Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчёт по теоретическому заданию в рамках курса «Суперкомпьютерное моделирование и технологии»

Выполнила:
студентка 611 группы
В.М. Кобзева
Вариант 34

Москва, 2020

Содержание

Исходный фрагмент и описание информационной структуры

В качестве условия задачи выступает фрагмент программы на языке C. Исходный код, который описывается в данном отчете (34 вариант) приведен ниже:

Листинг 1 **Исходный фрагмент на C**

```
for(i = 2; i <= n+1; ++i)
    C[i] = C[i-2] * e;
for(i = 2; i <= n+1; ++i)
    for(j = 2; j <= m+1; ++j)
        B[i][j] = B[i-1][j-2];
for(i = 2; i <= n+1; ++i){
    A[i][1][1]=B[i][m] + C[i];
    for(j = 2; j <= m+1; ++j)
        for(k = 1; k <= n-1; ++k)
            A[i][j][k] = A[i][j][k] + A[i][j-1][1];
}
```

Требовалось выполнить исследование информационной структуры этого фрагмента, то есть выявить имеющиеся в ней зависимости по данным и их характер, после чего составить описание информационной структуры на языке разметки Algolang. Итоговый листинг описания структуры фрагмента на языке Algolang получился вот таким:

Листинг 2 **Описание структуры фрагмента на языке Algoland**

```
<algo>
  <params>
    <param name = "N" type = "int" value = "5"/>
    <param name = "M" type = "int" value = "4"/>
  </params>
  <block id = "0" dims = "1">
    <arg name = "i" val = "2..N+1"/>
    <vertex condition = "" type = "1">
      <in src = "i - 2"/>
    </vertex>
  </block>
```

```
12   <block id = "1" dims = "2">
13     <arg name = "i" val = "2..N+1"/>
14     <arg name = "j" val = "2..M+1"/>
15     <vertex condition = "" type = "1">
16       <in src = "i - 1, j - 2"/>
17     </vertex>
18   </block>
19   <block id = "2" dims = "3">
20     <arg name = "i" val = "2..N+1"/>
21     <arg name = "j" val = "1..M+1"/>
22     <arg name = "k" val = "1..N-1"/>
23     <vertex condition = "(j == 1) and (k == 1)" type = "1">
24       <in bsrc = "1" src = "i,M"/>
25       <in bsrc = "0" src = "i"/>
26     </vertex>
27     <vertex condition = "(j > 1)" type = "1">
28       <in src = "i, j - 1, 1"/>
29     </vertex>
30   </block>
31 </algo>
```

Информационный граф фрагмента и его свойства

Для получения визуализации информационного графа код из листинга 2 был загружен в систему Algoload и был получен следующий информационный граф:

