ПРОГРАММНЫЙ КОМПЛЕКС S-MPI. РЕАЛИЗАЦИЯ НЕБЛОКИРУЮЩЕГО ВВОДА-ВЫВОДА

Д.В. Нагорный, Н.М. Леонова

В рамках реализации отечественной библиотеки S-MPI [1], в качестве кодовой базы которой использовалась реализация стандарта MPI-2[2-3] с открытым кодом OpenMPI [4], были разработаны асинхронные операции ввода-вывода, позволяющие полноценно использовать неблокирующий интерфейс файлового ввода-вывода стандарта MPI. В качестве основы при этом использовался пакет ROMIO[5].

Архитектура пакета ROMIO позволяет легко заменить низкоуровневые базовые функции ввода-вывода новыми, что и было сделано в реализации для S-MPI. Был разработан набор новых базовых функций, реализующий асинхронный ввод-вывод для работы с непрерывными типами данных MPI, который обеспечил полноценное использование возможностей ядра Linux в части асинхронного ввода-вывода.

Асинхронный ввод-вывод в ядре Linux представлен набором функций io_submit, io_cancel, io_destroy, io_getevents и io_setup[6].

Для реализации требуемого стандартом MPI-2 поведения после инициации операции ввода-вывода создаётся объект типа “запрос” (MPI_Request), идентификатор которого возвращается в пользовательскую программу. Опрос состояния инициированных операций ввода-вывода и выставление признаков завершения операций, связанных с соответствующими объектами типа “запрос”, производится в основном цикле библиотеки MPI, который называется в реализациях MPI - progress engine.

В ходе выполнения основного цикла библиотеки S-MPI (как и библиотеки OpenMPI, на которой она основана) выполняется основной объём операций по диспетчированию пересылки данных путём обращения к функциям обратного вызова, которые могут быть зарегистрированы различными подсистемами библиотеки. Так как основной цикл в реализациях MPI является наиболее критическим с точки зрения производительности, а файловые операции используются не всеми приложениями, регистрация функции опроса завершения инициированных операции ввода-вывода выполняется только при первом запросе на асинхронный ввод-вывод от приложения.

В целом, как показало тестирование производительности, разработанный механизм удовлетворяет требованиям к асинхронному вводу-выводу стандарта MPI и не оказывает негативного влияния на производительность других подсистем библиотеки.

Для оценки эффективности реализации асинхронного ввода-вывода использовались синтетические тесты, которые показали значительное преимущество реализации асинхронного ввода-вывода в библиотеке S-MPI по сравнению с его исходной реализации в OpenMPI. Так например время исполнения функции MPI_File_iread сократилось на блоках данных размером 1-2Мбайта более чем в 15 раз.

Отличие, достигнутое в библиотеке S-MPI за счет описанной реализации асинхронного ввода-вывода, и является декларируемым в стандарте ожидаемым поведением, недоступным в OpenMPI. Собственно время, которое оригинальная реализация в OpenMPI проводит в блокировках, может быть использовано в прикладных программах, скомпонованных с библиотекой S-MPI, для вычислений, что может обеспечить существенное повышение производительности приложений.

Реализация асинхронного ввода-вывода успешно продемонстрировала требуемую функциональность и создала основу для дальнейшего развития подсистемы ввода-вывода библиотеки S-MPI. Стоит отметить, что реализация неблокирующих операций ввода-вывода позволит не только оптимизировать пользовательские приложения, но может также обеспечить высокую эффективность ряда внутренних механизмов библиотеки. Так, например, для блокирующих операций ввода-вывода разреженных типов данных MPI время между инициацией операции ввода-вывода и её завершением может быть использовано для агрегации данных, что уменьшит время ввода-вывода данных такого типа. Другим интересным применением описанного механизма может оказаться непрерывное выполнение основного цикла библиотеки S-MPI даже при исполнении блокирующих операций ввода-вывода MPI. Для обеспечения данной возможности рассматривается реализация блокирующих операций ввода на основе асинхронных базовых операций.

ЛИТЕРАТУРА:

1. Г.И. Воронов, В.Д. Трущин, В.В. Шумилин, Д.В. Ежов. “Создание программного комплекса S-MPI для обеспечения разработки, оптимизации и выполнения высокопараллельных приложений на суперкомпьютерных кластерных и распределенных вычислительных системах”. // XIV международная конференция “Супервычисления и математическое моделирование” Саров октябрь 2012 тезисы докладов, [с.54-56]
2. Message Passing Interface Forum, “MPI: A Message Passing Interface”. // In Proceedings of Supercomputing ’93. IEEE Computer Society Press, November 1993, pp. 878–883.

3. Message Passing Interface Forum, “MPI-2: Extensions to the Message-Passing Interface”. // November 15, 2003, http://www.mpi-forum.org/docs/mpi2-report.pdf
4. Open MPI: Open Source High Performance Computing [Электронный ресурс] // URL: http://www.open-mpi.org/ (дата обращения: 17.05.2013).
5. ROMIO: A High-Performance, Portable MPI-IO Implementation [Электронный ресурс] // URL: http://www.mcs.anl.gov/research/projects/romio/ (дата обращения: 17.05.2013).
6. S. Bhattacharya, S. Pratt, B. Pulavarty, J. Morgan. “Asynchronous I/O Support in Linux 2.5” // Proceedings of the Linux Symposium, pp. 371--386, 2003