ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ

УДК 004.27
doi: 10.18101/2304-5728-2017-1-23-27

ОБ ОРГАНИЗАЦИИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ СРЕДСТВАМИ WINSOCK И MPI НА ЛОКАЛЬНОЙ СЕТИ

© Олзоева Сэсэг Ивановна
доктор технических наук, доцент
Восточно-Сибирский государственный университет технологий и управления
Россия, 670033, Улан-Удэ, ул. Ключевская, 40б
E-mail: sseseg@yandex.ru

© Михайлов Евгений Леонидович
аспирант
Восточно-Сибирский государственный университет технологий и управления
Россия, 670033, Улан-Удэ, ул. Ключевская, 40б
E-mail: evgmkhpost@gmail.com

Статья посвящена проблеме организации параллельных вычислений на компьютерной сети. Рассматриваются вопросы, связанные с выбором программных средств, обеспечивающих связь между вычислительными процессами, выполняющимися на разных компьютерах локальной сети. Приведены результаты исследований по сравнению производительностей программных средств обмена сообщениями между процессами: WinSock и MPI.
Ключевые слова: параллельные вычисления, программные средства обмена сообщениями между процессами, вычислительные кластеры.

Введение

В настоящее время наблюдается тенденция к использованию вычислительных кластеров, построенных на уже имеющихся локальных сетях, для решения сложных прикладных задач с крупномасштабным объемом вычислений, таких как глобальные задачи математического моделирования, обработки больших потоков данных в реальном времени, расчета динамических процессов в сложно устроенных средах и др.

В данной работе вычислительный кластер рассматривается как совокупность компьютеров, объединенных в рамках некоторой сети для решения одной задачи. Кластер представляет собой набор рабочих станций общего назначения, для связи узлов используется стандартная сетевая технология Fast Ethernet на базе коммутатора. Узлы кластера могут одновременно использоваться в качестве пользовательских рабочих станций. В качестве операционной системы используется стандартная Linux, вместе со специальными средствами поддержки параллельного программирования MPI (Message Passing Interface) и распределения нагрузки. Биб-

лиотеки MPI реализованы практически на всех современных суперкомпьютерах, а также могут использоваться и в кластерных системах и сетях (версия MPICH), т.е. пригодны для использования в различных системах с распределенной памятью.

Задача состояла в том, чтобы провести сравнительный анализ быстродействия параллельных программ, использующих для обмена сообщениями между вычислительными процессами библиотеку MPICH и интерфейс сокетов Windows Sockets, используя сеть как параллельный вычислитель.

1. Особенности разработанных параллельных программ

Для создания сетевых программ можно использовать интерфейс сокетов. Стандарт Windows Sockets (Winsock) позволяет разрабатывать сетевые приложения, способные работать с любым стеком TCP/IP. Winsock версии 2 способен работать и с другими транспортными протоколами. Существует мнение, что интерфейс сокетов – слишком низкоуровневый для большинства применений. Так, в [1] замечено, что написать и отладить программы, основанные непосредственно на интерфейсе сокетов, не легче, чем написать вручную и отладить программу в машинных кодах. Поэтому крупным достижением считалось создание и стандартизация интерфейса передачи сообщений MPI (Message Passing Interface).

Алгоритмы параллельного решения задач проектируются как системы параллельных и взаимодействующих между собой процессов, допускающих исполнение на независимых процессорах. Для проведения исследований, связанных с выбором программных средств для организации параллельных вычислений в компьютерной сети, была выбрана типовая задача линейной алгебры: решение систем линейных алгебраических уравнений большой размерности точными методами.

MPI-программа работает следующим образом: каждый узел считывает свою часть разбиения, результаты вычислений с каждого узла отправляются головному узлу для их обработки и вывода окончательного результата. При запуске MPI-программы создается коммуникатор по умолчанию MPI_COMM_WORLD, в котором задается число вычислительных параллельных процессов – p и передается каждой подпрограмме в качестве первого аргумента. Остальные аргументы определяют источник сообщения и буферы для хранения сообщений.

Каждый процесс считывает соответствующие строки системы уравнений из общего ресурса данных с шагом p. Первый процесс считывает первую строку, второй процесс – вторую, p-ую строку – p-ый процесс, затем p+1 строка будет считываться первым процессом, p+2 строка – вторым и т.д. Для определения номера процесса используется функция MPI_Comm_rank(MPI_COMM_WORLD,&r). Переменная r получает значение равное номеру процесса, вызвавшего эту функцию. Для определе-

ния количества процессов, которое задает пользователь, используется функция: MPI_Comm_size(MPI_COMM_WORLD,&p).

