ТЕХНОЛОГИЯ ГИБРИДНЫХ ОБМЕНОВ СООБЩЕНИЯМИ НА БАЗЕ СТАНДАРТОВ MPI И OPENMP ДЛЯ КЛАСТЕРНЫХ СИСТЕМ

Е.В. Аксенова, М.Л. Цымблер

Введение. Кластер представляет собой трехуровневую иерархическую многопроцессорную систему. На первом уровне иерархии располагаются процессоры (в настоящее время, как правило, многоядерные). На втором уровне иерархии – SMP-модули. На третьем уровне SMP-модули объединяются в одну установку с помощью высокоскоростной сети. Скорость обменов данными между процессорами (процессорными ядрами) может существенно уменьшаться при движении снизу вверх по уровням иерархии системы. В соответствии с этим актуальной задачей становится разработка иерархических алгоритмов [1], существенным образом учитывающих гибридную структуру многопроцессорных иерархий. В данной работе рассматривается технология гибридных обменов сообщениями для кластеров [2], реализованная на базе стандартов MPI и OpenMP.

В настоящее время стандарт MPI является де-факто стандартом параллельного программирования для многопроцессорных систем с распределенной памятью. Стандарт OpenMP используется для разработки параллельных приложений в модели общей памяти. Данные стандарты могут быть использованы совместно в рамках создания параллельного приложения для проведения научно-инженерных расчетов на кластерной вычислительной системе [3, 4]. Нами предлагается использовать подход MPI+OpenMP для организации эффективных обменов сообщениями в однородных кластерах.

Гибридная технология передачи сообщений. Пусть имеется параллельное приложение, реализованное с помощью функций стандарта MPI. Мы предполагаем, что данное приложение выполняется на кластерной вычислительной системе в соответствии с моделью SPMD (Single-Program-Multiple-Data) в виде нескольких процессов. В соответствии с моделью SPMD параллельные процессы приложения должны иметь один и тот же исходный код. Такие процессы будем называть *MPI-процессами*.

Пусть в этом приложении имеются директивы стандарта OpenMP, создающие нити (например, `parallel`, `parallel for` и др.). В соответствии с этими директивами каждый MPI-процесс создает заданное количество нитей, которые будем называть *OpenMP-нитями.*

Гибридным процессом будем называть корневую нить MPI-процесса либо OpenMP-нить. Нумерация гибридных процессов начинается с 0, подобно нумерации MPI-процессов и OpenMP-нитей.

Два гибридных процесса будем называть *удаленными*, если их предками являются разные MPI-процессы. Два гибридных процесса будем называть *локальными*, если их предком является один и тот же MPI-процесс.

Технология гибридной передачи сообщений в однородных кластерных системах представлена на Рис. 1.

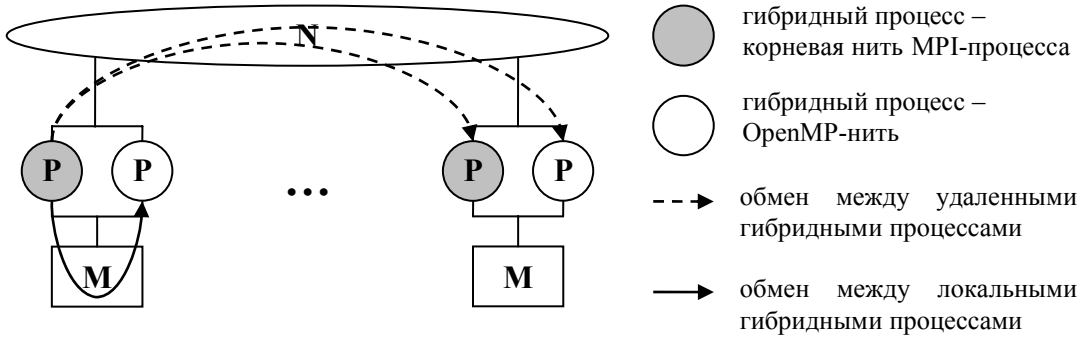

Рис. 1. Технология гибридной передачи сообщений для кластеров
(**N** – сеть, **P** – процессор, **M** – модуль памяти)

Передача сообщений между гибридными процессами основана на использовании различных методов реализации передачи сообщений между удаленными гибридными процессами и между локальными гибридными процессами. Передача сообщений между удаленными гибридными процессами осуществляется через соединительную сеть на базе коммуникационных функций стандарта MPI. Передача сообщений между локальными гибридными процессами осуществляется через общую память на базе директив стандарта OpenMP.

Данная технология реализована нами в виде библиотеки системных функций, названной Messaging Interface eXtension (MIX) [5]. Основные функции данной библиотеки представлены на Рис. 2. Их интерфейсы идеологически близки к интерфейсам функций библиотеки MPI.

