Исследование масштабируемости многомерного индекса на основе R-дерева

Выполнил: Ерохин Г.А. 545 гр.

Научный руководитель: Чернышев Г.А.

СПбГУ

03.06.2014

Потенциал современных платформ

- Чтобы задействовать потенциал многоядерных процессоров, необходимо эффективно использовать параллелизм
- Необходимо обеспечить корректность параллельного доступа к данным
- Доступность большого объема оперативной памяти позволяет держать БД полностью в памяти

Постановка задачи

Исследовать масштабируемость алгоритмов обеспечения транзакционности и параллельного доступа к R-дереву

- Выбрать наиболее эффективные алгоритмы параллельного доступа к R-дереву
- Проанализировать производительность выбранных алгоритмов на различном количестве потоков
- Сделать выводы о масштабируемости

Выбор алгоритмов

- Обеспечение целостности
  - Структура дерева должна остаться целостной при параллельном доступе
- Транзакционность
  - Поддержка ACID свойств

Алгоритм Корнэкера + OLFIT

- Описание
  - Если поток изменяет узел, он берет защелку, а когда отпускает - увеличивает версию
  - При обходе дерева замки не берутся. Данные считываются пока не будет получена актуальная версия
- Свойства
  - Приспособлен для работы в оперативной памяти
  - Масштабируемость близка к линейной
  - При работе в памяти, показывает наилучшие результаты

Изоляция транзакций

- Уровень изоляции - Read Committed
- Классический алгоритм: S2PL + защелки при чтении
- Алгоритм, представленный Павлом Федотовским на конференции СПИСОК'2012
  - Выше степень параллелизма (не используются защелки при чтении), лучше производительность

Измерение масштабируемости

Универсальный закон масштабируемости (USL) [Neil J. Gunther, 2007]

$$\frac{tps(N)}{tps(1)} = C(N) = \frac{N}{1 + A \times (N-1) + B \times N \times (N-1)}$$

A, B - параметры

A отвечает за ограничение производительности вследствие наличия разделяемых ресурсов

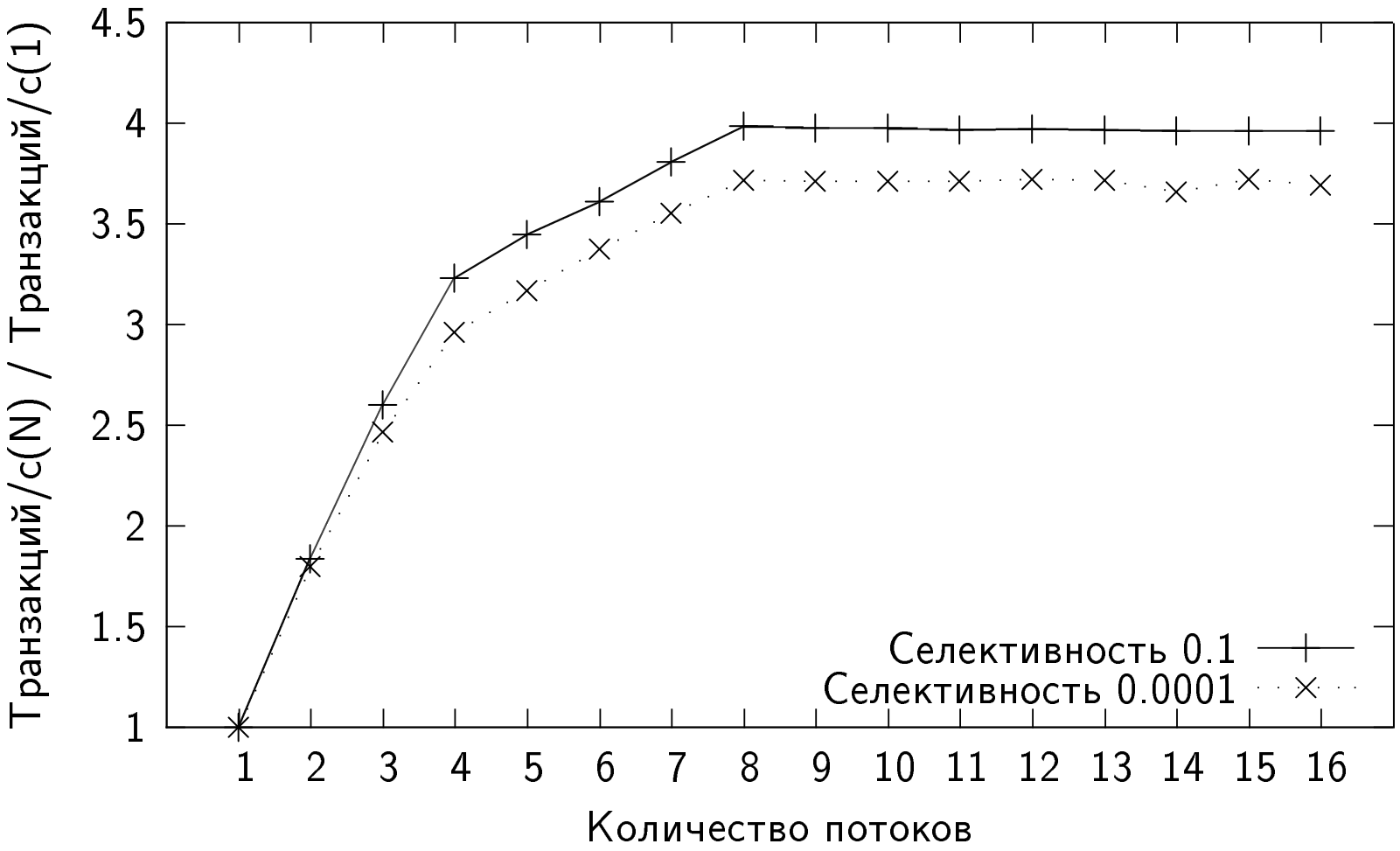
B за деградацию производительности вследствие попарного взаимодействия потоков