Приведем фрагменты программы для пересылки текущей строки:

```
for(k=0;k<p;k++)
                    {
                        if(k!=r)
MPI_Send(&array1[n1][0],n+1,MPI_FLOAT,k,1,MPI_COMM_WORLD);
                }
```

Здесь аргументами функции являются: адрес начала расположения пересылаемой строки; n+1 – число передаваемых элементов в текущей строке (n-размерность матрицы коэффициентов при неизвестных корнях системы уравнений); тип пересылаемых элементов; k – номер процесса получателя.

Получение текущей строки другими вычислительными процессами реализуется функцией:

```
MPI_Recv(&v[0],n+1, MPI_FLOAT,n1%p,1,MPI_COMM_WORLD,&st).
```

Аргументами являются: адрес начала расположения принимаемой строки; n+1 – число принимаемых элементов в строке; тип пересылаемых элементов; n1%p – значение при котором будет принято сообщение, если оно совпадет с тегом отправляемого сообщения; &st – атрибут принятого сообщения.

Winsock – программа использует тот же алгоритм решения системы алгебраических уравнений. Отличие в том, что необходимо каждому параллельному вычислительному процессу присвоить соответствующие значения переменных r и p. Технология Winsock основана на архитектуре «клиент - сервер». «Программа – клиент» начинает диалог первой. Поэтому параллельная программа спректирована так, что использует оба режима работы, как «программа – клиент» и как «программа – сервер». Для пересылки сообщения «программой – сервером» реализован следующий фрагмент:

```
for(perc=1;perc<p;perc++)
send(client_socket[perc],(char*)array1[n1],sizeof(array1[n1]),0);
```

Здесь аргументами функции являются – сокет программы, принимающей сообещние; начальный адрес массива передаваемых байтов; длина в байтах передаваемого массива. Получение текущей строки «программой – сервером» от других вычислительных процессов осуществляется с помощью функции:

```
recv(client_socket[(n1%p)],(char*)v,sizeof(v),0).
```

Аргументы функции: сокет программы, отправившей сообещние; остальные аналогичны предыдущей функции.

Отправка и получение сообщений «программой – клиентом» отличается от «программы – сервера» только аргументом my_sock – сокет программы, отправляющей и принимающей сообещние: send(my_sock,(char*)array1[n1],sizeof(array1[n1]),0) и recv(my_sock,(char*)v,sizeof(v),0);

В таблице 1 приведены значения времени, которые потребовались для решения системы уравнений размерностью 1000×1000. При вычислении использовалось 4 узла с двуядерными процессорами Intel® Core ™ i3-2120 CPU@3.30Hz.

Таблица 1. Результаты затраченного времени на выполнение программ

	1 процесс	2 процесса	4 процесса	8 процессов
Последовательная программа	3,034 сек.			
MPI		1,906 сек.	0,958 сек.	0,639 сек.
WinSock		1,428 сек.	0,718 сек.	0,479 сек.

Заключение

По результатам затраченного времени на выполнение программ можно сделать вывод, что при решении типовой вычислительной задачи технология WinSock имеет более высокую производительность, чем MPI.

При решении более сложных вычислительных задач для уменьшения накладных расходов при передаче данных между процессами лучше использовать низкоуровневые взаимодействия в сети на уровне сокетов. При всех достоинствах MPI, обеспечение удобного сервиса для прикладного программиста не совсем благоприятно сказывается на характеристиках эффективного исполнения функций обмена сообщениями, не способствует уменьшению времени организации обмена сообщениями между процессами, что особенно важно для организации параллельных вычислений на компьютерных сетях. Так как стандартные программные интерфейсы, как в стиле MPI, так и языковые в стиле стандарта HPF сами построены на базе эффективных низкоуровневых библиотек обмена сообщениями.

Литература

1. Арапов Д. Можно ли превратить сеть в суперкомпьютер? // Открытые системы. — 1997. — № 4.

2. Корнеев В. Д. Параллельное программирование в MPI: Учеб. пособие. — Ярославль: Изд-во Ярославского гос. ун-та, 2002. — 104 с.

ABOUT ORGANIZATION OF PARALLEL COMPUTING USING WINSOCK AND MPI ON A LOCAL NETWORK

Seseg I. Olzoeva
Dr. Sci., A/Prof.
East-Siberian State University of Technologies and Management
40b Klyuchevskaya St., Ulan-Ude 670033, Russia

Evgeniy L. Mikhailov
Research Assistant
East-Siberian State University of Technology and Management
40b Klyuchevskaya St., Ulan-Ude 670033, Russia

The article is devoted to problem of organization of parallel computing in computer network. We discuss the issues related to selection of software that provides communication between the computing processes running on different computers on the local network. The results of studies on comparison of performance of software messaging between the processes of WinSock and MPI are presented.
Keywords: parallel computing, tools for software messaging between the processes, computational clusters.