Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №1

З предмету: «Паралельні та розподілені обчислення»

Виконав: студент групи ІО-91

Щербина О.В.

Київ 2012р.

Задание на лабораторную работу:

Математическая задача:

A=sort(B+C(MO\*MX)).

Ввод данных: поток1-A,B,C; поток 2-MO,MX.

Структура ПКС:

ОП

2

1

A, B, C MO, MX

Средства программирования:

Язык АДА, семафоры

**Этап 1.Разработка параллельного алгоритма**

1. Tj=sort(BH+C\*(MOH\*MX))
2. A=sort(Tj,Tj)

OP: C,MX

**Этап 2. Разработка алгоритмов роботы каждого процесса**

Задача Т1

1. Ввод B,C
2. Сигнал задаче Т2 про завершение ввода данных-----S2-1
3. Ждать сигнала от Т2 про окончание ввода -------------W2-1
4. Копировать MX1=MX и C1= С– критический блок - Skd
5. Счет Tj=sort(BH+C\*(MOH\*MX))
6. Ждать окончание расчета в Т2 --------W2-2
7. Счет A=sort(Tj,Tj)
8. Вывести результат счета

Задача Т2

1. Ввод MO,MX
2. Сигнал задаче Т1 про завершение ввода ----------S1-1
3. Ждать сигнал от Т1 про окончание ввода---------W1-1
4. Копировать матрицу MX и вектор С– критический блок – Skd
5. Счет Tj=sort(BH+C\*(MOH\*MX))
6. Сигнал Т1 про окнчание вычислений ---------------S1-2

**Этап3. Разработка структурной схемы взаимодействия задач**

На структурной схеме взаимодействия задач введены такие семафоры:

Skd- для управления доступом к общим ресурсам C,MX

Sem1 – для синхронизации про завершение ввода в Т1

Sem2 – для синхронизации про завершение ввода в Т2

Sem3 – для синхронизации про завершение вычислений в Т2

T2

T1

Листинг программы

Source file: ..\main.ada Thu Feb 02 12:01:59 2012

1 --Prepeared by Alexandr Shcherbina

2 --Group IO-91

3 --Date 1.02.2012

4 with Ada.Synchronous\_Task\_Control;

5 with Ada.Text\_IO;

6 use Ada.Synchronous\_Task\_Control;

7 procedure main is

8

9 N:integer:=7000;

10 P:integer:=2;

11 H:integer:=N/P;

12

13 type Vector is array (1..N) of integer;

14 type HVector is array(1..H) of integer;

15 type Matrix is array (1..N) of Vector;

16 type HMatrix is array (1..H) of Vector;

17

18 MO,MX:Matrix;

19 A,B,C:Vector;

20 Sem1,Sem2,Sem3,Skd:Suspension\_Object;

21 A22:HVector;

22 --Function wich add vectors

23 function addVector(V1:in HVector;V2:in HVector) return HVector is

24 Vr: HVector;

25 begin

26 for i in 1..H loop

27 Vr(i):=V1(i)+V2(i);

28 end loop;

29 return Vr;

30 end addVector;

31

32 --Function wich multiply vector and matrix

33 function multiplyVectorMatrix(V:in Vector;M:in HMatrix) return HVector is

34 Vr: HVector;

35 begin

36 for i in 1..H loop

37 Vr(i):=0;

38 for j in 1..N loop

39 Vr(i):=Vr(i)+V(j)\*M(i)(j);

40 end loop;

41 end loop;

42 return Vr;

43 end multiplyVectorMatrix;

44

45 --Function wich multiply two matrix

46 function multiplyMatrixMatrix(M1:in HMatrix;M2:in Matrix) return HMatrix is

47 Mr: HMatrix;

48 begin

49 for i in 1..H loop

50 for j in 1..N loop

51 Mr(i)(j):=0;

52 for k in 1..N loop

53 Mr(i)(j):=Mr(i)(j)+M1(i)(j)\*M2(j)(k);

54 end loop;

55 end loop;

56 end loop;

57 return Mr;

58 end multiplyMatrixMatrix;

59

60 --Function wich sort vector

61 function firstSort(V:in Hvector) return HVector is

62 Vr:HVector;

63 max:integer;