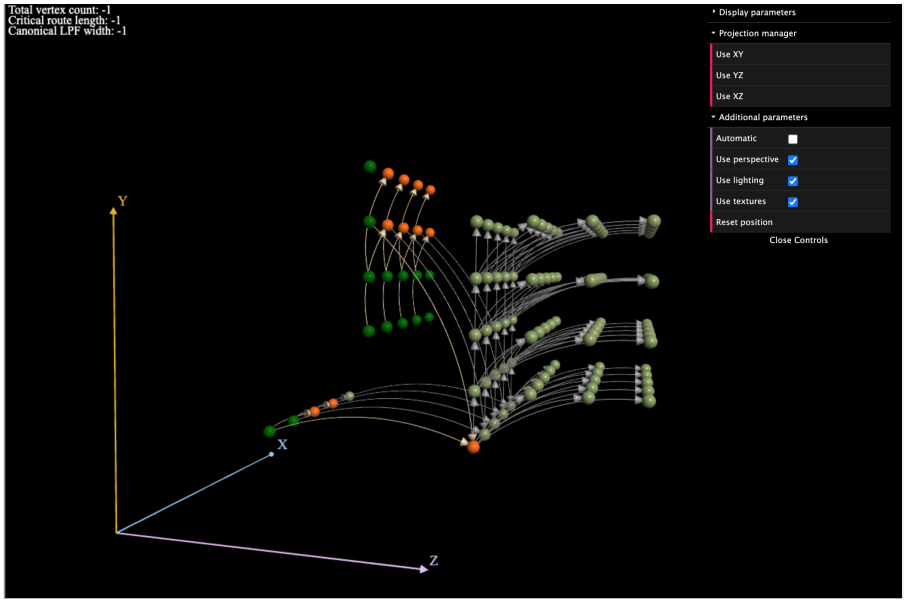

Рисунок 1 Визуализация информационного графа

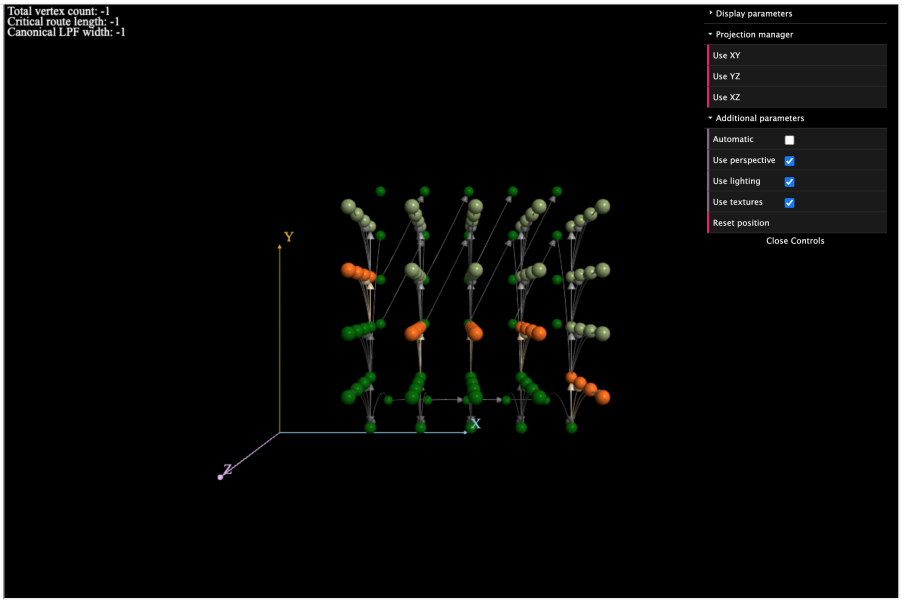

Рисунок 2 Проекция oXY

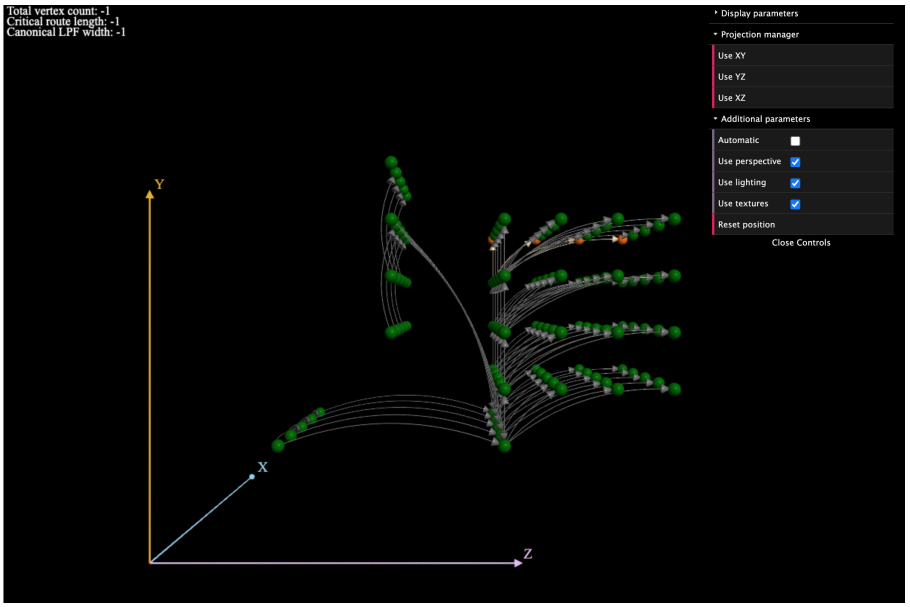

Рисунок 3 Проекция oYZ

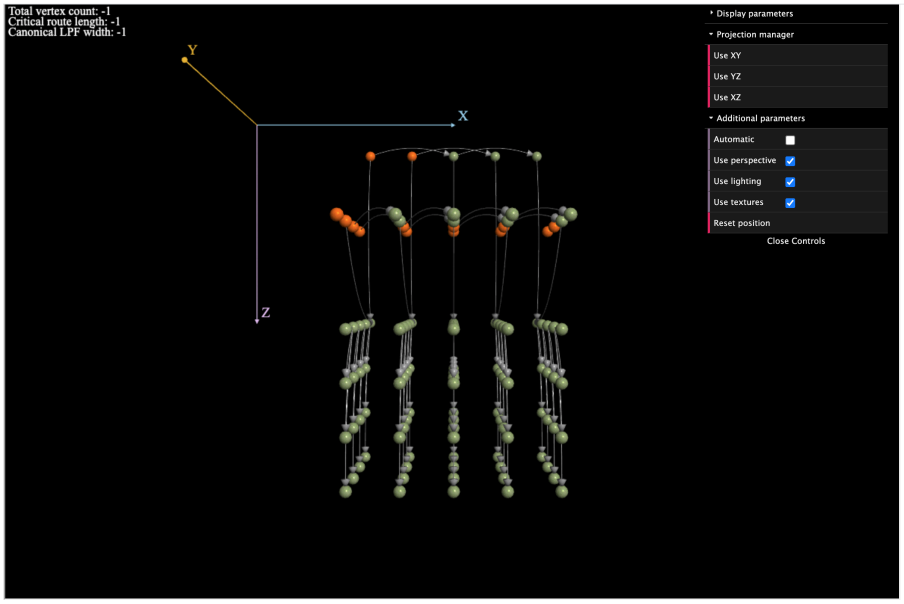

Рисунок 4 Проекция oXZ

Базовые свойства информационного графа оказались следующими:

1. Число вершин в информационном графе (последовательная сложность) в общем виде: $N + N * M + N + M \cdot N \cdot (N-1)$. Для нарисованного графа количество вершин равно 110.

2. Длина критического пути в информационном графе (параллельная сложность) в общем виде: $m + 1 + max\left(\lceil \frac{N}{2} - 1 \rceil; min\left(N-1; \lfloor \frac{M}{2} - 1 \rfloor\right)\right)$, где $\lceil\rceil$ - округление до целого в большую сторону, $\lfloor\rfloor$ - округление до целого в меньшую сторону. Вычисления элементов A[N+2][M+1][k], k = 1..n-1 - длина критического пути. Для нарисованного графа длина критического пути равна 7 $(4 + 1 \quad max(2; min(4, 1)) = 7)$.

3. Ширина ярусно-параллельной формы зависит от значений параметров N и M. Было выведено правило, с помощью которого можно определить ширину ЯПФ и на каком ярусе достигается данное значение впервые.

   (a) Если $\lceil \frac{N}{2} \rceil - 1 < M$, то число верши W:

   $$W = max\left[N(N-1); max(M,2) + min(M,2) \cdot N\right]$$

      i. Если $W == N(N-1)$, то значение яруса $level$:

      $$level = max\left[\left\lceil \frac{N}{2} \right\rceil; min\left(N, \left\lfloor \frac{M}{2} \right\rfloor\right)\right] + 2;$$

      ii. Если $W == max(M,2) + min(M,2) \cdot N$, то:

      $$level = 1;$$

   (b) Если $\lceil \frac{N}{2} \rceil - 1 \geq M$, то число верши W:

   $$W = max\left[2M(N-1) + min((N-2M), 4); max(M,2) + min(M,2) \cdot N\right]$$

      i. Если $W == 2M(N-1) + min((N-2M), 4)$, то значение яруса $level$:

      $$level = M + 2;$$

ii. Если $W == max(M, 2) + min(M, 2) \cdot N$, то: $level = 1$.

