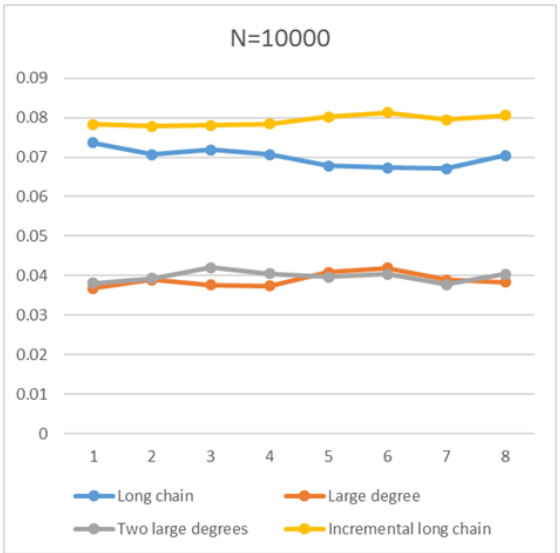


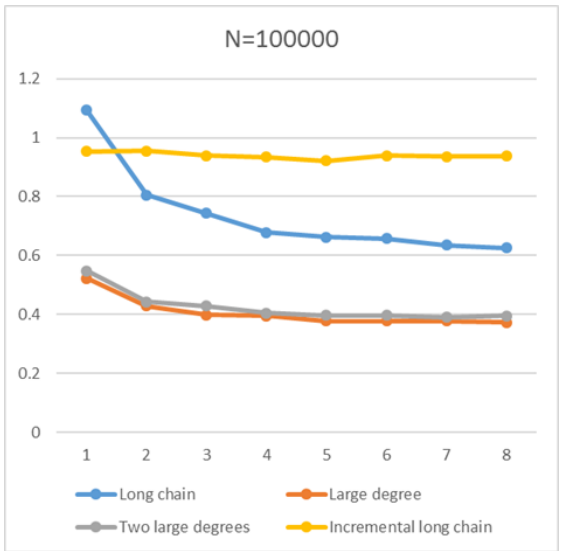


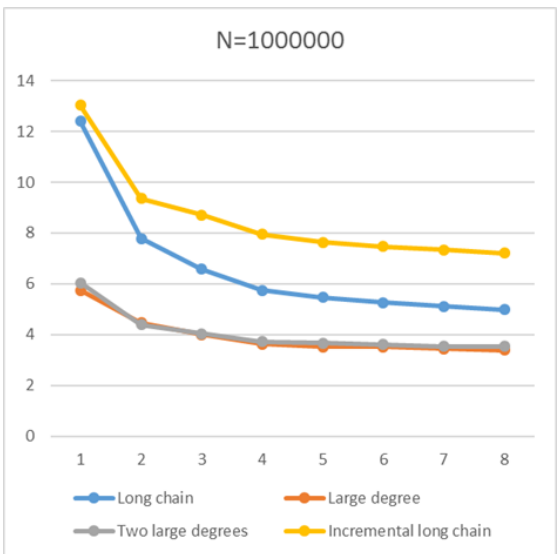


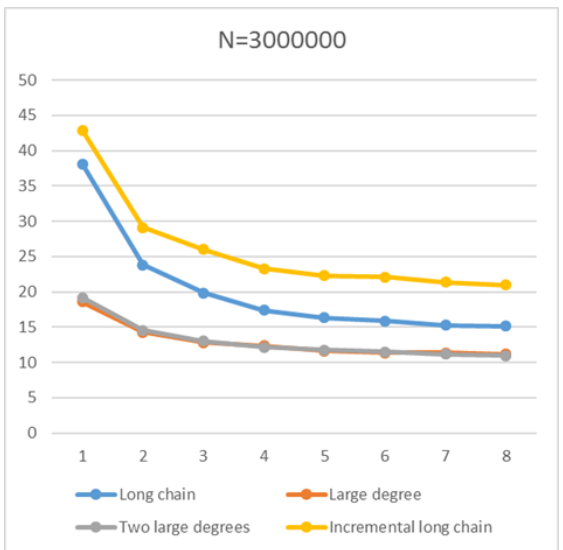


Сводные результаты экспериментов: PASL









Заключение

- Разработан параллельный алгоритм построения и применения пакетов изменений к RC-деревьям
- Разработаны параллельные реализации алгоритма с использованием библиотек параллельных вычислений OpenMP и PASL
- Распараллеливание с использованием PASL менее эффективно для работы с RC-деревьями

Апробация работы

- Публикации:
  -  Цзюньфэн У. “Обобщенная реализация RC-деревьев с параллельными обновлениями”. *V Всероссийский конгресс молодых ученых*. 2016.
  -  Цзюньфэн У и Максим Буздалов. “Обобщенная реализация укорененных Rake-and-Compress деревьев”. *Труды всероссийской конференции СПИСОК*. 2016.

Спасибо за внимание!