УДК 004.4

РЕАЛИЗАЦИЯ ПРИМИТИВОВ СИНХРОНИЗАЦИИ ДЛЯ ЛЁГКИХ ПОТОКОВ НА ЯЗЫКЕ C++ ДЛЯ NUMA-АРХИТЕКТУРЫ

Скаженик Т.М. (ИТМО)
Научный руководитель – PhD, науки Аксенов В.Е. (ИТМО)

Введение. Многолетняя эволюция индустрии вычислительных устройств привела к смещению вектора развития чипов с увеличения производительности одного ядра к эффективной организации многоядерных и многопроцессорных систем. Таким образом, помимо классической архитектуры с однородным доступом к памяти (UMA, Uniform Memory Access), значительное место заняли системы с неравномерным доступом к памяти (NUMA, Non-Uniform Memory Access). В отличие от UMA, где доступ к памяти занимает равное время для любого чипа, в NUMA-архитектуре память можно разделить на два типа локальную и удаленную с существенно разным временем доступа.

На сегодняшний день для эффективного использования аппаратных ресурсов большая доля программного обеспечения выполняет свои вычисления параллельно. Поток – одна из основных абстракций, используемых для этой цели. Долгое время в языках программирования использовались только тяжелые потоки уровня операционной системы. Но в последние годы стали набирать популярность легкие потоки, существующие в пространстве пользователя. Как правило, множество легких потоков выполняется в рамках одного тяжелого, что позволяет достичь малых затрат на переключение контекста между выполняемыми задачами.

Стандартной задачей многопоточного программирования является обеспечение эксклюзивного доступа к разделяемым ресурсам. Решение может быть достигнуто использованием примитива синхронизации – мьютекса. В научных статьях [1-3] можно найти различные варианты реализации примитива для NUMA-архитектуры в условиях использования тяжелых потоков. Но концепция легких потоков до сих не представлена в стандарте языка C++, хотя существует набор библиотек, предлагающих свои реализации, Argobots [4] наиболее мощная и известная из них. Ввиду отсутствия единой реализации легких потоков возможность эффективного применения существующих алгоритмов синхронизации в окружении легких потоков остается неизученным вопросом. Адаптация классических алгоритмов синхронизации и создание новых может ускорить работу проектов с открытым исходным кодом, использующих экосистему Argobots, а также подготовить фундамент для использования получившихся примитивов совместно с другими реализациями легких потоков.

Основная часть. Для получения корректных и воспроизводимых результатов необходимо перед выполнением исследования зафиксировать целевую аппаратную платформу и разработать подход к оценке эффективности получившихся результатов.

В качестве тестовых машин было выбрано семейство ARM процессоров Kunpeng-920, построенное на базе двухуровневой и трехуровневой NUMA-архитектуры.

Первым шагом исследования стала разработка собственного бенчмарка для оценки эффективности создаваемых примитивов синхронизации в условиях произвольного типа потока. Адаптация существующих решений затруднялась необходимостью тестировать код, работающий в окружении легких потоков. В качестве базовых метрик были выбраны пропускная способность и задержка получения блокировки. Позднее было добавлено среднеквадратичное отклонение пропускной способности легкого потока. Такая метрика потребовалась, поскольку при использовании нечестных примитивов какие-то из легких потоков могут получать мало процессорного времени, что говорит о плохой кооперативности примитива синхронизации.

На втором этапе были реализованы наиболее эффективные с точки зрения производительности на тяжелых потоках примитивы синхронизации, например, MCS [1], HMCS [2], CNA [3]. Графики полученных результатов на тяжелых потоках совпали с

выводами из научных статей, что с одной стороны показало корректность созданного бенчмарка, с другой – подтвердило необходимый уровень эффективности реализованных примитивов.

На третьем этапе была проведена адаптация написанных алгоритмов к работе в окружении легких потоков, а также создание на их основе новых модификаций. Для доработки примитивов потребовалось внедрить механизм трассировки исполнения легких потоков в бенчмарк. Это позволило выявить негативные особенности поведения примитивов и предложить способы их устранения.

Заключительное измерение производительности полученных примитивов синхронизации в различных сценариях использования показало, что наивысшими по совокупности метрик результатами обладают иерархические примитивы на базе MCS.

Выводы. В рамках работы были реализованы примитивы синхронизации, учитывающие NUMA-архитектуру памяти процессора и адаптированные к применению в окружении легких потоков экосистемы Argobots. Для оценки эффективности примитивов был разработан бенчмарк с возможностью трассировки легких потоков. Результаты показали, что некоторые модификации примитивов на базе MCS показывают более высокую производительность по сравнению со стандартным решением из Argobots и базовыми реализациями классических алгоритмов. Полученные примитивы могут быть использованы в проектах на базе Argobots и в дальнейших исследованиях в области легких потоков.

Список использованных источников:

1. Mellor-Crummey J. M., Scott M. L. Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors // ACM Trans. Comput. Syst. — New York, NY, USA, 1991. — Февр. — С. 21–65.
2. Chabbi M., Fagan M., Mellor-Crummey J. High Performance Locks for Multi-Level NUMA Systems // SIGPLAN Not. — New York, NY, USA, 2015. — Янв. — Т. 50, № 8. — С. 215–226.
3. Dice D., Kogan A. Compact NUMA-Aware Locks // Proceedings of the Fourteenth EuroSys Conference 2019. — Dresden, Germany : Association for Computing Machinery, 2019. — (EuroSys ’19).
4. Argobots: A Lightweight Low-Level Threading and Tasking Framework / S. Seo [и др.] // IEEE Transactions on Parallel and Distributed Systems. — 2018. — Т. 29, No 3. — С. 512–526