**Сравнительный анализ эффективности библиотек динамического выделения памяти**

И. А. Киреев, М. Г. Курносов[[1]](#footnote-2)

Проведено экспериментальное исследование алгоритмов глобальной редукции стандарта MPI на вычислительном кластере с сетью связи Gigabit Ethernet. Определены оптимальные алгоритмы глобальной редукции. Эксперименты проведены на вычислительном кластере с сетью связи Gigabit Ethernet.

*Ключевые слова*: глобальная редукция, allreduce, MPI, вычислительные системы.

1. Введение

Библиотеки динамического выделения памяти — инструменты для управления динамическим выделением памяти в соответствии с задачами приложения, в котором используются. Библиотеки динамического выделения памяти предоставляют эффективные алгоритмы контроля памяти, а также функции управления блоками памяти, её фрагментации для оптимизации производительности. Одними из наиболее распространённых библиотеками динамического выделения памяти являются: *ptmalloc2*, *jemalloc*, *tcmalloc*.

Цель исследования — выполнить сравнительный анализ эффективности библиотек динамического выделения памяти: *ptmalloc2*, *jemalloc*, *tcmalloc*.

2. Библиотеки динамического выделения памяти (различия?)

Основными пользовательскими функциями в библиотеках динамического выделения памяти являются: функция выделения (резервирования) памяти, функция освобождения памяти. В общем случае функция malloc выделения памяти резервирует блок памяти размера *n* и возвращает указатель на *void,* который указывает на начало блока зарезервированной памяти.

void \*malloc(size\_t n)

Основной функцией для очистки памяти является функция free*,* которая принимает указатель *ptr* на начало выделенной памяти и освобождает занимаемый блок

void free(void \*ptr)

Рассматриваемые библиотеки динамического выделения памяти имеют собственные механизмы контроля памяти. Каждая из исследуемых библиотек использует механизм выделения блоков памяти для каждого потока, что уменьшает вероятность возникновения конкуренции за общие ресурсы. Из рассматриваемых библиотек динамического выделения памяти, стандартной для большинства систем на базе Linux является ptmalloc2.

3. Результаты экспериментов

Эксперименты выполнены на вычислительной машине, который содержит процессор Ryzen 5 (6 ядер). Операционная система GNU/Linux. Размер страницы памяти – 4КБайт. Сравниваемые библиотеки: Jemalloc 5.3.0, Ptmalloc2, Tcmalloc 2.10. В качестве теста производительности использованы функции выделения и освобождения памяти для каждой библиотеки, а так же функция

void \*memset(void \*ptr, int value, size\_t num)

, которая принимает указатель *ptr,* который указывает на начало памяти, количество *num* элементов которой необходимо установить в значение *value*.

На рис. 1 показано время выполнения операции выделения 512Мбайт памяти в зависимости от количества потоков, а также количество тактов процессора. Библиотека tcmalloc показывает худшую производительность, так как время выполнения операции выделения динамической памяти из библиотеки tcmalloc линейно зависит от числа *p* потоков. Операция выделения памяти из библиотеки jemalloc показывает наименьшее время выполнения для выделения памяти.

Рекомендуется использовать библиотеку jemalloc в многопоточном режиме, так как она показывает лучшую масштабируемость, по сравнению с tcmalloc и ptmalloc2.

|  |  |
| --- | --- |
| а) | б) |

Рис. 1. Время выполнения и количество тактов процессора операций выделения памяти: а) время; б) количество тактов.

На рис. 2 показано количество свободной памяти в Мбайтах при выделении памяти размером 512 Мбайт блоками по 1 Кбайт, а также количество свободной памяти в Мбайтах при выделении памяти размером 512 Мбайт блоками по 4 Кбайта. На эксперименте выделения памяти блоками размера 4 Кбайт jemalloc показывает более стабильные результаты по сравнению с tcmalloc и ptmalloc2.

Для размеров блоков 4 Кбайт рекомендуется использовать библиотеку jemalloc, так как она показывает лучшую стабильность в выделении памяти блоками в сравнении с tcmalloc и ptmalloc2.

|  |  |
| --- | --- |
| а) | б) |

(80 процессов – 10 узлов по 8 ядер, тип MPI\_FLOAT, бинарная операция MPI\_SUM)

**4. Заключение**

Проведено экспериментальное исследование алгоритмов глобальной редукции стандарта MPI на вычислительном кластере с сетью связи стандарта Gigabit Ethernet и библиотекой Open MPI 5.0.0. На тестах OSU Mirco-Benchmarks показано, что при выполнении глобальной редукции для сообщений меньше 8 Кбайт рекомендуется использовать алгоритм *nonoverlapping* (Reduce + Bcast). В случае сообщений больше 8 Кбайт, наиболее эффективен алгоритм *segmented\_ring,* который выполняет автоматическую сегментацию сообщения.

**Литература**

1. MPI-4.0 Standard // URL: <https://www.mpi-forum.org/docs/mpi-4.0/mpi40-report.pdf> (дата обращения: 15.03.2022).
2. Open MPI // URL: <https://www.open-mpi.org/> (дата обращения: 15.03.2022).
3. OSU Micro-Benchmarks // URL: <https://mvapich.cse.ohio-state.edu/benchmarks/> (дата обращения 15.03.2022)
4. *Thakur R., Rabenseifner R., Gropp W.* Optimization of collective communication operations in MPICH // Int. Journal of High Performance Computing Applications. – 2005. – Vol. 19 (1). – P. 49‑66.
5. *Balaji P.* MPI on Millions of Cores / Balaji P., Buntinas D., Goodell D., Gropp W., Hoefler T., Kumar S., Lusk E., Thakur R., Traff J. // Parallel Processing Letters. – 2011. – Vol. 21, Issue 1. – P. 45-60.

**Киреев Илья Анатольевич**

студент группы ИВ-121 СибГУТИ (630102, Новосибирск, ул. Кирова, 86), тел. (383) 269-83-82, e-mail: ilya\_kireev\_2004[@bk.](mailto:rmashkalev054@gmail.com)ru.

**Курносов Михаил Георгиевич**

д.т.н., профессор, профессор кафедры вычислительных систем Сибирского государственного университета телекоммуникаций и информатики (630102, г. Новосибирск, ул. Кирова, 86), тел. (383) 269-83-82, e-mail: [mkurnosov@sibguti.ru](mailto:mkurnosov@sibguti.ru); старший научный сотрудник федерального государственного бюджетного учреждения науки Института физики полупроводников им. А.В. Ржанова Сибирского отделения Российской академии наук (630090, г. Новосибиpск, пp. Ак. Лавpентьева, 13), тел. (383) 330-56-26, e-mail: mkurnosov@isp.nsc.ru.

**Experimental Study of Allreduce Algorithms on a Multicore Cluster**

**R. Mashkalev, M. Kurnosov**

An experimental study of global reduction algorithms on a computing cluster has been carried out. Optimal algorithms for global reduction are determined. The obtained theoretical estimates are supported by the results of experiments on a computing cluster with a Gigabit Ethernet communication network.

*Keywords*: global reduction, allreduce, MPI, computing systems.

1. Исследования выполнены в рамках госзадания ИФП СО РАН (0242-2021-0011). [↑](#footnote-ref-2)