Реализация операций над B^+ деревьями для массивно-параллельного процессора MALT-D

Кривов М.А., Елизаров С.Г.

МГУ им. М.В. Ломоносова

Москва, 18 апреля 2019

Реализация операций над B^+ деревьями для массивно-параллельного процессора MALT-D

manycore

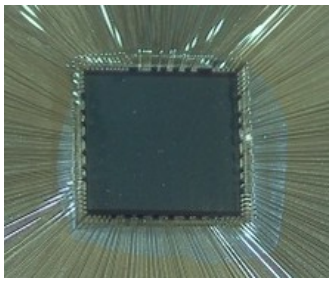
Кривов М.А., Елизаров С.Г.

МГУ им. М.В. Ломоносова

Москва, 18 апреля 2019

MALT: Manycore Architecture with Lightweight Threads

- Энергоэффективные процессоры с сотнями ядер на одном кристалле
- Разрабатываются в ЦИФ МГУ
- Архитектура адаптируется под задачи заказчика: MALT-C, MALT-D и MALT-F
- Имеются инженерные образцы (TSMC 28 нм)



Кристалл 9Mb96G,
1-ое поколение

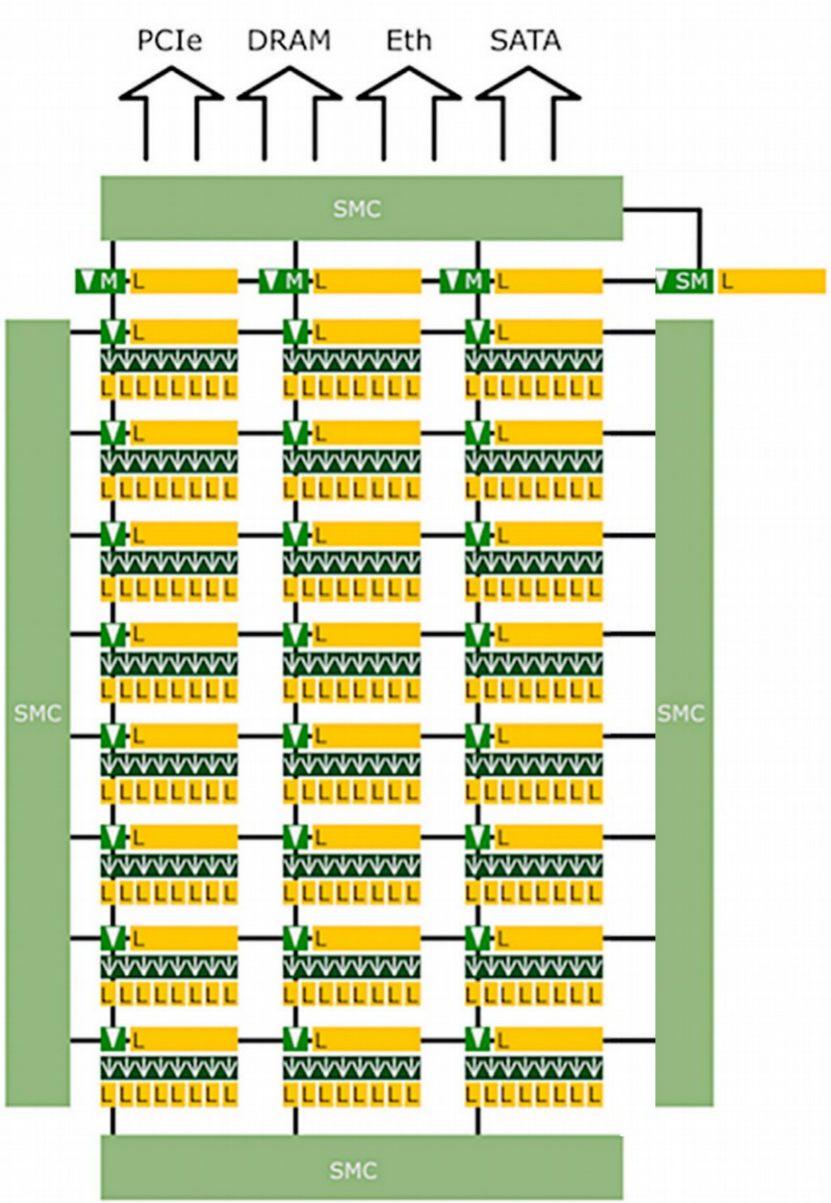
Демонстрационный
стенд

Подробнее на https://maltsystem.ru

TTG LABS

MALT: Manycore Architecture with Lightweight Threads

- Рабочая частота до 1200 МГц
- До 256 универсальных ядер
- До 1024 векторных ядер
- До 8 Мбайт ОЗУ на кристалле
- До 96 Гбайт DDR3
- PCIe, 1Gb Eth, SATA
- TDP не более 50 Вт
- Производительность до 9.8 ТОп/с



