Кэш-зависимые структуры данных

Ерохин Георгий, 445 группа

Научные руководители:
Чернышев Г.А.
Смирнов К.К.

Введение

- Структуры в стандартной библиотеке универсальны, но недостаточно производительные для ряда задач
- Одна из возможных оптимизаций - учет кэша процессора
  - Организовывать доступ к данным более рациональным образом
  - При проектировании структур
  - При выделении памяти (менеджер памяти)
  - Перестройка структур во время исполнения программы

Постановка задачи

Прототип высокопроизводительной многопоточной системы обработки документов:

- Изучить, как можно эффективно использовать кэш в системе
- Реализовать критичные места с учетом использования кэша

Узкие места

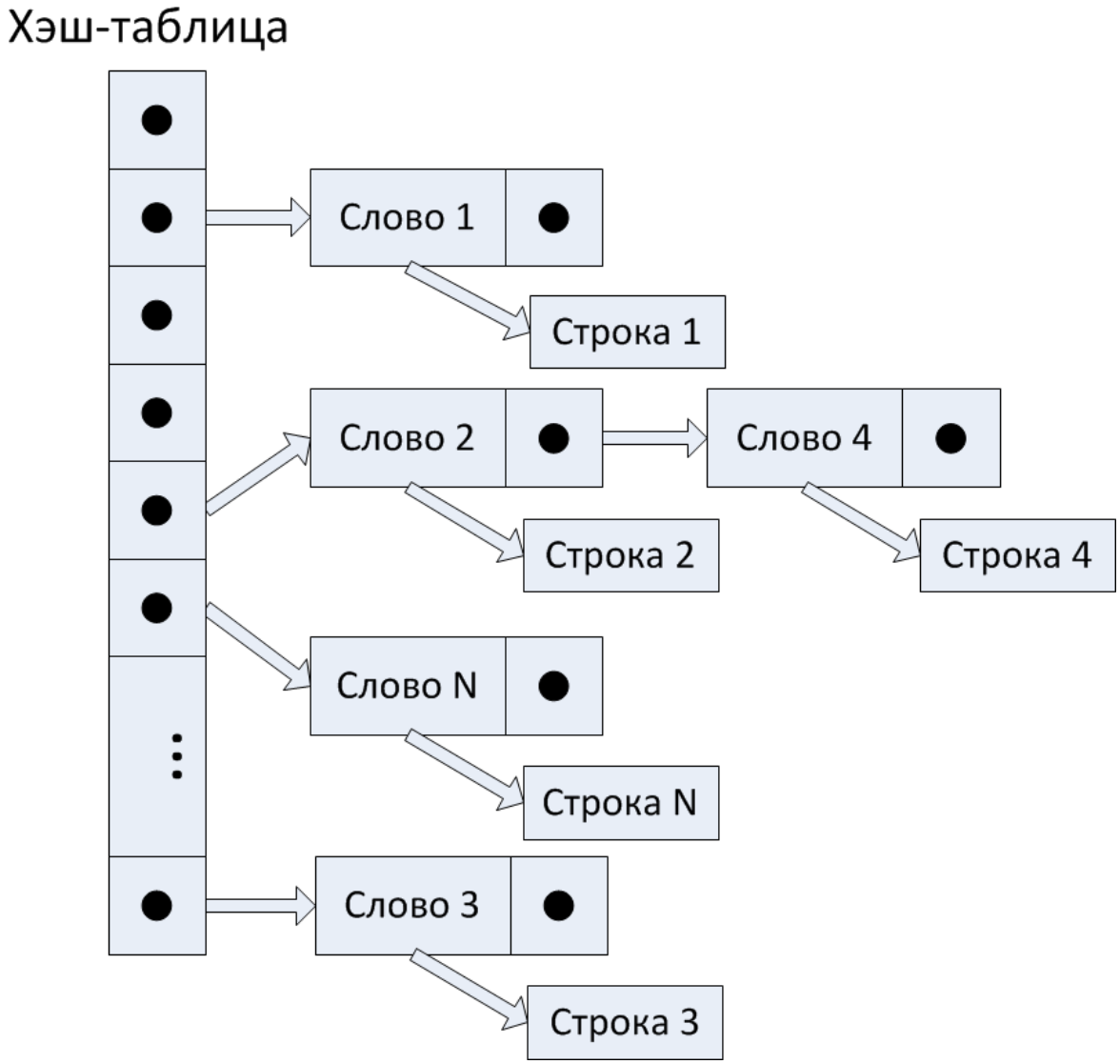
- 53% времени проводится в методах класса стандартной библиотеки std::unordered_set
- Требуемые операции:
  - Добавление
  - Поиск
  - Обход

