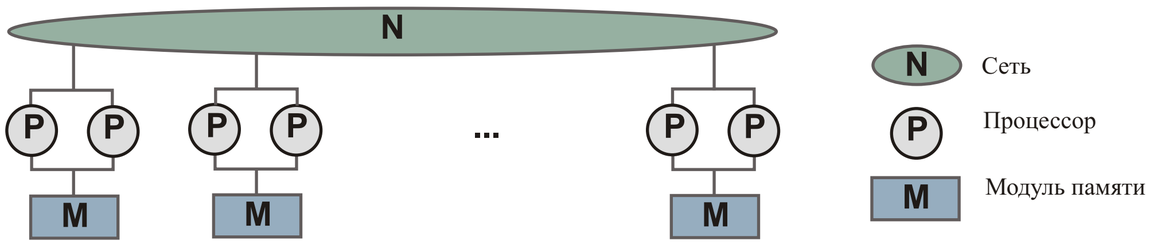
Использование технологий MPI и OpenMP для организации обменов сообщениями в вычислительных системах с кластерной архитектурой

Е.В. Аксенова, М.Л. Цымблер
evaksen@mail.ru, mzym@susu.ru

Южно-Уральский государственный университет

ВЫПОЛНЕНИЕ ПРИЛОЖЕНИЙ НА КЛАСТЕРЕ

Кластер Infinity ЮУрГУ



Запуск приложения на кластере

Схема запуска Shared-Everything (SE)

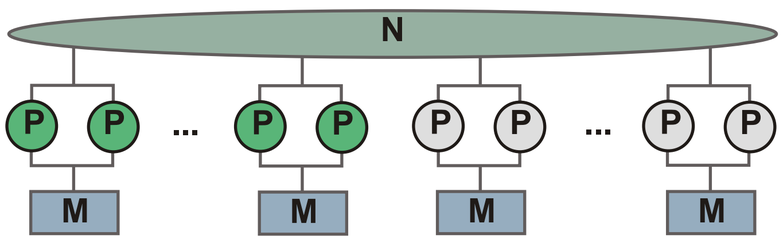
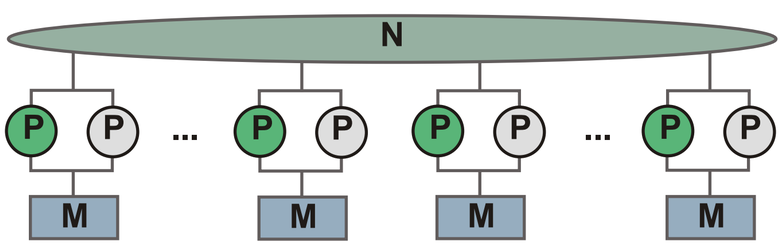


Схема запуска Shared-Nothing (SN)

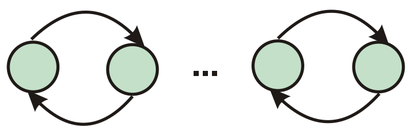


P Процессор с запущенным на нем процессом

P Свободный процессор

ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ ПЕРЕДАЧИ СООБЩЕНИЙ

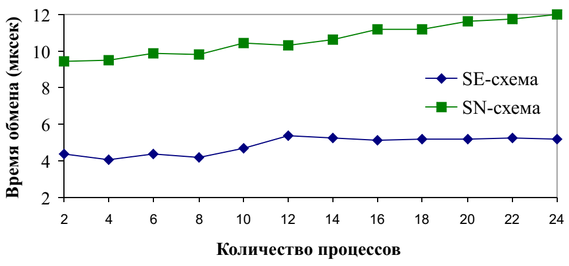
Тест Multi-PingPong (пакет Intel ® MPI Benchmarks)



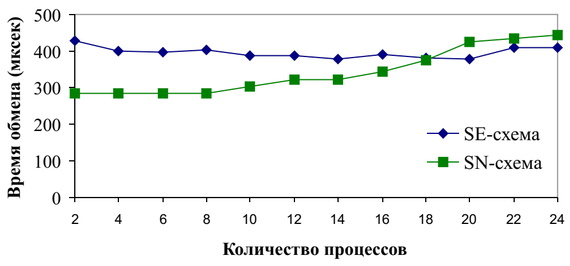
Процесс

Пересылка сообщения

Среднее время обмена (размер сообщения 1 Kb)

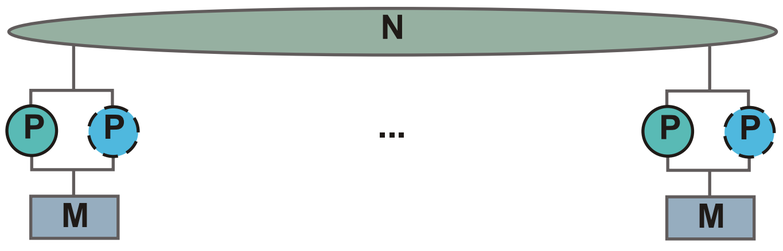


Среднее время обмена (размер сообщения 256 Kb)