Подробнее на https://maltsystem.ru

MALT: Manycore Architecture with Lightweight Threads

- Что важно знать программисту:
  - Отсутствует кэш, но есть локальная память
  - Полноценный C++17 (gcc), но медленный std::
  - Микроархитектура ядра XILINX MicroBlaze
  - Операции с плавающей точкой опциональны
  - Имеются аппаратные «мьютексы» — FE-биты
  - Общение между ядрами через атомики и пакеты

Подробнее на https://maltsystem.ru

Задача

Реализовать B^+-деревья на MALT-D

TTG LABS

Задача

Реализовать B^+-деревья на MALT-D

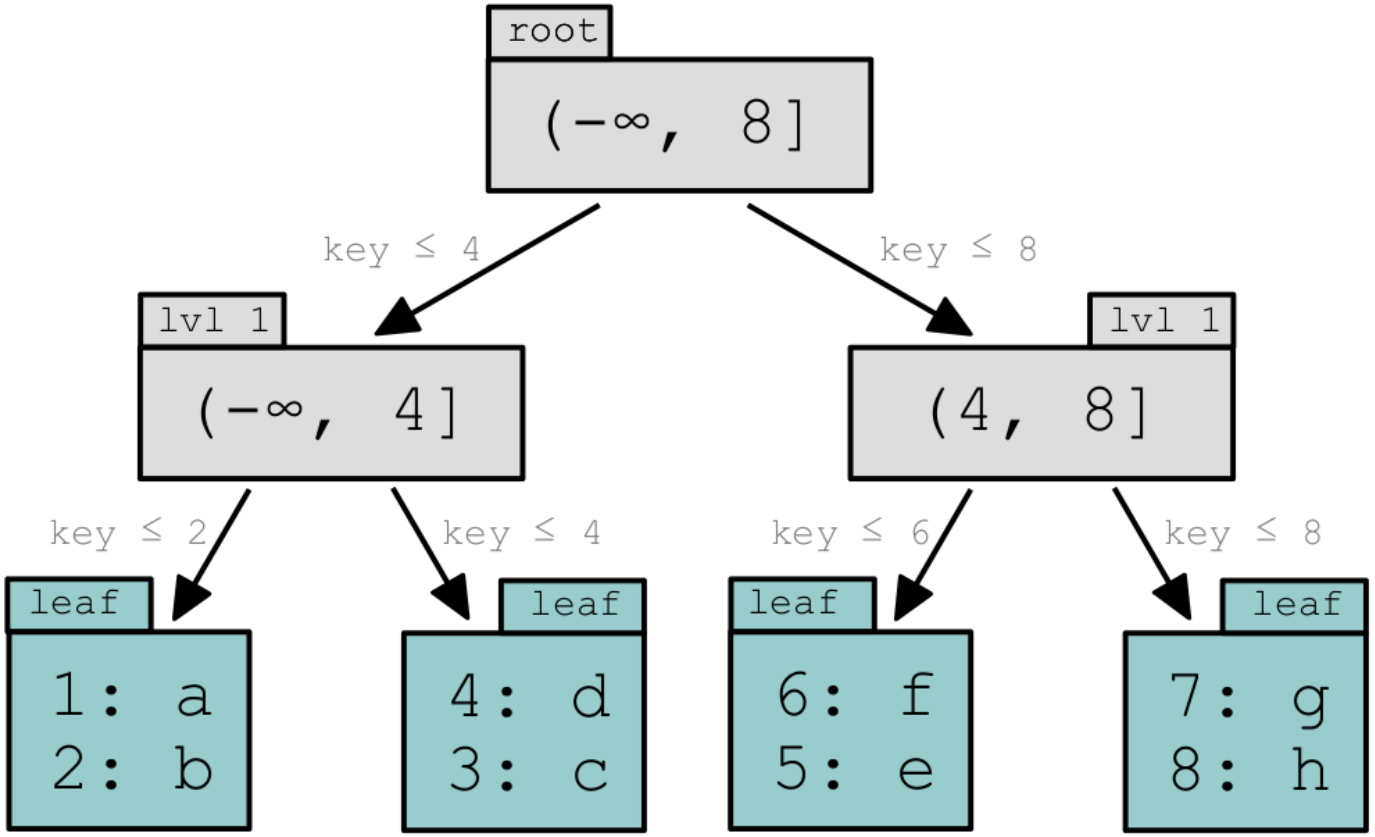
- Линейная масштабируемость на сотни ядер
- Быстрее, чем Intel Xeon по Op/(s*Watt)
- Целевой тест: R80-W20, 1 млн. ключей

