Национальный технический университет Украины

«Киевский политехнический институт»

Лабораторная работа №5

По курсу

Параллельные и распределенные вычисления

Тема:

Java. Мониторы

Работу выполнил

Студент 3-го курса ФИВТ

Группы ИВ-73

Грубый Павел

Киев-2010

Техническое задание



MA = a×(MB×(MZ + MO×MF)

Этап 1. Построение параллельного алгоритма

MAH = a×(MB×(MZH + MO×MFH)

ОР: a, MB, MO

Этап 2. Разработка алгоритмов работы каждого процесса

|  |  |
| --- | --- |
| **Задача Т1** | Критические участки |
| 1. Ввод a, MB, MZ, MO, MF. |  |
| 1. Сигнал задачам Т2, T3, T4 о завершении ввода a, MB, MZ, MO, MF. |  |
| 1. Копия a1 := a | КУ |
| 1. Копия MB1 := MB | КУ |
| 1. Копия MO1 := MO | КУ |
| 1. Вычисление MAH = a1×(MB1×(MZH + MO1×MFH) |  |
| 1. Сигнал задачe Т3 о завершении вычислений. |  |

|  |  |
| --- | --- |
| **Задача Т2** | Критические участки |
| 1. Ожидание ввода a, MB, MZ, MO, MF в задаче Т1. |  |
| 1. Копия a2 := a | КУ |
| 1. Копия MB2 := MB | КУ |
| 1. Копия MO2 := MO | КУ |
| 1. Вычисление MAH = a2×(MB2×(MZH + MO2×MFH) |  |
| 1. Сигнал задачe Т3 о завершении вычислений. |  |

|  |  |
| --- | --- |
| **Задача Т3** | Критические участки |
| 1. Ожидание ввода a, MB, MZ, MO, MF в задаче Т1. |  |
| 1. Копия a3 := a | КУ |
| 1. Копия MB3 := MB | КУ |
| 1. Копия MO3 := MO | КУ |
| 1. Вычисление MAH = a3×(MB3×(MZH + MO3×MFH) |  |
| 1. Ожидание сигнала от задач Т1, Т2, Т4 о завершении вычислений. |  |
| 1. Вывод MA. |  |

|  |  |
| --- | --- |
| **Задача Т4** | Критические участки |
| 1. Ожидание ввода a, MB, MZ, MO, MF в задаче Т1. |  |
| 1. Копия a4 := a | КУ |
| 1. Копия MB4 := MB | КУ |
| 1. Копия MO4 := MO | КУ |
| 1. Вычисление MAH = a4×(MB4×(MZH + MO4×MFH) |  |
| 1. Сигнал задачe Т3 о завершении вычислений. |  |

Этап 3. Разработка структурной схемы взаимодействия задач



рис. 3.1. Схема взаимодействия задач.

Сравнение времени выполнения программной реализации алгоритма на языках Ada и Java

В результате тестирования программ время выполнения реализации на языке Ада для матриц размерности 1000х1000 показало 55 секунд, а для Java реализации – 22 секунды, что более чем в 2 раза быстрее.

Этап 4. Разработка программы

**import** java.util.Calendar;

/\*\*

\* Java Monitors. Lab 5

\* **@category** Paralel and distributed computing

\* **@author** Grubyi Pavel IO-73

\* **@since** 31.03.2010

\*/

**public** **class** CalcSync {

**private** **static** **int** *F* = 0;

**public** **synchronized** **void** calcSignal() {

*F*++;

**if** (*F* >= 3) notifyAll();

}

**public** **synchronized** **void** waitForCalc() {

**try** {

**if** (*F* < 3)

wait();

**if** (Exec.*N* <= 8)

Process.*printMatrix*(Exec.*MA*);

} **catch**(Exception e){

e.printStackTrace();

}

}

}

**public** **class** InputSync {

**private** **static** **int** *F* = 0;

**public** **synchronized** **void** inputSignal() {

*F*++;

**if** (*F* >= 1) notifyAll();

}

**public** **synchronized** **void** waitForInput() {

**try** {

**if** (*F* < 1) wait();

} **catch**(Exception e){

e.printStackTrace();

