Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

Паралельне програмування в OpenMP

Виконав:

студент групи ІП-22

Сочка О. О.

Перевірив:

Корочкін О. В.

Київ - 2015

**Технічне завдання**



**Етап 1. Побудова паралельного алгоритму**

\* - сортування злиттям

**Спільні ресурси:** MT, a, X

**Етап 2. Розробка алгоритмів процесів (задач)**

|  |  |  |
| --- | --- | --- |
| **№** | **Т1** | **ТС/КУ** |
| 1 | Ввід |  |
| 2 | Сигнал іншим задачам про завершення вводу |  |
| 3 | Очікування вводу |  |
| 4 | Очікування вводу |  |
| 5 | Копія MT, a, X | КУ |
| 6 | Підрахунок |  |
| 7 | Очікування завершення підрахунку в T2 |  |
| 8 | Підрахунок |  |
| 9 | Очікування завершення підрахунку в T3 |  |
| 10 | Підрахунок |  |
| 11 | Очікування завершення підрахунку в T5 |  |
| 12 | Підрахунок |  |
| 13 | Сигнал іншим потокам про підрахунок V |  |
| 14 | Підрахунок |  |
| 15 | Очікування завершення підрахунку у інших потоках |  |
| 16 | Вивід |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т2** | **ТС/КУ** |
| 1 | Очікування вводу |  |
| 2 | Копія MT, a, X | КУ |
| 3 | Підрахунок |  |
| 4 | Сигнал T1 про підрахунок |  |
| 5 | Очікування завершення підрахунку V в T1 |  |
| 6 | Підрахунок |  |
| 7 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т3** | **ТС/КУ** |
| 1 | Ввід |  |
| 2 | Сигнал T1 про завершення вводу |  |
| 3 | Очікування вводу |  |
| 4 | Копія MT, a, X | КУ |
| 5 | Підрахунок |  |
| 6 | Очікувати завершення підрахунку в T4 |  |
| 7 | Підрахунок |  |
| 8 | Сигнал T1 про підрахунок |  |
| 9 | Очікування завершення підрахунку V в T1 |  |
| 10 | Підрахунок |  |
| 11 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т4** | **ТС/КУ** |
| 1 | Очікування вводу |  |
| 2 | Копія MT, a, X | КУ |
| 3 | Підрахунок |  |
| 4 | Сигнал T3 про підрахунок |  |
| 5 | Очікування завершення підрахунку V в T1 |  |
| 6 | Підрахунок |  |
| 7 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т5** | **ТС/КУ** |
| 1 | Ввід |  |
| 2 | Сигнал T1 про завершення вводу |  |
| 3 | Очікування вводу |  |
| 4 | Копія MT, a, X | КУ |
| 5 | Підрахунок |  |
| 6 | Очікувати завершення підрахунку в T6 |  |
| 7 | Підрахунок |  |
| 9 | Очікування завершення підрахунку в T7 |  |
| 10 | Підрахунок |  |
| 11 | Сигнал T1 про підрахунок |  |
| 12 | Очікування завершення підрахунку V в T1 |  |
| 13 | Підрахунок |  |
| 14 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т6** | **ТС/КУ** |
| 1 | Очікування вводу |  |
| 2 | Копія MT, a, X | КУ |
| 3 | Підрахунок |  |
| 4 | Сигнал T5 про підрахунок |  |
| 5 | Очікування завершення підрахунку V в T1 |  |
| 6 | Підрахунок |  |
| 7 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т7** | **ТС/КУ** |
| 1 | Очікування вводу |  |
| 2 | Копія MT, a, X | КУ |
| 3 | Підрахунок |  |
| 4 | Очікувати завершення підрахунку в T8 |  |
| 5 | Підрахунок |  |
| 6 | Сигнал T5 про підрахунок |  |
| 7 | Очікування завершення підрахунку V в T1 |  |
| 8 | Підрахунок |  |
| 9 | Сигнал потоку T1 про завершення підрахунку |  |

|  |  |  |
| --- | --- | --- |
| **№** | **Т8** | **ТС/КУ** |
| 1 | Очікування вводу |  |
| 2 | Копія MT, a, X | КУ |
| 3 | Підрахунок |  |
| 4 | Сигнал T7 про підрахунок |  |
| 5 | Очікування завершення підрахунку V в T1 |  |
| 6 | Підрахунок |  |
| 7 | Сигнал потоку T1 про завершення підрахунку |  |