Коммуникационные функции библиотеки MIX имеют следующую семантику. Возвращаемое значение `MIX_SUCCESS` означает, что выполнение соответствующей операции обмена завершено. В случае операции `MIX_Recv` это означает, что текущий процесс получил сообщение от процесса-отправителя. В случае операции `MIX_Send` это значит, что сообщение текущего процесса отправлено или сохранено в системном буфере. Значение `MIX_FAIL` означает, что в данный момент по каким-либо причинам операция обмена не может быть выполнена. В этом случае для осуществления операции обмена пользователь должен выполнить повторный вызов коммуникационной функции с теми же параметрами.

```
#define MIX_SUCCESS (0) // Успешное завершение функции.
#define MIX_FAIL (-1) // Неуспешное завершение функции.

// Инициализировать MIX-программу.
int MIX_Init(int* argc, char** argv[]);

// Завершить MIX-программу.
int MIX_Finalize();

// Количество гибридных процессов.
int MIX_Size(int* size);

// Номер текущего гибридного процесса.
int MIX_Rank(int* rank);

// Отправить сообщение.
int MIX_Send(void* buf, int length, int dest, int tag);

// Получить сообщение.
int MIX_Recv(void* buf, int length, int dest, int tag);
```

Рис. 2. Интерфейсы основных функций библиотеки MIX

На Рис. 3 приведен пример программы, в которой гибридные процессы обмениваются сообщениями по топологии "кольцо".

```
#include "mix.h"
void main(int argc, char *argv[])
{
  MIX_Init(&argc, &argv);
  #pragma omp parallel MIX_SHARED_DATA // запуск гибридного процесса
  {
    int sendbuf=1, recvbuf;
    int rank, size, i;
    MIX_Size(&size);
    MIX_Rank(&rank);
// обмены данными между гибридными процессами
    while (MIX_Send(&sendbuf, 4, (rank+1)%size, 0)!=MIX_SUCCESS);
    while (MIX_Recv(&recvbuf, 4, (size+rank-1)%size, 0)!=MIX_SUCCESS);
  }
  MIX_Finalize();
}
```

Рис. 3. Пример MIX-программы, реализующей обмены по топологии "кольцо"

Результаты экспериментов. Для исследования эффективности разработанной библиотеки гибридных обменов сообщениями нами были проведены вычислительные эксперименты на Высокопроизводительном вычислительном кластере Infinity (36 место в 6-й редакции списка TOP50 от 11.04.2007) [6]. Мы использовали свободно распространяемый в виде исходных текстов на языке C пакет тестов Intel MPI Benchmarks (IMB), который предназначен для тестирования производительности коммуникационной системы на уровне MPI, библиотеку MPICH версии 1.2.7 и компилятор Intel версии 9.1.

Тест Multi-PingPong, входящий в IMB, предполагает выполнение серий обменов между парами "соседних" процессов (0-й и 1-й, 2-й и 3-й и т.д.) и оценивает время и скорость обмена данными между двумя процессами. Мы рассмотрели следующие схемы запуска процессов: SE-схема, SN-схема и MIX-схема. *Схема SE (Shared Everything)* предполагает, что в каждой паре взаимодействующих процессов процессы запускаются на одном узле кластера и имеют возможность взаимодействовать через общую память. *Схема SN (Shared Nothing)* предполагает, что в каждой паре взаимодействующих процессов процессы запускаются на различных узлах кластера и могут взаимодействовать только через соединительную сеть. Обмены данными в схемах SE и SN реализуются на базе коммуникационных функций MPI типа "точка-точка". *Схема MIX* предполагает, что взаимодействующие процессы являются гибридными и обмениваются сообщениями с помощью функций библиотеки MIX, реализованных на базе стандартов MPI и OpenMP.

Результаты экспериментов на тесте Multi-PingPong представлены на Рис. 4.

