ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»  
(СИБГУТИ)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине

“Структуры и алгоритмы обработки данных ”

на тему

ИССЛЕДОВАНИЕ АЛГОРИТМОВ КОЛЛЕКТИВНЫХ ОБМЕНОВ ИНФОРМАЦИЕЙ МЕЖДУ ВЕТВЯМИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ  
В РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

|  |  |
| --- | --- |
| Выполнил студент |  |
|  | Ф.И.О. |

|  |  |
| --- | --- |
| Группы |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работу принял |  | доцент к.т.н. О.В. Молдованова |
|  | подпись |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Оценка |  |  |  |
|  |  |  |  |

Новосибирск – 2017

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc461027512)

[1 Алгоритмы трансляционно-циклических обменов информацией в распределенных ВС 5](#_Toc461027513)

[1.1 Трансляционно-циклический обмен информацией в распределенных ВС 5](#_Toc461027514)

[1.2 Алгоритм Bruck 6](#_Toc461027515)

[2 Экспериментальное исследование эффективности алгоритма 8](#_Toc461027516)

[2.1 Организация моделирование 8](#_Toc461027517)

[2.2 Результаты моделирования 8](#_Toc461027518)

[ЗАКЛЮЧЕНИЕ 9](#_Toc461027519)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 10](#_Toc461027520)

[ПРИЛОЖЕНИЕ 11](#_Toc461027521)

[1 Исходный текст программы 12](#_Toc461027522)

ВВЕДЕНИЕ

Начиная с середины 1960-х годов распределенные вычислительные системы (ВС) активно используются как инструментальные средства решения сложных задач в различных отраслях науки и техники. В архитектурном плане распределенная ВС представляется композицией множества взаимодействующих элементарных машин (оснащенных средствами коммуникаций и внешними устройствами) и сети межмашинных связей [1, 2]. Элементарная машина (ЭМ) – это основной функциональный и структурный элемент ВС; конфигурация ЭМ допускает варьирование в широких пределах – от процессорного ядра до ЭВМ. Все основные ресурсы распределенных ВС (арифметико-логические устройства, память, средства управления и коммуникаций) являются логически и технически рассредоточенными. Число ЭМ в распределённых ВС допускает варьирование от нескольких единиц до сотен тысяч.

Параллельные алгоритмы и программы для распределенных ВС преимущественно разрабатываются в модели передачи сообщений (Message Passing). В этой модели ветви параллельной программы взаимодействуют друг с другом путем обменов информационными сообщениями по каналам межмашинных связей ВС. Выделяют два типа обменов информацией между ветвями параллельных программ [1]: дифференцированный (point-to-point) и коллективные (collective) обмены. При дифференцированном обмене осуществляется передача информации из одной ветви в любую другую ветвь. Коллективные обмены подразделяются на несколько видов: трансляционный (ТО, One-to-all Broadcast), трансляционно-циклический (ТЦО, All-to-all Broadcast) и коллекторный обмены (КО, All-to-one broadcast). При трансляционном обмене данные из одной ветви передаются во все остальные; при трансляционно-циклическом обмене информация из ветвей передается каждой ветви и принимается из всех; коллекторный обмен подразумевает прием информации из всех ветвей в одну.

Анализ использования в параллельных алгоритмах и программах коллективных обменов показывает, что суммарное время их выполнения достигает 45% времени выполнения программ [3]. Как правило, количество обращений в алгоритмах и программах к коллективным операциям обменов имеет функциональную зависимость от размера входных данных и в среднем находится в интервале от до [4].

В настоящее время в коммуникационных библиотеках стандарта MPI и системах параллельного программирования (в частности в модели PGAS – Partitioned Global Address Space) для реализации коллективных обменов используются алгоритмы рассылки данных по кольцу, рекурсивного сдваивания, алгоритм Дж. Брука (J. Bruck) и алгоритмы, упорядочивающие ветви в деревья различных видов [5, 6]. Перечисленные алгоритмы характеризуются различным временем выполнения и опираются на предположение об однородности каналов связи между ЭМ распределенных ВС.

Цель работы – исследовать эффективность реализации заданного алгоритма коллективного обмена информацией между ветвями параллельных программ в распределенных ВС.

1 Алгоритмы трансляционно-циклических обменов информацией в распределенных ВС

1.1 Трансляционно-циклический обмен информацией в распределенных ВС

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

Формулы следует набирать в Microsoft Equation, например, так . Если на формулы есть ссылка в тексте, то формула набирается по центру.

|  |  |
| --- | --- |
|  | (1.1) |

Рисунки по центру.



