***17. Подсчет ссылок. Двухпальцевый алгоритм.***

Jones, Hosking, Moss - The garbage collection handbook  
3.1 Two-finger compaction стр. 32

Reference counting стр. 57

* Счетчик ссылок для каждого объекта
* Алгоритм
* Установка ссылки – увеличение счетчика
* Разрушение ссылки – уменьшение счетчика
* Объект с 0 ссылок мертв
* Достоинства
* Не требует остановки нитей
* Работает только с локальными объектами, 0 задержка
* Недостатки
* Дополнительная память для каждого объекта
* Требует инструментации управляемого кода изменением счетчиков
* Не отслеживает циклические ссылки, например, двусвязный список
  + Фоновое сканирование циклов
* Модификация счетчика – атомарная операция
  + Журналирование операций каждой нити. Требует регулярного объединения журналов. Создает ненулевую задержку.

Двухпальцевый алгоритм:

Относится к категории compact(Mark and compact) — полный и останавливающий. Сейчас такие алгоритмы не используются.

The major benefit of a compacted heap is that it allows very fast, sequential allocation, simply by testing against a heap limit

and 'bumping' a free pointer by the size of the allocation request

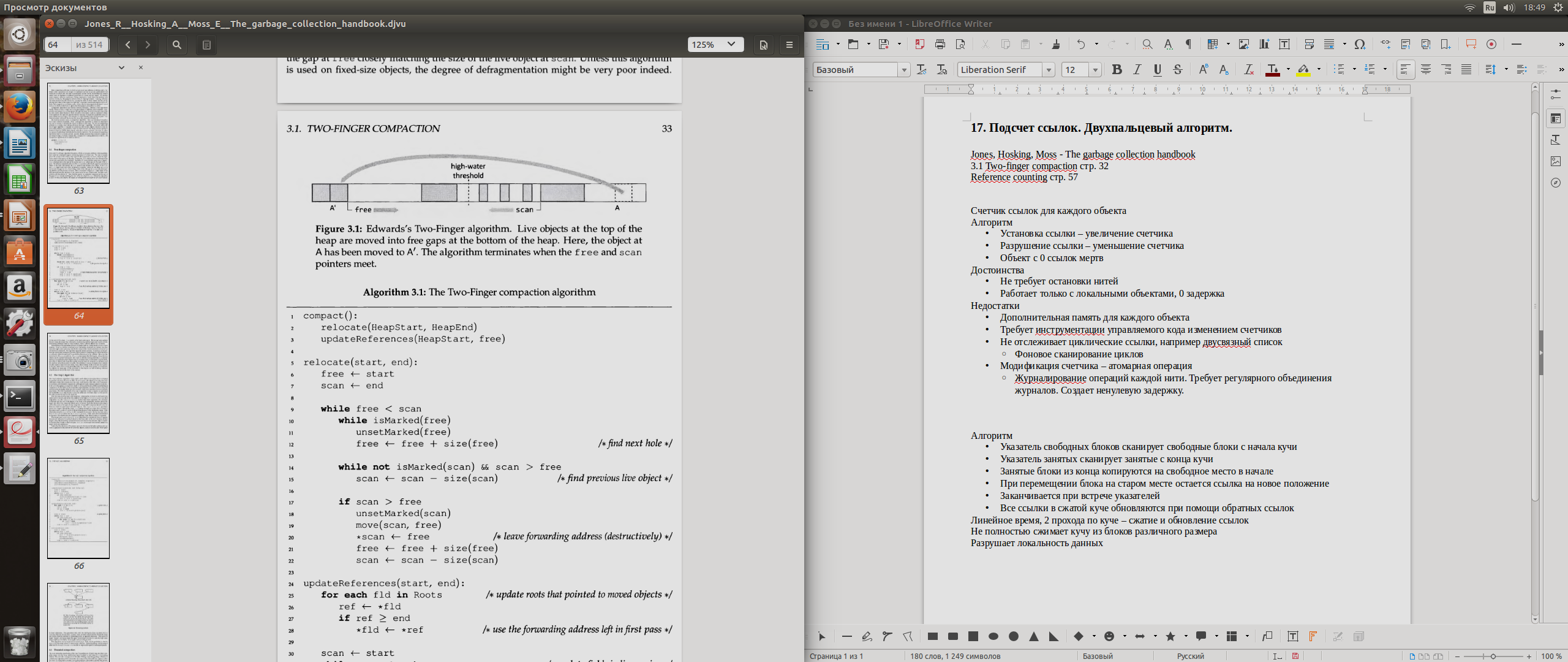
Алгоритм

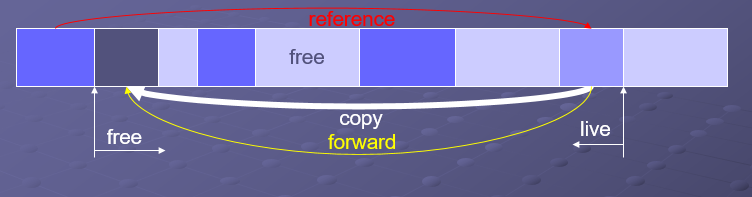
* Указатель свободных блоков сканирует свободные блоки с начала кучи
* Указатель занятых сканирует занятые с конца кучи
* Занятые блоки из конца копируются на свободное место в начале
* При перемещении блока на старом месте остается ссылка на новое положение
* Заканчивается при встрече указателей
* Все ссылки в сжатой куче обновляются при помощи обратных ссылок

Линейное время, 2 прохода по куче – сжатие и обновление ссылок

Не полностью сжимает кучу из блоков различного размера

Разрушает локальность данных





Mark & Compact:

Уплотнение кучи – объединение свободных блоков в непрерывную область

Алгоритм

* Остановка пользовательских нитей
* (Mark) Перечисление живых объектов методом трассировки ссылок (tracing)
* (Compact) уплотнение кучи и обновление ссылок
* Возобновление пользовательских нитей

Сложность распараллеливания