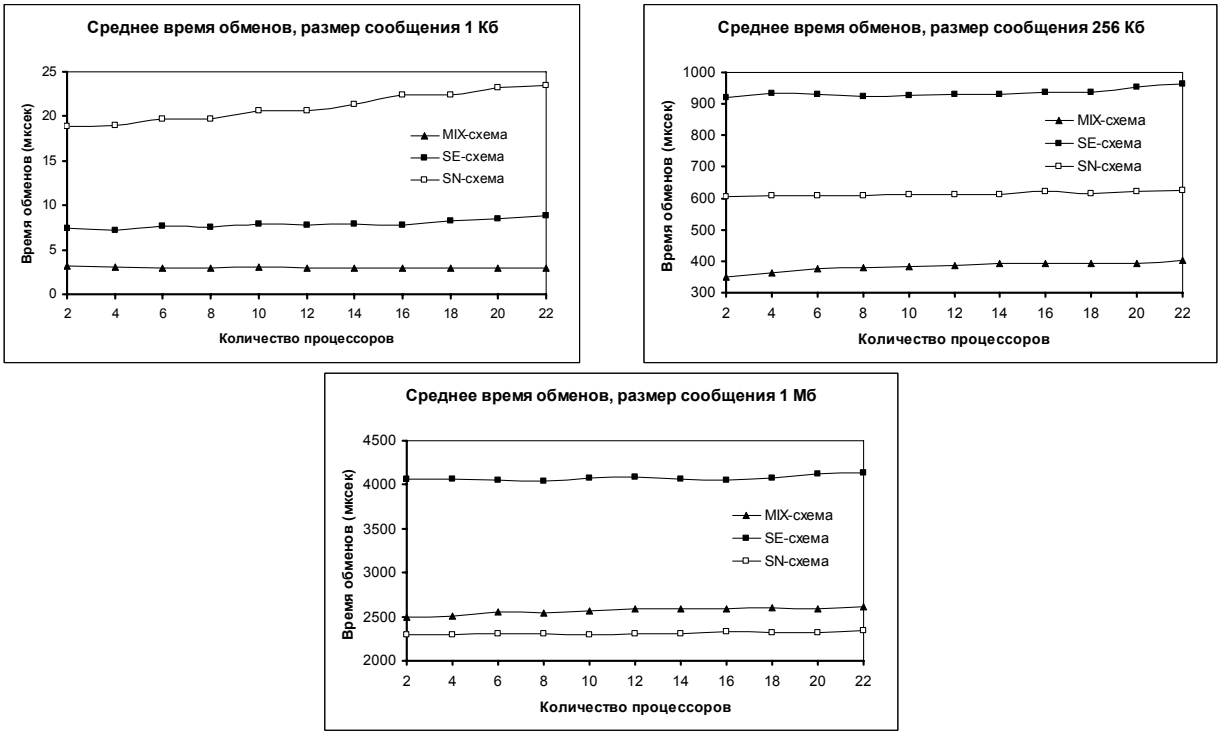

Рис. 4. Результаты экспериментов на тесте Multi-PingPong пакета IMB

Как можно видеть из графиков, при небольших размерах сообщения (до 256 Кб) MIX-схема демонстрирует лучшие результаты, чем SE-схема и SN-схема. Однако при увеличении размера сообщения до 1 Мб первенство переходит к SN-схеме. Интересно отметить, что при размере сообщения до 256 Кб в случае SE-схемы результаты теста лучше, чем у SN-схемы, а при увеличении размера сообщения мы получаем обратную картину. По-видимому, это объясняется особенностью реализации обменов в используемой нами библиотеке, когда алгоритм обмена выбирается в зависимости от размера сообщения. В качестве направления будущих исследований мы планируем проведение большего количества экспериментов, в том числе для других тестов пакета IMB и других тестовых пакетов, а также доработку реализацию библиотеки MIX для увеличения ее эффективности.

Заключение. В работе описана гибридная технология передачи сообщений для кластерных вычислительных систем, основанная на совместном использовании стандартов MPI и OpenMP. Данная технология реализована в виде библиотеки системных функций и представлены интерфейсы данных функций. Описаны результаты экспериментов, подтверждающие эффективность предложенной технологии.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект 06-07-89148), Фонда содействия развитию малых форм предприятий в научно-технической сфере (проект 7434) и Правительства Челябинской области (проект 080.07.06-07.АХ).

ЛИТЕРАТУРА:

1. Соколинский Л.Б. Иерархический параллелизм: новая парадигма программирования // Информационный бюллетень Ассоциации математического программирования. -№ 11. -Екатеринбург: УрО РАН. -2007. -С. 76-77.
2. Аксенова Е.В., Цымблер М.Л. Использование технологий MPI и OpenMP для организации обменов сообщениями в вычислительных системах с кластерной архитектурой // Параллельные вычислительные технологии: Труды международной научной конференции (29 января - 2 февраля 2007 г., г. Челябинск). -Челябинск: Изд-во ЮУрГУ. - 2007. -Т. 2. -С. 282.
3. Крюков В.А. Разработка параллельных программ для вычислительных кластеров и сетей // Информационные технологии и вычислительные системы. -2003. -Вып. 1-2. -С. 42-61.
4. Smith L., Bull M. Development of mixed mode MPI / OpenMP applications // Scientific Programming. -No. 9 (2-3), 2001. -pp. 83-98.
5. Аксенова Е.В., Цымблер М.Л. Библиотека гибридной передачи сообщений. Технич. отчет. -Челябинск: ЮУрГУ, 2007. -12 с. [http://foreva.susu.ru/science/treport/TR_MIX.pdf]
6. Техническая информация и характеристики кластера Infinity [http://cluster.susu.ru/information/]