Конфигурация оборудования

Процессор	Intel Core i7-2670QM, 2.20 GHz (4 ядра)
Оперативная память	6 ГБ
Кэш	32 КБ L1, 256 КБ L2, 6 МБ L3
Hyperthreading	включен
ОС	Linux Ubuntu 3.2.0-29-generic x86-64
Компилятор	GCC 4.6.3

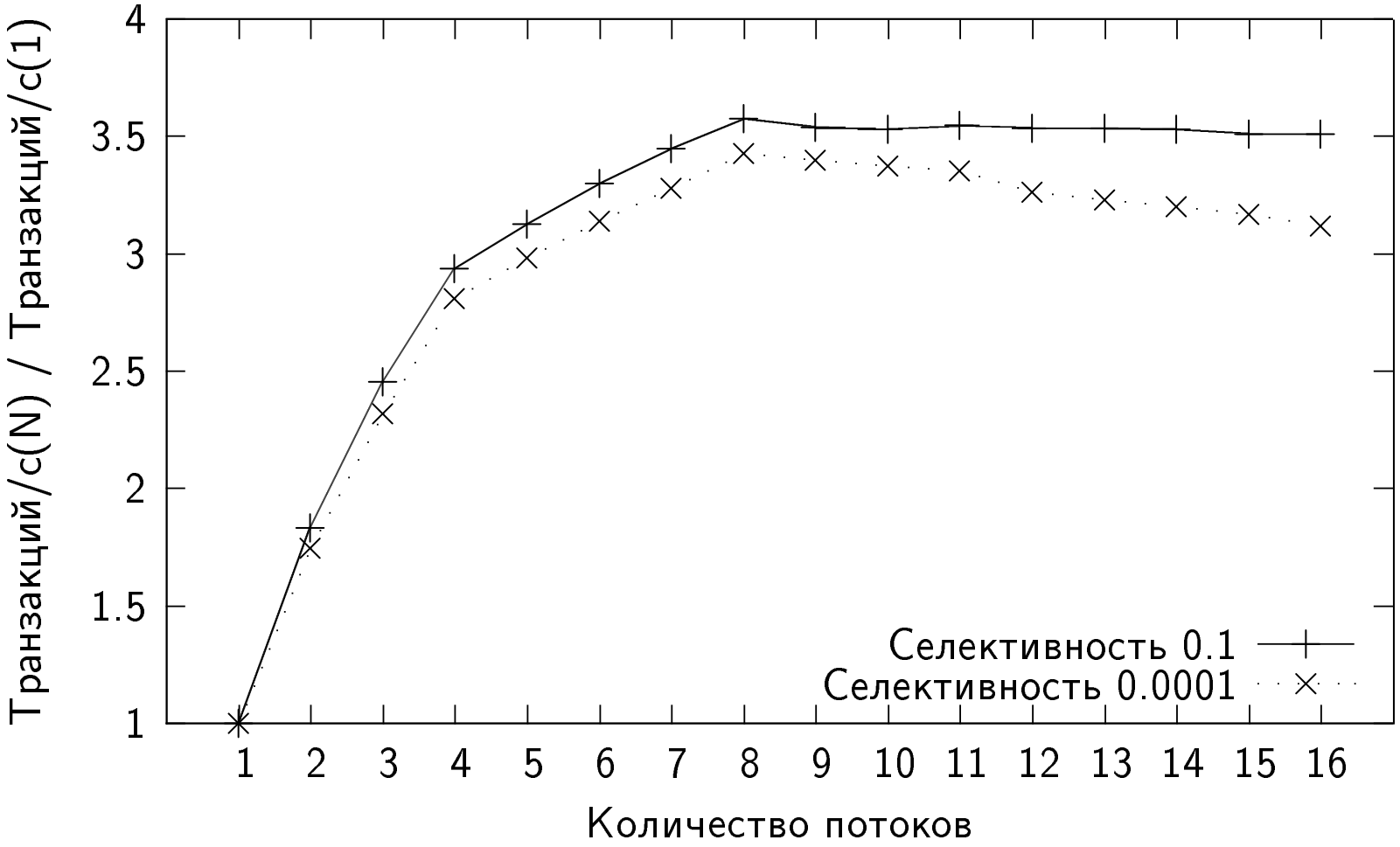
Мало запросов, изменяющих данные

Поиск: 85% Обновление: 5% Вставка: 5% Удаление: 5%



Много запросов, изменяющих данные

