ПРОБЛЕМЫ ПРОИЗВОДИТЕЛЬНОСТИ JAVA-ПРИЛОЖЕНИЙ

Спичеков А. В., Спичекова Н. В.

ООО «Иншуранс Солюшенс», Минск, Беларусь,
УО «Белорусский государственный университет информатики и радиоэлектроники», Минск, Беларусь,
e-mail: a.spichakou@gmail.com, n.spichekova@gmail.com

Анализ показывает, что большинство проблем, связанных с производительностью java приложений, могут быть отнесены к одной из следующих категорий: проблемы с базой данных, проблемы с памятью, проблемы, вызванные одновременным (параллельным) выполнением нескольких действий [1].

Являясь базовым компонентом подавляющего большинства современных веб-приложений, база данных одновременно выступает в качестве основного источника проблем с производительностью, причиной которых могут быть неверно организованный доступ к базе данных, неверный размер пула соединения базы данных, отсутствие тюнинга. В частности, неверный выбор опций lazy / eager fetching, отсутствие кэширования или его неверная настройка ведут к увеличению времени отклика приложения.

Большинство проблем с памятью в java-приложениях связаны со сборщиком мусора и утечками памяти. Сбор мусора может приводить к прерыванию работы приложения. Для уменьшения негативного влияния сбора мусора на производительность приложения необходимо правильно настроить размер кучи, а также использовать циклический запуск виртуальных машин java в эластичном окружении.

Если в часто исполняемом фрагменте кода приложения сохраняются ссылки на неиспользуемые в последующем объекты, то потребность в памяти возрастает, размер кучи становится недостаточным и происходит утечка памяти, приводящая к ошибке OutOfMemory, которая, как правило, требует перезагрузки JVM. Для решения проблемы утечки памяти необходимы правильная конфигурация параметров JVM и аккуратная работа с java-контейнерами, допускающими неограниченный рост.

При параллельном выполнении нескольких действий в java используются механизмы синхронизации и блокировок, которые могут привести к следующим проблемам производительности: взаимоблокировкам (thread deadlocks), возникающим тогда, когда один из двух потоков, пытающихся получить доступ к одним и тем ресурсам, ждет, пока второй поток освободит нужный ему ресурс, и наоборот; thread gridlocks, возникающим в том случае, когда все потоки долго ждут освобождения одного и того же ресурса; блокировкам настройки пула потоков (thread pool configuration locks), возникающих при неверной настройке размера пула потоков.

Литература