64 begin

65 for i in 1..H loop

66 Vr(i):=V(i);

67 end loop;

68 for i in 1..H loop

69 max:=V(i);

70 for j in i+1..N loop

71 if max<V(j) then

72 max:=V(j);

73 end if;

74 end loop;

75 end loop;

76 return Vr;

77 end firstSort;

78

79 --Procedure wich contain two paralel task

80 procedure startTasks is

81 task T1;

82 package Int\_IO is new Ada.Text\_IO.Integer\_IO(Integer);

83 C1:Vector;

84 MX1:Matrix;

85 p1,p2:Integer;

86 MO1,MM1:HMatrix;

87 D1,A1,E1,B1:HVector;

88 task body T1 is

89 begin

90 for i in 1..N loop

91 B(i):=1;

92 C(i):=1;

93 end loop;

94

95

96 Set\_True(Sem1);

97 Suspend\_Until\_True(Sem2);

98

99 Suspend\_Until\_True(Skd);

100

101 for i in 1..N loop

102 C1(i):=C(i);

103 end loop;

104 for i in 1..N loop

105 for j in 1..N loop

106 MX1(i)(j):=MX(i)(j);

107 end loop;

108 end loop;

109

110 Set\_True(Skd);

111

112

113 for i in 1..H loop

114 for j in 1..N loop

115 MO1(i)(j):=MO(i)(j);

116 end loop;

117 end loop;

118 for i in 1..H loop

119 B1(i):=B(i);

120 end loop;

121

122 MM1:=multiplyMatrixMatrix(MO1,MX1);

123 D1:=multiplyVectorMatrix(C1,MM1);

124 E1:=addVector(B1,D1);

125 A1:=firstSort(E1);

126 Suspend\_Until\_True(Sem3);

127 p1:=1;

128 p2:=1;

129 for i in 1..N loop

130 if p1>H then

131 A(i):=A22(p2);

132 p2:=p2+1;

133 else

134 if p2>H then

135 A(i):=A1(p1);

136 p1:=p1+1;

137 else

138 if A1(p1)>A22(p2) then

139 A(i):=A1(p1);

140 p1:=p1+1;

141 else

142 A(i):=A22(p2);

143 p2:=p2+1;

144 end if;

145 end if;

146 end if;

147 end loop;

148 Ada.Text\_IO.Put\_Line("Vector A elements");

149 for i in 1..N loop

150 Int\_IO.Put(A(i));

151 Ada.Text\_IO.Put(" ");

152 end loop;

153 Ada.Text\_IO.Put\_Line("End vector A");

154 end T1;

155

156 task T2;

157 task body T2 is

158 C2:Vector;

159 B2,D2, E2, A2:HVector;

160 MX2:Matrix;

161 MM2,MO2:HMatrix;

162 begin

163 for i in 1..N loop

164 for j in 1..N loop

165 MO(i)(j):=1;

166 MX(i)(j):=1;

167 end loop;

168 end loop;

169

170

171 Set\_True(Sem2);

172 Suspend\_Until\_True(Sem1);

173

174 Suspend\_Until\_True(Skd);

175

176 for i in 1..N loop

177 for j in 1..N loop

178 MX2(i)(j):=MX(i)(j);

179 end loop;

180 end loop;

181

182 for i in 1..N loop

183 C2(i):=C(i);

184 end loop;

185 Set\_True(Skd);

186

187

188 for i in 1..H loop

189 for j in 1..N loop

190 MO2(i)(j):=MO(i+H)(j);

191 end loop;

192 end loop;

193 for i in 1..H loop

194 B2(i):=B(i+H);

195 end loop;

196 MM2:=multiplyMatrixMatrix(MO2,MX2);

197 D2:=multiplyVectorMatrix(C2,MM2);

198 E2:=addVector(B2,D2);

199 A2:=firstSort(E2);

200 for i in 1..H loop

201 A22(i):=A2(i);

202 end loop;

203 Set\_True(Sem3);

204 end T2;

205

206 begin

207 null;

208 end startTasks;

209

210 begin

211 Ada.Text\_IO.put\_line("Start main procedure");

212 Set\_True(Skd);

213 Set\_False(Sem3);

214 Set\_False(Sem1);

215 Set\_False(Sem2);

216 startTasks;

217 end main;

218