**20. Break Table, Mark/Compact, Mark/Sweep/Compact.**

Break Table

https://en.wikipedia.org/wiki/Mark-compact\_algorithm#Table-based\_compaction

Алгоритм

* Непрерывные занятые блоки последовательно сдвигаются к началу кучи
* При сдвиге блока с номером i в таблицу заносится (ai, si)
  + ai – адрес начала блока до сдвига
  + si – сдвиг
* Для каждой ссылки r двоичным поиском в таблице находится блок ai, такой ai ≤ r < a(i+1)
* Значение ссылки заменяется на r’ = r – si

Для хранения таблицы используются пустые блоки кучи

Для ускорения поиска в таблице используется хеш-таблица по старшим битам ссылки.

Mark/Compact

Обычный Mark/Sweep приводит к фрагментации. Цель алгоритмов Mark/Compact — свести живые объекты в куче.

Jones, Hosking, Moss - The garbage collection handbook

Mark-compact garbage collection стр. 31

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

Более подробное описание конкретного алгоритма

3.2 The Lisp 2 algorithm стр 34

Mark/Sweep/Compact

M&C может использоваться совместно M&S при обнаружении избыточной фрагментации

Избыточная фрагментация – невозможность разместить непрерывный блок в памяти при наличии достаточного свободного места в блоках меньшего размера

Стадии алгоритма:

1) «Mark»: помечаются неиспользуемые объекты (красные).  
  
2) «Sweep»: эти объекты удаляются из памяти. Обратите внимание на пустые слоты на диаграмме.  
  
3) «Compact»: объекты размещаются, занимая свободные слоты, что освобождает пространство на тот случай, если потребуется создать «большой» объект.   