TTC LABS

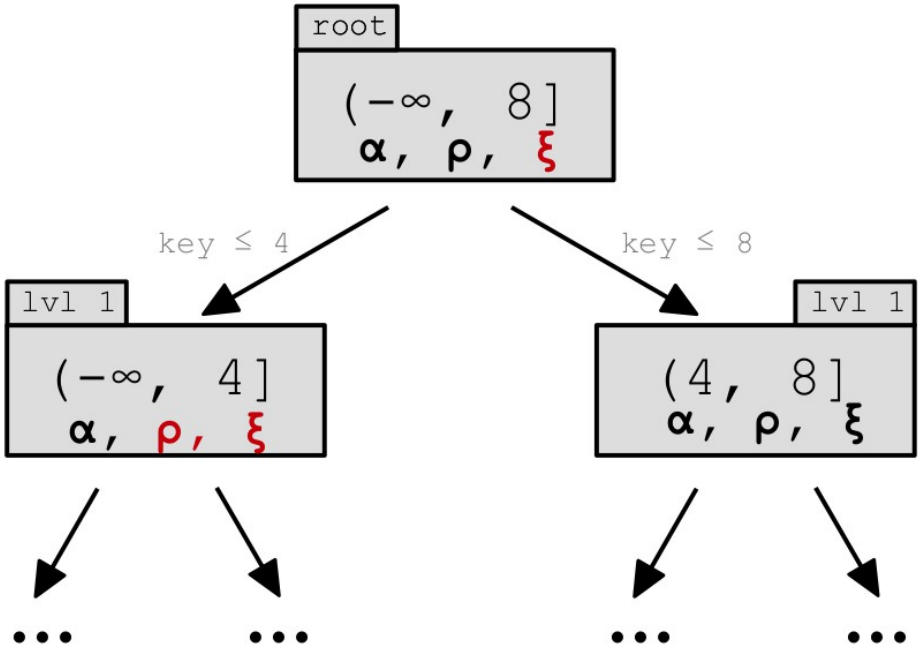
История вопроса

```
bptree map;

map[1] = 'a';
map[2] = 'b';
map[3] = 'c';
map[4] = 'd';
map[5] = 'e';
map[6] = 'f';
map[7] = 'g';
map[8] = 'h';
```



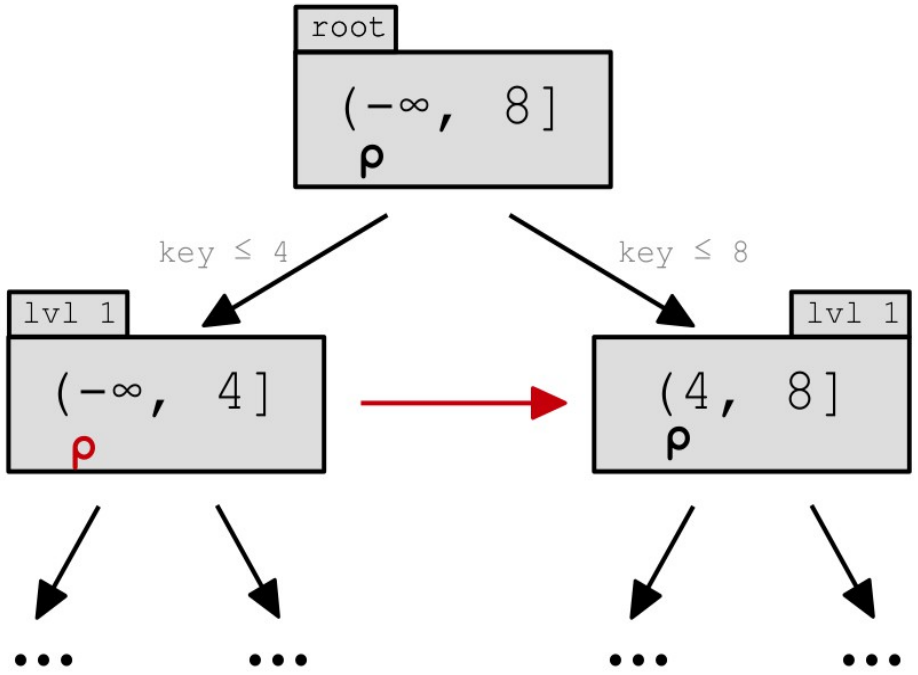
История вопроса



- Bayer (1977)*
  - Паттерн читатели-писатели
  - На узел по 3 типа замка
  - Есть правила конвертации замков