**Етап 3. Розробка схеми взаємодії задач**



**Етап 4. Розробка програми**

Program.cs

1 /\*\*

2 \* @author Sochka Oleksandr IP-22

3 \* PP-2, Lab #4 (C#)

4 \* P = 8

5 \* A = sort(X) (MT \* MK) - a \* Z

6 \* 02.04.2015

7 \*/

8

9 #include <algorithm>

10 #include <iostream>

11 #include <vector>

12 #include <omp.h>

13 using namespace std;

14

15 const auto P = 8;

16 const auto N = 8;

17 const auto H = N / P;

18

19 static\_assert(N % P == 0, "N should be divisible by P");

20

21 int a = 0;

22 vector<int> A(N);

23 vector<int> V(N);

24 vector<int> X;

25 vector<int> Z;

26 vector<vector<int>> MK;

27 vector<vector<int>> MT;

28

29 int main() {

30 omp\_set\_dynamic(0);

31 omp\_set\_num\_threads(P);

32

33 int Ti;

34 #pragma omp parallel shared(a, MT, X) private(Ti)

35 {

36 Ti = omp\_get\_thread\_num();

37

38 #pragma omp critical

39 cout << "T" << Ti << " created" << endl;

40

41 if (Ti == 0) {

42 X.assign(N, 1);

43 }

44

45 if (Ti == 2) {

46 Z.assign(N, 1);

47 MT.assign(N, vector<int>(N, 1));

48 }

49

50 if (Ti == 4) {

51 a = 1;

52 MK.assign(N, vector<int>(N, 1));

53 }

54

55 #pragma omp barrier

56 ;

57 vector<vector<int>> MT;

58 vector<int> X;

59 int a;

60 #pragma omp critical

61 {

62 a = ::a;

63 MT = ::MT;

64 X = ::X;

65 }

66

67 #pragma omp for

68 for (auto i = 0; i < P; ++i)

69 sort(X.begin() + i\*H, X.begin() + i \* H + H);

70

71 #pragma omp for

72 for (auto i = 0; i < P; i += 2) {

73 vector<int> part(H \* 2);

74 const auto it = X.begin() + i \* H;

75 merge(it, it + H, it + H, it + 2 \* H, part.begin());

76 copy(part.begin(), part.end(), it);

77 }

78

79 #pragma omp for

80 for (auto i = 0; i < P; i += 4) {

81 vector<int> part(H \* 4);

82 const auto it = X.begin() + i \* H;

83 merge(it, it + 2 \* H, it + 2 \* H, it + 4 \* H, part.begin());

84 copy(part.begin(), part.end(), it);

85 }

86

87 #pragma omp master

88 merge(X.begin(), X.begin() + 4 \* H, X.begin() + 4 \* H, X.end(), V.begin());

89

90 #pragma omp barrier

91

92 #pragma omp for

93 for (auto k = 0; k < N; ++k) {

94 for (auto i = 0; i < N; ++i)

95 for (auto j = 0; j < N; ++j)

96 A[k] += V[k] \* MT[i][j] \* MK[j][k];

97 A[k] -= a \* Z[k];

98 }

99

100 #pragma omp master

101 if (N <= 12) {

102 cout << '\n';

103 for (auto val : A)

104 cout << val << ' ';

105 cout << '\n' << endl;

106 }

107

108 #pragma omp barrier

109 #pragma omp critical

110 cout << "T" << Ti << " finished it's job" << endl;

111 }

112

113 cout << "Press enter to exit..." << endl;

114 cin.get();

115 }