Подходы к реализации программных тредов для мультитредово-потокового суперкомпьютера

Фролов А.С., Эйсымонт Л.К.

В ОАО "НИЦЭВТ" ведется работа по созданию отечественного суперкомпьютера стратегического назначения (СКСН) «Ангара» с мультитредово-потоковой архитектурой и глобально адресуемой памятью. СКСН «Ангара» предназначен для эффективного решения задач DIS-класса (Data Intensive System), для которых характерен интенсивный нерегулярный доступ к памяти большого объема (от сотен терабайт до нескольких петабайт).

Разрабатываемый СКСН «Ангара» ориентирован на использование массового тредового параллелизма, то есть в задачах должно порождаться, синхронизироваться и завершаться огромное количество программных тредов. В целом по всей системе их количество может достигать нескольких десятков-сотен тысяч. Очевидно, что для управлением таким количеством тредов необходимы эффективные и удобные средства.

В данной работе исследуются два подхода к реализации программных тредов для СКСН «Ангара». Первый подход основан на использовании программных тредов в соответствии с требованиями стандарта POSIX. Для реализации этого подхода была разработана библиотека программных тредов jthreads. Интерфейс jthreads является подмножеством стандартного интерфейса pthreads, но также в jthreads введены и некоторые расширения.

Программные треды в jthreads порождаются на свободных тредовых устройствах и могут выполняться только в данном ядре микропроцессора. Максимальное количество программных тредов ограничено количеством тредовых устройств, выделенных операционной системой для задачи. При создании нового программного треда ему выделяется память для фрейма и стека. Фрейм программного треда содержит необходимую для библиотеки информацию о треде - идентификатор треда, тип треда, размер стека треда и др. Для стеков программных тредов используется специальный сегмент данных.

Второй подход основан на применении сверхлегких безстековых программных тредов. Для реализации этого подхода была разработана библиотека lwt (light-weight threads). В отличие от тредов в jthreads, треды в lwt не обладают собственным программным стеком, то есть все данные хранятся на регистрах тредовых устройств. Кроме того треды в lwt имеют доступ к стеку породившего их треда. Такое свойство делает треды в lwt значительно более эффективными, но вместе с тем и менее функциональными по сравнению с тредами в jthreads.

Библиотека lwt предоставляет набор макроопераций для объявления функций для легких тредов, управления размещением локальных переменных, а также создания и синхронизации легких тредов. Легкие треды разделяют стек породившего их треда. Тредом-родителем может быть либо тред, выполняющий функцию main, либо тред, созданный с помощью jthreads. На легкие треды накладывается ряд ограничений: во-первых, легкие треды не могут вызывать функции, во-вторых, легкие треды не могут изменять значения локальных переменных, расположенных на стеке. Также как и в библиотеке jthreads количество программных тредов в lwt ограничено количеством тредовых устройств.

Для сравнения эффективности двух библиотек был выбран тест "сложение векторов". Результаты, полученные на имитационной модели СКСН «Ангара», показали, что при мелкозернистом параллелизме эффективность применения lwt значительно выше, чем jthreads. При увеличении работы, приходящейся на один программный тред, эффективность jthreads растет и в итоге достигает эффективности lwt. Нужно понимать, что за высокую эффективность библиотеки lwt приходится расплачиваться ограничениями, накладываемыми на легкие треды, и более сложной организаций мультитредовой программы.

В будущем исследование производительности jthreads и lwt будет продолжено. Будут вестись работы по их доработке и оптимизации. Также будет рассмотрен еще один интерфейс для разработки мультитредовых программ - qthreads, разработанный в Национальной лаборатории Сандия для систем с массовым тредовым параллелизмом и распределенной общей памятью.