* Bayer R., Schkolnick M., Concurrency of Operations on B-Trees, 1977

История вопроса



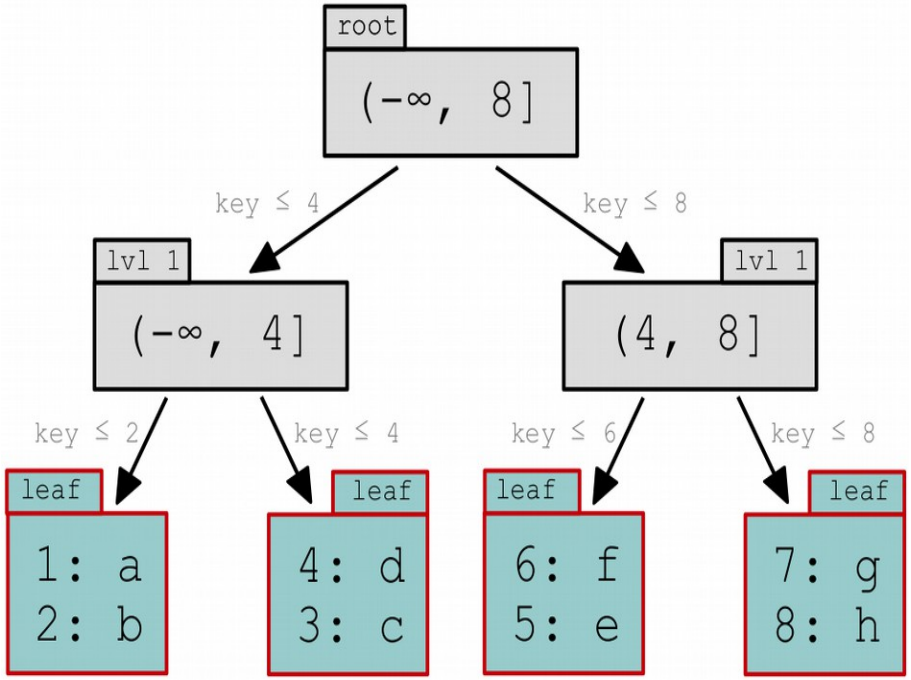
- Lehman (1981)*
  - Изменение структуры, теперь B^{link}-tree
  - Вводятся горизонтальные ссылки между узлами
  - Достаточно замков одного типа

* Lehman P., Yao S., Efficient Locking for Concurrent Operations on B-Trees, 1981

История вопроса

...

История вопроса

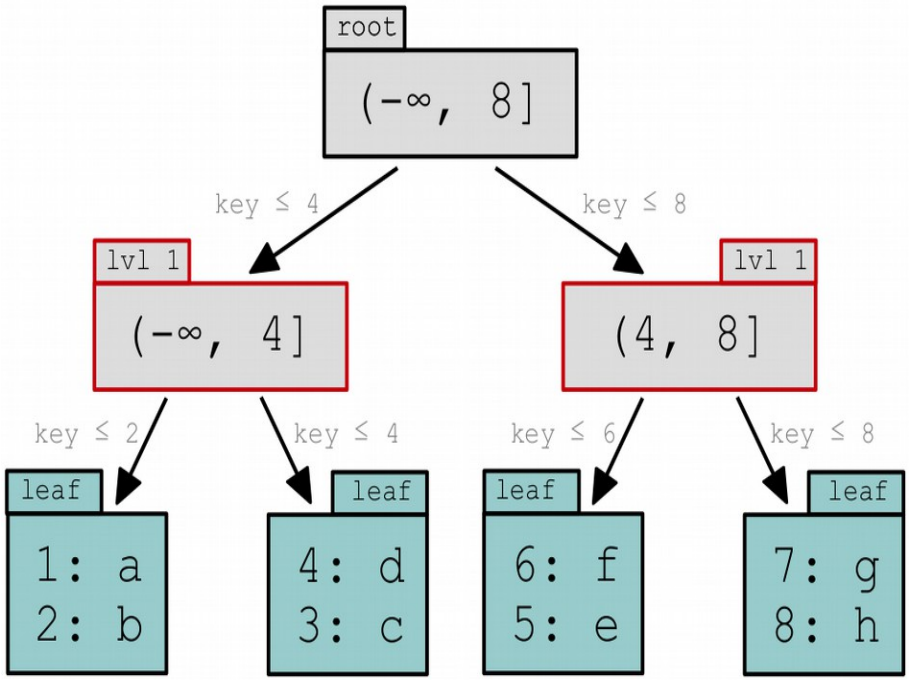


- Sewall (2011)*
  - Классические B$^+$-деревья, но оригинальный алгоритм
  - Запросы обрабатываются группами, параллелизм только внутри группы
  - Без каких-либо замков, до 19 быстрее аналогов в RW
  - Реализован сторонними коллективами**, ***

* Sewall J. et al, PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors, 2011
** Saldanha N., Mahadevan K. Implementation of PALM: Parallel Architecture-Friendly LatchFree Modifications to B+ Trees on Many-Core Processors, 2012
*** https://github.com/runshenzhu/palmtree