Рисунок 1.1 – Пример иерархической организации  
коммуникационной среды кластерной ВС

На рис. 1.1 приведен пример иерархической организации коммуникационной среды кластера.



Рисунок 1.2 – Пример иерархической организации  
коммуникационной среды вычислительного кластера:  
три вычислительных узла на базе 2 x AMD Opteron 275;  
*N* = 12; *L* = 3; *n*23 = 2; *C*23 = {9, 10, 11, 12}; *с*23 = 4; *g*(3, 3, 4) = 2; *z*(1, 7) = 1

На рис. 1.2 приведен комплексный пример.

Таблицы оформлять так.

Таблица 5.1 – Результаты экспериментов

|  |  |  |
| --- | --- | --- |
| Граф | Время работы пакета Maple 11, с | Время работа созданного алгоритма, с |
| *C*1 | 2,8 | < 0,01 |
| *C*2 | 14,3 | < 0,01 |
| *C*3 | 32,4 | 0,01 |
| *G*1 | 3,0 | 0,02 |
| *G*2 | 1123,0 | 0,50 |
| *G*3 | > 24 часов | 507,2 |

1.2 Алгоритм Bruck

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

1.3 Выводы

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

2 Экспериментальное исследование эффективности алгоритма

2.1 Организация моделирование

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

2.2 Результаты моделирования

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст. Текст.

ЗАКЛЮЧЕНИЕ

В результате выполнения работы разработан и исследован алгоритм …

Осуществлено моделирование разработанного алгоритма. Показано, что …

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хорошевский В.Г. Архитектура вычислительных систем. – М.: МГТУ им. Н.Э. Баумана,  
   2008. – 520 с.
2. Евреинов Э.В., Хорошевский В.Г. Однородные вычислительные системы. – Новосибирск: Наука, 1978. – 320 с.
3. Rabenseifner R.. Automatic MPI Counter Profiling // Proceedings of the 42nd Cray User Group. – Noorwijk, The Netherlands, 2000. – 19 pp.
4. Han D., Jones T.. MPI Profiling // Technical Report UCRL-MI-209658 – Lawrence Livermore National Laboratory, USA, 2004. – 15 pp.
5. Thakur R., Rabenseifner R., and Gropp W. Optimization of collective communication operations in MPICH // Int. Journal of High Performance Computing Applications. – 2005. – Vol. 19, No. 1. – P. 49‑66.
6. Balaji P., Buntinas D., Goodell D., Gropp W., Kumar S., Lusk E., Thakur R. and Traff J. L. MPI on a Million Processors // Proc. of the PVM/MPI – Berlin: Springer-Verlag, 2009. – P. 20‑30.
7. Khoroshevsky V., Kurnosov M. Mapping Parallel Programs into Hierarchical Distributed Computer Systems // Proc. of “Software and Data Technologies”.  Sofia: INSTICC, 2009.  Vol. 2.  P. 123‑128.

ПРИЛОЖЕНИЕ

1 Исходный текст программы

/\*

\* pi\_mpi.c: MPI simple integration example.

\*

\*/

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include "mpi.h"

int main(int argc, char \*argv[])

{

double PI25DT = 3.141592653589793238462643;

int i, rank, commsize;

double nsteps, step, local\_pi, pi, sum, x;

double time = 0.0;

int namelen;

char processor\_name[MPI\_MAX\_PROCESSOR\_NAME];

MPI\_Init(&argc,&argv);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &commsize);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Get\_processor\_name(processor\_name, &namelen);

fprintf(stdout,"Process %d of %d is on %s\n",

rank, commsize, processor\_name);

fflush(stdout);

nsteps = (argc > 1) ? atoi(argv[1]) : 100000;

if (rank == 0)

time -= MPI\_Wtime();

MPI\_Bcast(&nsteps, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

step = 1.0 / (double)nsteps;

sum = 0.0;

for (i = rank + 1; i <= nsteps; i += commsize)

{

x = ((double)i - 0.5) \* step;

sum += 4.0 / (1.0 + x \* x);

}

local\_pi = nsteps \* sum;

MPI\_Reduce(&local\_pi, &pi, 1, MPI\_DOUBLE, MPI\_SUM, 0,

MPI\_COMM\_WORLD);

if (rank == 0) {

time += MPI\_Wtime();

printf("PI is approximately %.16f, Error is %.16f\n",

pi, fabs(pi - PI25DT));

printf("(nsteps = %d, step = %f)\n", nsteps, step);

printf("Elapsed time = %.4f sec.\n", time);

fflush(stdout);

}

MPI\_Finalize();

return 0;

}