Поиск: 20% Обновление: 40% Вставка: 20% Удаление: 20%



Вычисление параметров модели USL

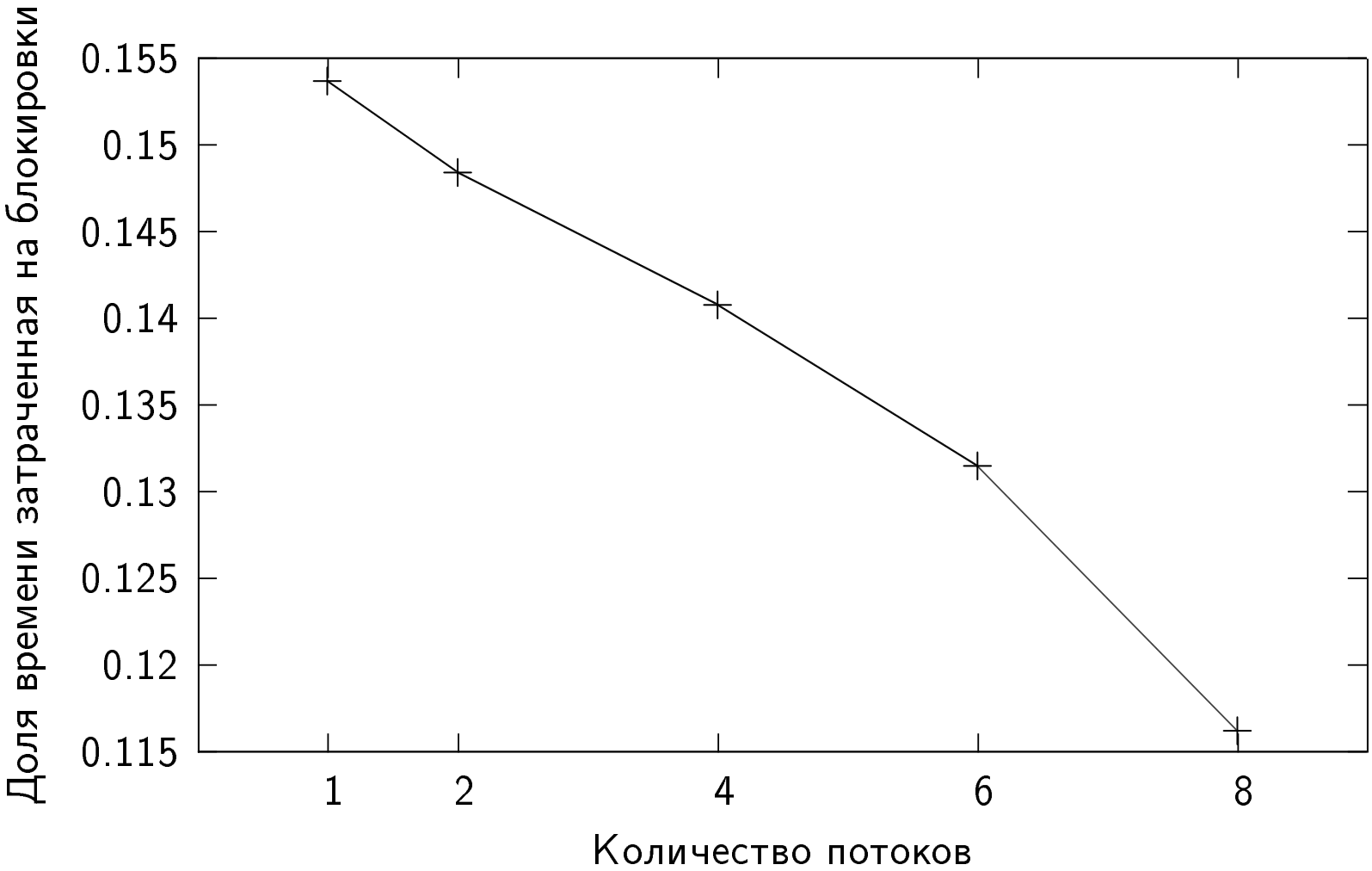
Селективность	0.1	0.0001
Поиск	$A \in (0.096, 0.117)$ $B \in (0.0068, 0.0087)$	$A \in (0.063, 0.093)$ $B \in (0.0071, 0.0097)$
Обновление	$A \in (0.097, 0.124)$ $B \in (0.0073, 0.0097)$	$A \in (0.107, 0.122)$ $B \in (0.010, 0.011)$