TTC LABS

История вопроса

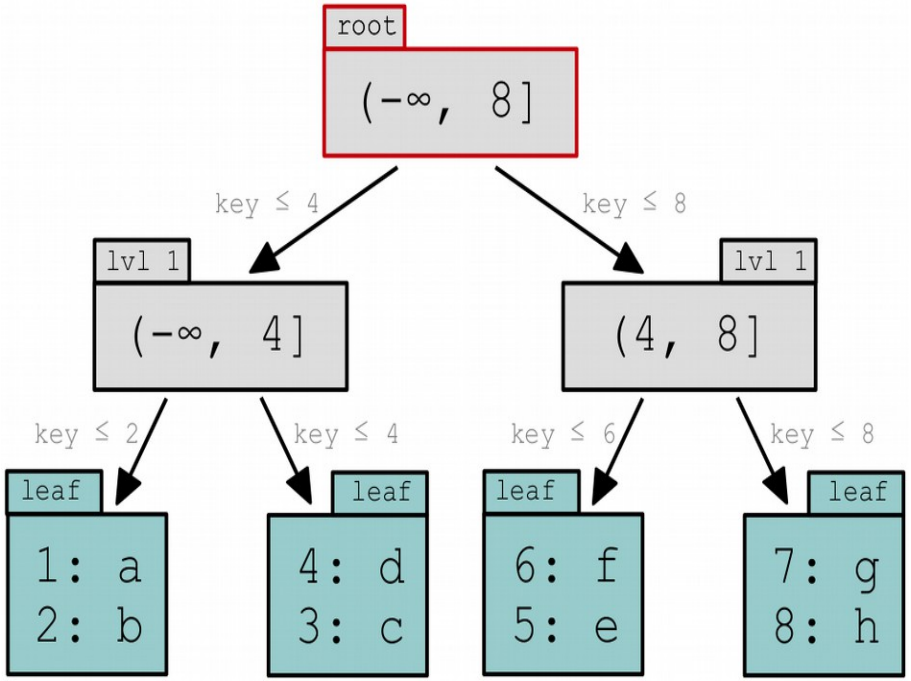


- Sewall (2011)*
  - Классические B$^+$-деревья, но оригинальный алгоритм
  - Запросы обрабатываются группами, параллелизм только внутри группы
  - Без каких-либо замков, до 19 быстрее аналогов в RW
  - Реализован сторонними коллективами**, ***

* Sewall J. et al, PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors, 2011
** Saldanha N., Mahadevan K. Implementation of PALM: Parallel Architecture-Friendly LatchFree Modifications to B+ Trees on Many-Core Processors, 2012
*** https://github.com/runshenzhu/palmtree

TTC LABS

История вопроса

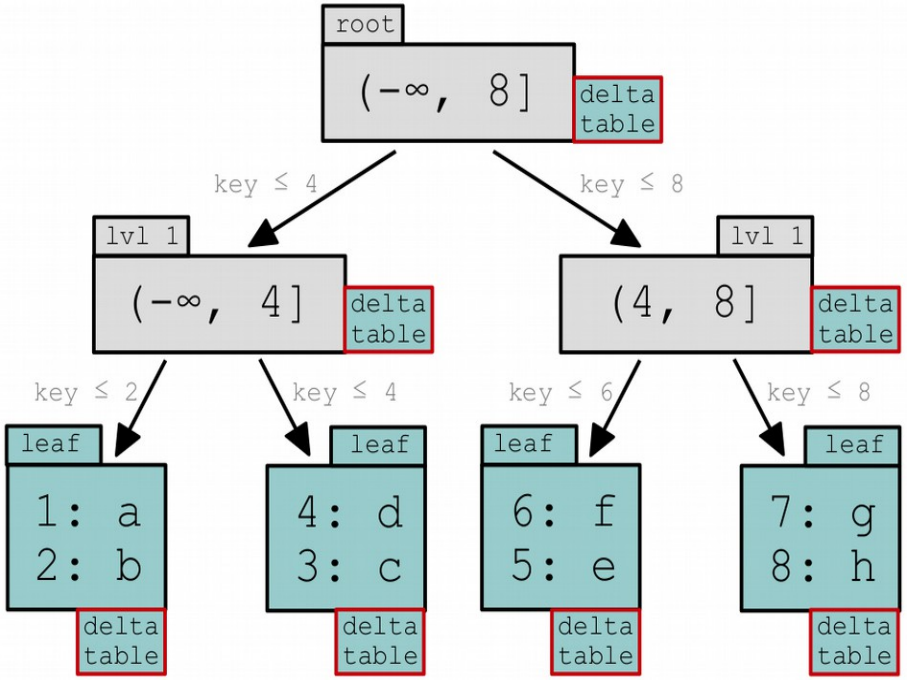


- Sewall (2011)*
  - Классические B^+-деревья, но оригинальный алгоритм
  - Запросы обрабатываются группами, параллелизм только внутри группы
  - Без каких-либо замков, до 19 быстрее аналогов в RW
  - Реализован сторонними коллективами**, ***