}

}

}

**public** **class** Resources {

**private** **static** **int** *N* = Exec.*N*;

**private** **int** a;

**private** **int**[][] MB = **new** **int**[*N*][*N*];

**private** **int**[][] MO = **new** **int**[*N*][*N*];

**public** **int** getA() {

**return** a;

}

**public** **int**[][] getMB() {

**return** MB;

}

**public** **int**[][] getMO() {

**return** MO;

}

**public** **synchronized** **void** setA(**int** a) {

**this**.a = a;

}

**public** **synchronized** **void** setMB(**int** x) {

Process.*inputMatrix*(MB, x);

}

**public** **synchronized** **void** setMO(**int** x) {

Process.*inputMatrix*(MO, x);

}

}

**public** **class** Exec {

**public** **static** **int** *N* = 1000;

**public** **static** **int** *P* = 4;

**public** **static** **int** *H* = *N*/*P*;

**public** **static** **int**[][] *MA* = **new** **int** [*N*][*N*];

**public** **static** **int**[][] *MZ* = **new** **int** [*N*][*N*];

**public** **static** **int**[][] *MF* = **new** **int** [*N*][*N*];

/\*\*

\* **@param** args

\*/

**public** **static** **void** main(String[] args) {

InputSync inputSync = **new** InputSync();

CalcSync calcSync = **new** CalcSync();

Resources resources = **new** Resources();

T1thread T1 = **new** T1thread(inputSync,calcSync,resources);

T2thread T2 = **new** T2thread(inputSync,calcSync,resources);

T3thread T3 = **new** T3thread(inputSync,calcSync,resources);

T4thread T4 = **new** T4thread(inputSync,calcSync,resources);

T1.start();

T2.start();

T3.start();

T4.start();

}

}

**class** T1thread **extends** Thread {

**int** id = 0;

**int**[][] MB;

**int**[][] MO;

**int** a;

InputSync inputSync;

CalcSync calcSync;

Resources resources;

**public** T1thread(InputSync inputSync, CalcSync calcSync, Resources resources) {

**this**.inputSync = inputSync;

**this**.calcSync = calcSync;

**this**.resources = resources;

}

**public** **void** run() {

**long** start,stop;

Calendar cal = Calendar.*getInstance*();

start = cal.getTimeInMillis();

System.*out*.println("T1 started");

System.*out*.println("T1 inputs data");

resources.setA(1);

resources.setMB(1);

resources.setMO(1);

Process.*inputMatrix*(Exec.*MZ*, 1);

Process.*inputMatrix*(Exec.*MF*, 1);

inputSync.inputSignal();

System.*out*.println("T1 input finished");

a = resources.getA();

MB = resources.getMB().clone();

MO = resources.getMO().clone();

System.*out*.println("T1 copy finished");

Process.*calculate*(id, a, MO, MB);

System.*out*.println("T1 calc ended");

calcSync.calcSignal();

cal = Calendar.*getInstance*();

stop = cal.getTimeInMillis();

System.*out*.println("T1 finished in " + (stop-start));

}

}

**class** T2thread **extends** Thread {

**int** id = 1;

**int**[][] MB;

**int**[][] MO;

**int** a;

InputSync inputSync;

CalcSync calcSync;

Resources resources;

**public** T2thread(InputSync inputSync, CalcSync calcSync, Resources resources) {

**this**.inputSync = inputSync;

**this**.calcSync = calcSync;

**this**.resources = resources;

}

**public** **void** run() {

**long** start,stop;

Calendar cal = Calendar.*getInstance*();

start = cal.getTimeInMillis();

System.*out*.println("T2 started");

inputSync.waitForInput();

a = resources.getA();

MB = resources.getMB().clone();

MO = resources.getMO().clone();

System.*out*.println("T2 copy finished");

Process.*calculate*(id, a, MO, MB);

System.*out*.println("T2 calc ended");

calcSync.calcSignal();

cal = Calendar.*getInstance*();

stop = cal.getTimeInMillis();