Для приведенного примера данное значение равно: Код на языке Python, которые может вывести значения для различных M и N:

Листинг 3 **Значения яруса и ширины на Python и разных M, N**

```
import math
for n in range(13):
        for m in range(13):
            if(n >= 2 and m >= 1):
                if(round(n/2.0) - 1 < m):
                    W = max(n * (n - 1), max(m, 2) + min(m, 2) * n
                        )
                    if W == max(m, 2) + min(m, 2) * n:
                        print('n = ', n, 'm = ', m, 'level =', 1,
                              'Vertex count =', W)
                    elif W == n * (n - 1):
                        print('n = ', n, 'm = ', m,
                          'level =', max(round(n/2.0),
                                         min(n, math.floor(m/2.0))
                                             ) + 2,
                          'Vertex count =', W)
                if(round(n/2.0) - 1 >= m):
                    W = max(2 * m * (n - 1) + min((n - 2 * m), 4),
                            max(m, 2) + min(m, 2) * n)
                    if W == max(m, 2) + min(m, 2) * n:
                        print('n = ', n, 'm = ', m, 'level = ', 1,
                              'Vertex count= ', W)
                    elif W == 2 * m * (n - 1) + min((n - 2 * m),
                        4):
                        print('n = ', n, 'm = ', m, 'level = ', m
                            + 2,
                              'Vertex count = ', W)
```

Для нарисованного графа ширина ЯПФ равна 20 и достигается на 5 уровне(впервые).

4. Максимальная глубина вложенности циклов 3 (достигается в третьем гнезде циклов).

5. Число различных типов дуг: $(N-1)+1+[\text{if}\quad(M\geq 2)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]+$ $+[\text{if}\quad(M\geq 3)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]+[\text{if}\quad(N\geq 3)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]$. Первое слагаемое отвечает за дуги в массиве A, второе слагаемое отвечает за дуги из массива C в A, третье слагаемое отвечает за дуги из массива B в A (появляются при $M\geq 2$), четвертое слагаемое за дуги внутри массива B (появляются при $M\geq 3$), последнее слагаемое отвечает за дуги внутри массива C (появляются при $N\geq 3$). Для нарисованного графа количество различных типов дуг равно 8.

6. Длинные дуги присутствуют (то есть дуги, длина которых зависит от внешнего параметра), их число равно N, они появляются при $M\geq 2$. Это дуги идущие из массива B в A, так как они идут из B[i][M] и становятся длиннее с увеличением M. Для нарисованного графа количество длинных дуг 5.

7. Количество областей регулярности в информационном графе: $2+[\text{if}\quad(M\geq 2)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]+[\text{if}\quad(M\geq 3)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]+$ $+[\text{if}\quad(N\geq 3)\quad\text{then}\quad 1\quad\text{else}\quad 0\quad\text{end}]$. То есть максимальное количество областей регулярности 5. Первые две области регулярности - это в массиве C(для вершин из массива C в A) и A, они есть всегда. Далее при $M\geq 2$ появляются дуги из массива B в A и это еще одна область регулярности. При $M\geq 3$ появляются дуги внутри массива B и это еще одна область регулярности. При $N\geq 3$ в массиве C две области регулярности, одна область - вершины, из которых идут дуги в массив A и внутри массива C, другая область регулярности - вершины из которых идут дуги только к массиву A. Для нарисованного графа количество областей регулярности равно 5.

Параллельная реализация алгоритма

Параллельная (многонитевая) реализация исходного алгоритма с использованием технологии OpenMP:

Листинг 4 **Исходный фрагмент на C**

```
for(i = 2; i <= n+1; ++i)
  C[i] = C[i-2] * e;

for(i = 2; i <= n+1; ++i)
  for(j = 2; j <= m+1; ++j)
    B[i][j] = B[i-1][j-2];

#pragma omp parallel for
for(i = 2; i <= n+1; ++i){
  A[i][1][1]=B[i][m] + C[i];
  for(j = 2; j <= m+1; ++j)
    #pragma omp parallel for
    for(k = 1; k <= n-1; ++k)
      A[i][j][k] = A[i][j][k] + A[i][j-1][1];
}
```