* Sewall J. et al, PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors, 2011
** Saldanha N., Mahadevan K. Implementation of PALM: Parallel Architecture-Friendly LatchFree Modifications to B+ Trees on Many-Core Processors, 2012
*** https://github.com/runshenzhu/palmtree

TTG LABS

История вопроса

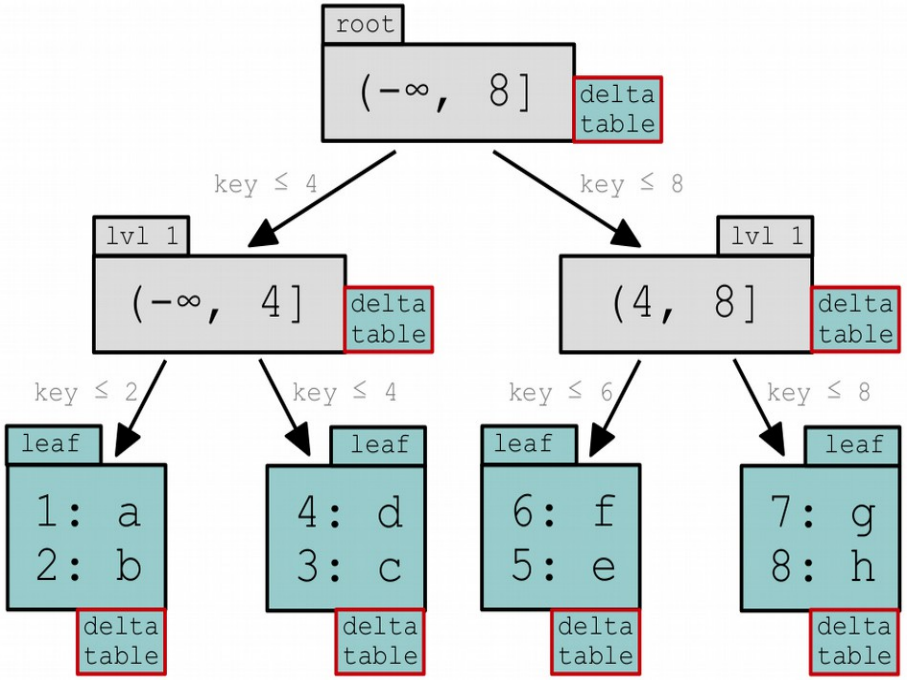


- Levandoski (2013)*
  - Изменение структуры, теперь B^w-tree
  - Узлы снабжаются журналом обновлений
  - Изменения вносятся при переполнении журнала
  - Отсутствуют блокировки за счёт дублирования работы

* Levandoski J., Lomet D., The Bw-Tree: A B-tree for New Hardware Platforms, 2013

TTC LABS

История вопроса



- Levandoski (2013)*
  - Изменение структуры, теперь B^w-tree
  - Узлы снабжаются журналом обновлений
  - Изменения вносятся при переполнении журнала
  - Отсутствуют блокировки за счёт дублирования работы