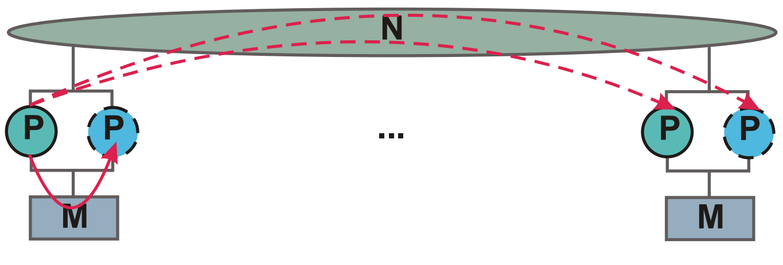
ГИБРИДНАЯ ПЕРЕДАЧА СООБЩЕНИЙ

Запуск гибридного приложения на кластере



Приложение запускается на одном процессоре каждого из узлов кластера. Наличие в исходном тексте программы директивы стандарта OpenMP `#pragma omp parallel` влечет за собой автоматическое создание нитей на оставшихся процессорах узла кластера. Приложение выполняется на кластере в виде MPI-процессов и OpenMP-нитей, которые могут рассматриваться как процессы исходного параллельного приложения.

Технология гибридной передачи сообщений



Передача сообщений между процессами, которые принадлежат одному узлу кластера, реализуется на базе директив стандарта OpenMP. Передача сообщений между процессами, которые принадлежат различным узлам кластера, реализуется на базе коммуникационных функций стандарта MPI.

БИБЛИОТЕКА MIX (MESSAGING INTERFACE EXTENSION)

MIX-процесс

- MPI-процесс (порождающий OpenMP-нити на всех процессорах узла)
- OpenMP-нить

Обмен сообщениями между MIX-процессами

Отправить сообщение

```
int MIX_Send(void*, int, int, int);

// void* buf - буфер с отправляемым сообщением
// int length - длина сообщения
// int dest - номер MIX-процесса получателя сообщения
// int tag - тег сообщения

// Результат:
// MIX_SUCCESS - операция выполнена
// MIX_FAILURE - операция не выполнена
```

Получить сообщение

```
int MIX_Recv(void*, int, int, int);

// void* buf - буфер для получаемого сообщения
// int length - длина сообщения
// int src - номер MIX-процесса отправителя сообщения
// int tag - тег сообщения

// Результат:
// MIX_SUCCESS - операция выполнена
// MIX_FAILURE - операция не выполнена
```

Сравнение коммуникационных функций MPI и MIX

Характеристика	MPI		MIX
	блокирующие	неблокирующие	
Приостановка процесса до завершения операции	+	–	–
Повторное использование буфера с сообщением до завершения операции	+	–	+

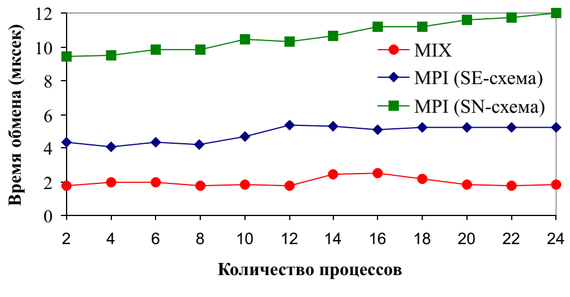
Пример MIX-программы

```
/* Обмен сообщениями между MIX-процессами по схеме "мастер - рабочий" */

#include "mix.h"
#include <stdio.h>

int main(int argc, char *argv[])
{
  // Инициализация MIX-процесса
  MIX_Init(&argc, &argv);
  #pragma omp parallel MIX_BLOCK
  {
    // Объявление локальных переменных
    int sendbuf, recvbuf;
    int rank, size, i;

    MIX_Size(&size);
    MIX_Rank(&rank);
    if (rank == 0)
      for (i=1; i<size; i++){
        sendbuf = i;
        while (MIX_Send(&sendbuf, sizeof(int), i, 10) != MIX_SUCCESS);
        printf("[%d]: Сообщение отправлено.\n", rank);
      }
    else{
      while (MIX_Recv(&recvbuf, sizeof(int), 0, 10) != MIX_SUCCESS);
      printf("[%d]: Сообщение получено.\n", rank);
    }
  }
  // Завершение MIX-процесса
  MIX_Finalize();
  return 0;
}
```

РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Тест Multi-PingPong

Среднее время обмена на тесте (размер сообщения 1 Kb)



Среднее время обмена на тесте (размер сообщения 256 Kb)