Профилирование

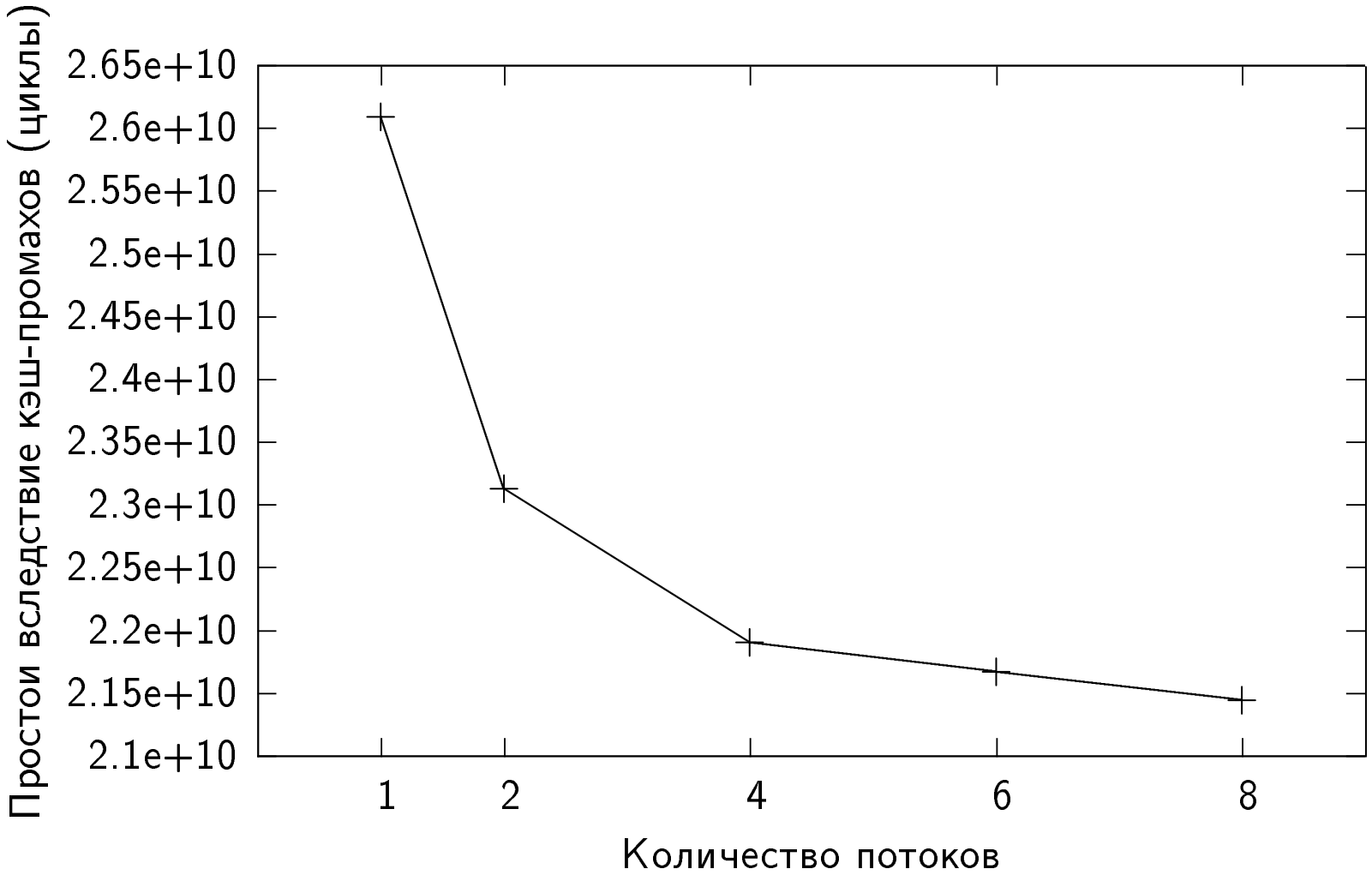
PAPI (Performance Application Programming Interface)

- Библиотека для C/C++, позволяет в коде считывать аппаратные счетчики производительности
- Корректно работает с многопоточными программами
- Позволяет считывать количество циклов, прошедших с определенного момента времени, количество простоев, связанных с обращением к памяти и т.д.

Блокировки


Доля времени затраченная на блокировки
0.155
0.15
0.145
0.14
0.135
0.13
0.125
0.12
0.115
1
2
4
6
8
Количество потоков

Кэш-промахи


Простои вследствие кэш-промахов (циклы)
2.65e+10
2.6e+10
2.55e+10
2.5e+10
2.45e+10
2.4e+10
2.35e+10
2.3e+10
2.25e+10
2.2e+10
2.15e+10
2.1e+10
1
2
4
6
8
Количество потоков

Выводы о масштабируемости

- Ухудшение производительности, связанное с кэш-промахами и замками в пересчете на один поток не увеличивается.
- В случае смешанной нагрузки и преобладании поисковых запросов, может быть принята гипотеза о том, что селективность влияет лишь на параметр альфа универсального закона масштабируемости, а параметр бета остается неизменным.
- В случае преобладания запросов, манипулирующих данными, гипотеза, по которой селективность влияет на оба параметра модели.

Результаты

- Выбраны наиболее эффективные алгоритмы параллельного доступа и обеспечения изоляции транзакций. Реализованы наиболее эффективные алгоритмы параллельного доступа: алгоритм Корнэкера и OLFIT.
- Проведен анализ производительности на различном количестве потоков для 2-х вариантов нагрузки: заполнение индекса и выполнение запросов
- Сделаны выводы о масштабируемости исследованных алгоритмов