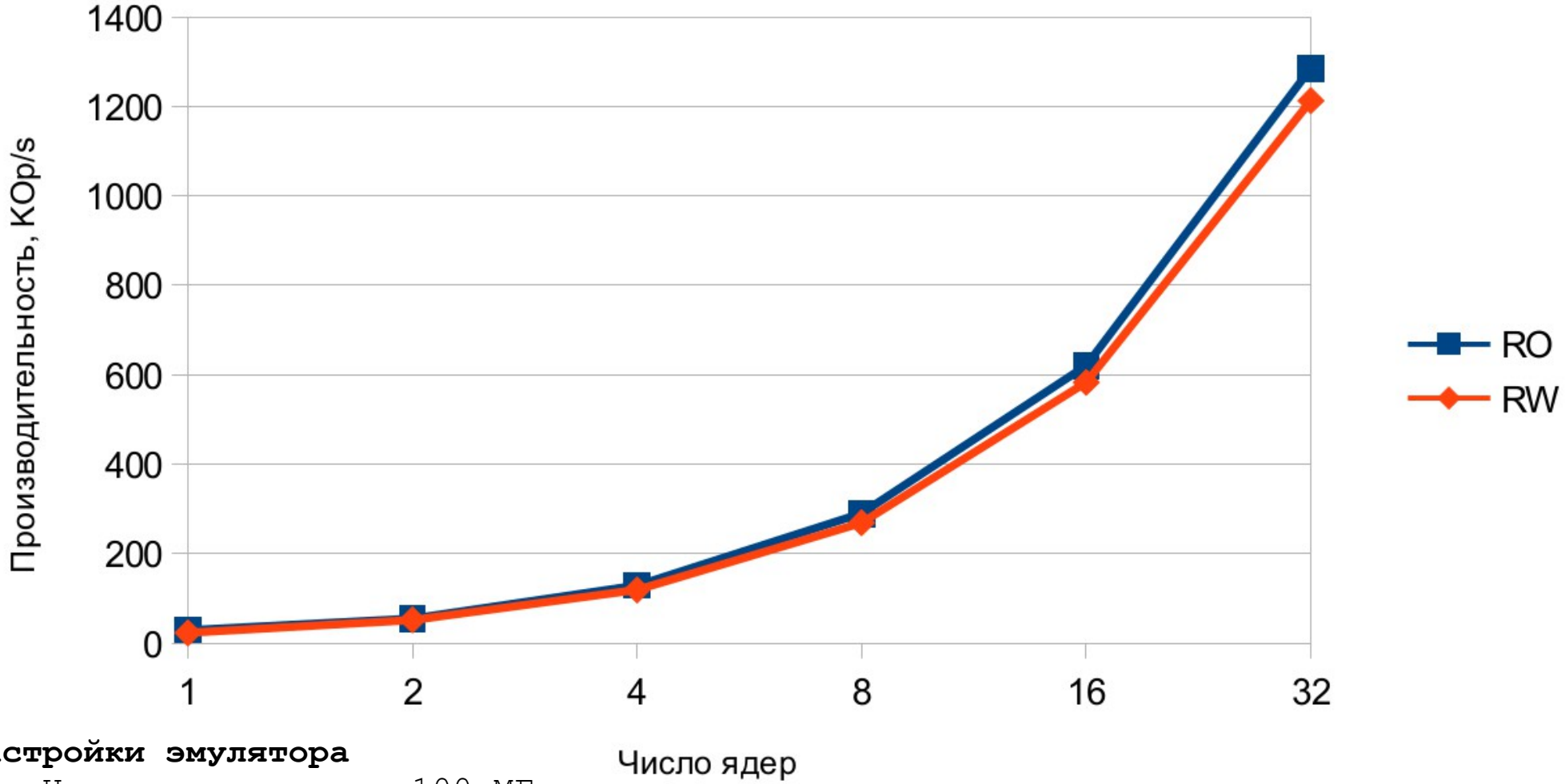
* Levandoski J., Lomet D., The Bw-Tree: A B-tree for New Hardware Platforms, 2013

TTC LABS

Что сделано нами

- Алгоритм PALM (Sewall 2011) перенесён на MALT-D и оптимизирован под MicroBlaze
- Проведено тестирование на 1-16 млн. int-ключах в режимах R80W20 и R100
- Программный эмулятор прогнозирует суперлинейную масштабируемость по ядрам
- При запусках на FPGA выявлены аппаратные проблемы → финальных результатов пока нет, только оценки

