***19. Parallel Mark/Sweep. Concurrent Mark/Sweep (IBM JDK)***

Jones, Hosking, Moss - The garbage collection handbook  
16 Concurrent mark-sweep стр. 323

Хорошая статья:

[Сборщик мусора Concurrent Mark-Sweep](https://blogs.oracle.com/vmrobot/entry/сборщик_мусора_concurrent_mark_sweep)

Concurrent Mark/Sweep не останавливает пользовательские нити, работает медленно и без задержек

Алгоритм

* Конкурентная разметка
* Конкурентный подбор изменений сохраняющий трехцветный инвариант
* Остановка нитей
* Завершение разметки
* Возобновление нитей
* Параллельное подметание кучи

А именно:

* начальная маркировка (initial mark): работа CMS начинается с очень короткой stop-the-world паузы, в течении которой сборщик мусора находит так называемые корневые (root) ссылки на объекты, созданные приложением (корневые ссылки- это, например, ссылки, сохранённые в регистрах или в стеках потоков приложения).
* одновременная маркировка (concurrent mark): во время этого этапа CMS отмечает все объекты, достижимые из корневых объектов, то есть все «живые» объекты, которые не должны удаляться сборщиком мусора. Этот этап выполняется одновременно с работой приложения.
* повторная маркировка (remark): так как во время этапа одновременной маркировки потоки приложения продолжали работать, и в это время они могли создавать новые объекты и модифицировать ссылки, то к концу этого этапа нет гарантии, что все живые объекты были отмечены. Чтобы решить эту проблему CMS ещё раз приостанавливает приложение и завершает маркировку, проверяя все объекты, которые были изменены пользовательскими потоками во время выполнения предыдущего этапа. Поскольку повторная маркировка может занимать довольно много времени, то на многопроцессорных машинах этот этап выполняется несколькими параллельными потоками.
* одновременная очистка (concurrent sweep): после завершения этапа повторной маркировки все живые объекты в приложении были отмечены, и во время этапа одновременной очистки CMS удаляет весь найденный мусор (все неотмеченные объекты). Этот этап не требует приостановки приложения.

Parallel Mark and Sweep плохо гуглится, есть просто Parallel GC

<http://www.tikalk.com/java/garbage-collection-serial-vs-parallel-vs-concurrent-mark-sweep/>

Next off is the Parallel collector. This is the JVM’s default collector. Much like its name, its biggest advantage is that is uses multiple threads to scan through and compact the heap. The downside to the parallel collector is that it will stop application threads when performing either a minor or full GC collection. The parallel collector is best suited for apps that can tolerate application pauses and are trying to optimize for lower CPU overhead caused by the collector.

В чем разница между Parallel и CMS?

CMS исполняет следующие шаги (все исполняются одной нитью GC):

- initial mark

- concurrent marking

- remark

- concurrent sweeping

there are two differences between a parallel and a CMS collectors:

1) the parallel использует множественные нити GC, в то время как CMS использует только одну.

2) параллельный является 'stop-the-world' сборщиком, в то время как CMS «останавливает мир» только во время фаз initial mark и remark.

Во время фаз concurrent marking и sweeping, нить CMS работает вместе с нитями приложения.