Решения

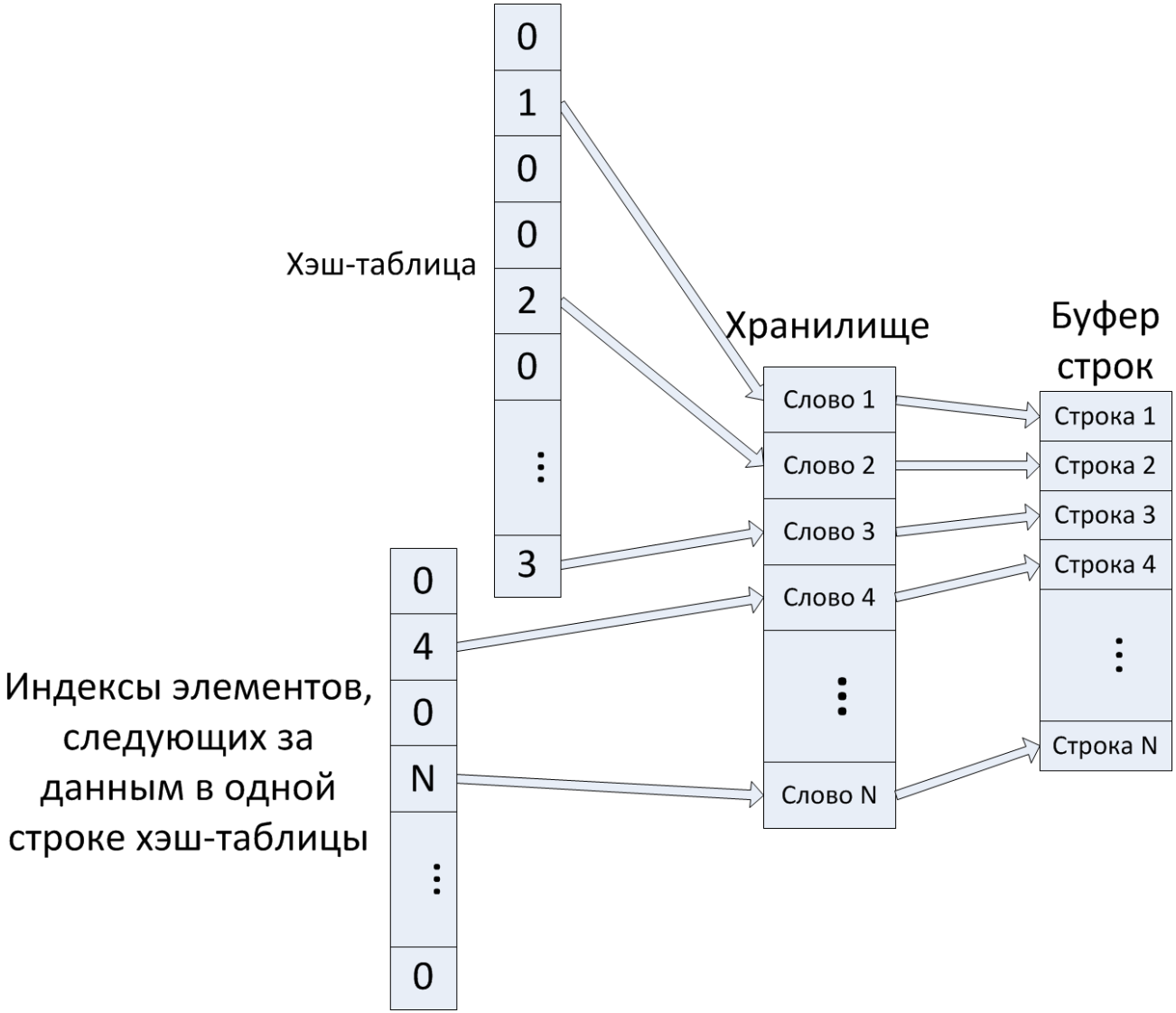
- CSS-Tree
- T-Tree
- Хэш-таблицы
- Если упорядоченность не требуется, хэш-таблицы обеспечивают наилучшее быстродействие[+]

+) Jun Rao and Kenneth A. Ross. 1999. Cache Conscious Indexing for Decision-Support in Main Memory. In *Proceedings of the 25th International Conference on Very Large Data Bases*

Классическая хэш-таблица



Реализация с учетом кэша



Тестирование

Производительность системы на различных тестах:

	unordered_set	Наша реализация
Test1	0.07 с	0.06 с
Test2	1.83 с, δ=2%	1.70 с, δ=1%
Test3	50.32 с, δ=1%	43.46 с, δ=1%

Результаты

- Изучены узкие места в системе - STL библиотека
- Ускорение всей системы на ~14% при замене std::unordered_set на более эффективную реализацию
- Результат войдет в доклад на конференции ACM SIGMOD'2013