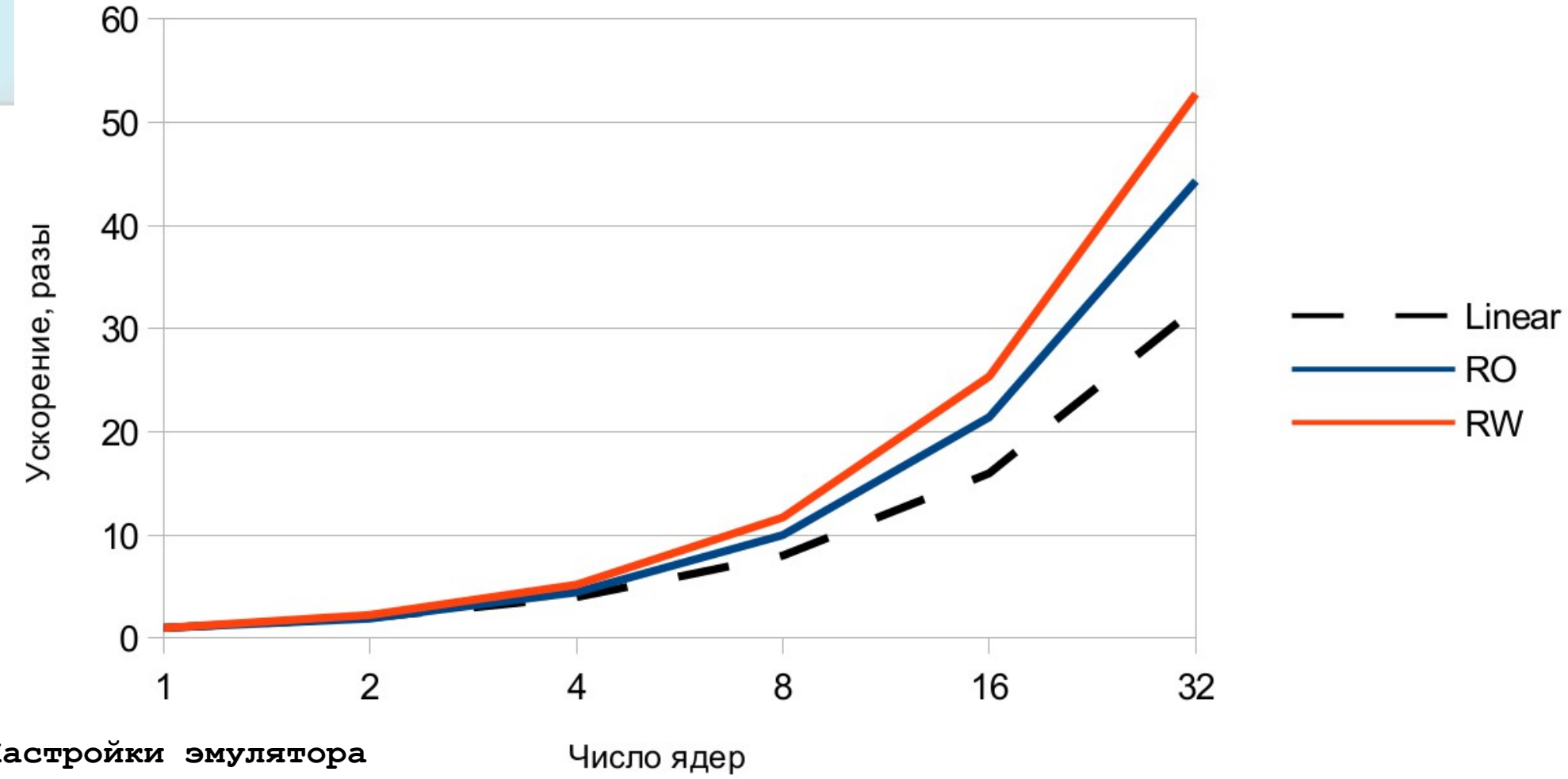
Масштабируемость по ядрам



Настройки эмулятора

```
Частота:            100 МГц
Задержка чтения:    50 тактов
Задержка записи:    5 тактов
Проп.способность:   ∞
Локальная память:   8 КБайт
```

Масштабируемость по ядрам



Настройки эмулятора

```
Частота:              100 МГц
Задержка чтения:      50 тактов
Задержка записи:      5 тактов
Проп.способность:     ∞
Локальная память:     8 КБайт
```

Оценка энергоэффективности

Процессор	Тех.проц., нм	Число ядер, шт	Частота, МГц	Производит., MOp/s	TDP, Ватт	Эффективн., MOp/(s*W)
MALT-D v1[1]	28	16	100	1,2	2	0,6
MALT-D v2[2]	20	96	100	8,1	11	0,73
Xeon X5680 x2[*]	32	6*2	3300	~110	130*2	0,42
Xeon E5-2666v3 x2[**]	22	10*2	2900	39,6	105*2	0,19

[1] Xilinx Kintex-7 xc7k325
[2] Xilinx Kintex UltraScale xcku115
[*] Sewall J. et al, PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors, 2011
[**] https://github.com/runshenzhu/palmtree

TTC LABS

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
- Самоочистка памяти
- Кэширование переменных
- Специализация функций
- Борьба за такты

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
  - Буфер изменяемого размера
  - Потокобезопасная хэш-таблица
- Самоочистка памяти
- Кэширование переменных
- Специализация функций
- Борьба за такты

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
- Самоочистка памяти
  - В каждом int'е 4 бита выделяются под «поколение», в рамках которого записано значение
- Кэширование переменных
- Специализация функций
- Борьба за такты

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
- Самоочистка памяти
- Кэширование переменных

```
for (int i = 0; i < p->size; i++)
{ … }
```

```
auto size = p->size;
for (int i = 0; i < size; i++)
{ … }
```

- Специализация функций
- Борьба за такты

TTG LABS

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
- Самоочистка памяти
- Кэширование переменных
- Специализация функций

```
void f(int *arr, int size)
{ … }
```

```
void f(int *arr, int size)
{ … }
void f_1(int *arr) { … }
void f_2(int *arr) { … }
```

- Борьба за такты

TTC LABS

Специфика оптимизации PALM под MALT-D

- Контейнеры данных
- Самоочистка памяти
- Кэширование переменных
- Специализация функций
- Борьба за такты
  - Ручная развёртка циклов for
  - Упаковка типов по битам
  - ...

TTG LABS

Вопросы?

(m_krivov@cs.msu.su)

LABS