System.*out*.println("T2 finished in " + (stop-start));

}

}

**class** T3thread **extends** Thread {

**int** id = 2;

**int**[][] MB;

**int**[][] MO;

**int** a;

InputSync inputSync;

CalcSync calcSync;

Resources resources;

**public** T3thread(InputSync inputSync, CalcSync calcSync, Resources resources) {

**this**.inputSync = inputSync;

**this**.calcSync = calcSync;

**this**.resources = resources;

}

**public** **void** run() {

**long** start,stop;

Calendar cal = Calendar.*getInstance*();

start = cal.getTimeInMillis();

System.*out*.println("T3 started");

inputSync.waitForInput();

a = resources.getA();

MB = resources.getMB().clone();

MO = resources.getMO().clone();

System.*out*.println("T3 copy finished");

Process.*calculate*(id, a, MO, MB);

System.*out*.println("T3 calc ended");

calcSync.waitForCalc();

cal = Calendar.*getInstance*();

stop = cal.getTimeInMillis();

System.*out*.println("T3 finished in " + (stop-start));

}

}

**class** T4thread **extends** Thread {

**int** id = 3;

**int**[][] MB;

**int**[][] MO;

**int** a;

InputSync inputSync;

CalcSync calcSync;

Resources resources;

**public** T4thread(InputSync inputSync, CalcSync calcSync, Resources resources) {

**this**.inputSync = inputSync;

**this**.calcSync = calcSync;

**this**.resources = resources;

}

**public** **void** run() {

**long** start,stop;

Calendar cal = Calendar.*getInstance*();

start = cal.getTimeInMillis();

System.*out*.println("T4 started");

inputSync.waitForInput();

a = resources.getA();

MB = resources.getMB().clone();

MO = resources.getMO().clone();

System.*out*.println("T4 copy finished");

Process.*calculate*(id, a, MO, MB);

System.*out*.println("T4 calc ended");

calcSync.calcSignal();

cal = Calendar.*getInstance*();

stop = cal.getTimeInMillis();

System.*out*.println("T4 finished in " + (stop-start));

}

}

**import** java.util.Arrays;

**public** **class** Process {

**private** **static** **int** *N* = Exec.*N*;

**private** **static** **int** *H* = Exec.*H*;

**public** **static** **void** inputVector(**int**[] vec, **int** x){

**for** (**int** i = 0; i < vec.length; i++)

vec[i] = x;

}

**public** **static** **void** printVector(**int**[] vec){

System.*out*.println(Arrays.*toString*(vec));

}

**public** **static** **void** inputMatrix(**int**[][] matrix, **int** x){

**for** (**int** i = 0; i < matrix.length; i++)

**for** (**int** j = 0; j < matrix.length; j++)

matrix[i][j] = x;

}

**public** **static** **void** printMatrix(**int**[][] matrix){

**for** (**int** i = 0; i < matrix.length; i++) {

**for** (**int** j = 0; j < matrix.length; j++) {

System.*out*.print(matrix[i][j] + " ");

}

System.*out*.println();

}

}

**public** **static** **void** calculate(**int** id, **int** a, **int**[][] MO, **int**[][]MB ) {

**int**[][] MX = **new** **int**[*N*][*N*];

**int**[][] MA = Exec.*MA*;

**int**[][] MF = Exec.*MF*;

**int**[][] MZ = Exec.*MZ*;

**for** (**int** j=*H* \* id; j<*H* \* (id + 1); j++) {

**for** (**int** i=0; i<*N*; i++) {

MX[i][j] = 0;

**for** (**int** k = 0; k<*N*; k++){

MX[i][j] += MO[i][k]\*MF[k][j];

}

MX[i][j] += MZ[i][j];

}

}

**for** (**int** j=*H* \* id; j<*H* \* (id + 1); j++) {

**for** (**int** i=0; i<*N*; i++) {

MA[i][j] = 0;

**for** (**int** k = 0; k<*N*; k++){

MA[i][j] += MB[i][k]\*MX[k][j];

}

MA[i][j] \*= a;

}

}

}

}