



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

Москва, 2020

# Содержание

Исходный фрагмент и описание информационной структуры	3
Информационный граф фрагмента и его свойства	5
Параллельная реализация алгоритма	10

## Исходный фрагмент и описание информационной структуры

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

Листинг 1 Исходный фрагмент на С

```
1 for (i = 2; i <= n+1; ++i)
2     C[i] = C[i-2] * e;
3 for (i = 2; i <= n+1; ++i)
4     for (j = 2; j <= m+1; ++j)
5         B[i][j] = B[i-1][j-2];
6 for (i = 2; i <= n+1; ++i){
7     A[i][1][1]=B[i][m] + C[i];
8     for (j = 2; j <= m+1; ++j)
9         for (k = 1; k <= n-1; ++k)
10            A[i][j][k] = A[i][j][k] + A[i][j-1][1];
11 }
```

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

```
1 <algo>
2   <params>
3     <param name = "N" type = "int" value = "5"/>
4     <param name = "M" type = "int" value = "4"/>
5   </params>
6   <block id = "0" dims = "1">
7     <arg name = "i" val = "2..N+1"/>
8     <vertex condition = "" type = "1">
9       <in src = "i - 2"/>
10    </vertex>
11  </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:

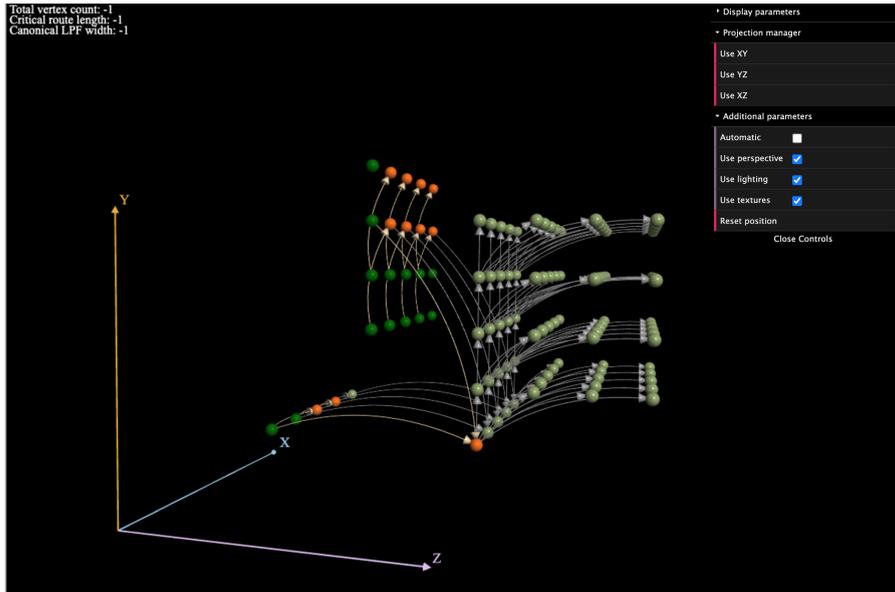


Рисунок 1 Визуализация информационного графа

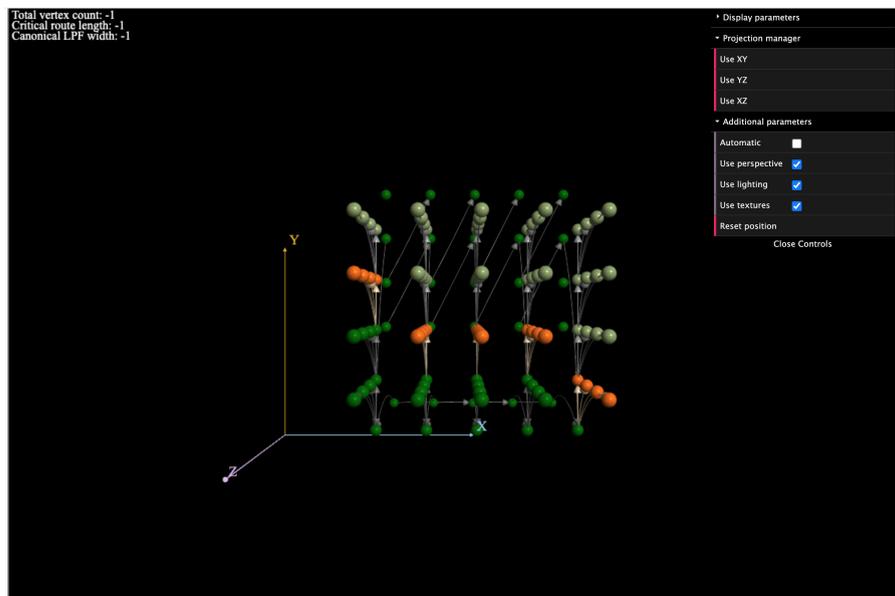


Рисунок 2 Проекция оХУ

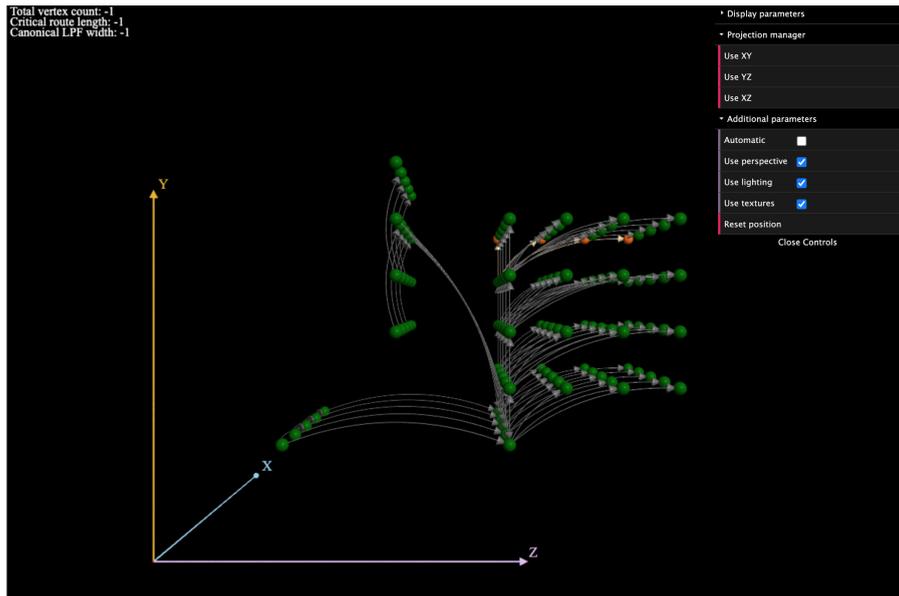


Рисунок 3 Проекция оYZ

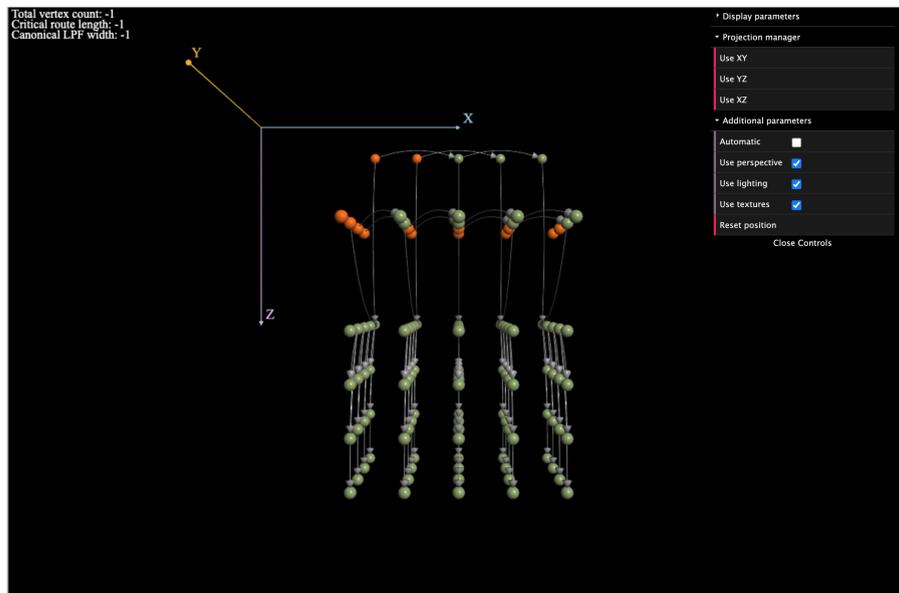


Рисунок 4 Проекция оXZ

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:  $m + 1 + \max\left(\lceil \frac{N}{2} - 1 \rceil; \min(N - 1; \lfloor \frac{M}{2} - 1 \rfloor)\right)$ , где  $\lceil \cdot \rceil$  - округление до целого в большую сторону,  $\lfloor \cdot \rfloor$  - 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 ( $4 + 1 + \max(2; \min(4, 1)) = 7$ ).
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  $W$ :

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

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  $W$ :

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

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

```
1 import math
2 for n in range(13):
3     for m in range(13):
4         if(n >= 2 and m >= 1):
5             if(round(n/2.0) - 1 < m):
6                 W = max(n * (n - 1), max(m, 2) + min(m, 2) * n
7                     )
8                 if W == max(m, 2) + min(m, 2) * n:
9                     print('n = ', n, 'm = ', m, 'level = ', 1,
10                        'Vertex count = ', W)
11                elif W == n * (n - 1):
12                    print('n = ', n, 'm = ', m,
13                        'level = ', max(round(n/2.0),
14                        min(n, math.floor(m/2.0))
15                        ) + 2,
16                        'Vertex count = ', W)
17                if(round(n/2.0) - 1 >= m):
18                    W = max(2 * m * (n - 1) + min((n - 2 * m), 4),
19                        max(m, 2) + min(m, 2) * n)
20                    if W == max(m, 2) + min(m, 2) * n:
21                        print('n = ', n, 'm = ', m, 'level = ', 1,
22                           'Vertex count = ', W)
23                    elif W == 2 * m * (n - 1) + min((n - 2 * m),
24                        4):
25                        print('n = ', n, 'm = ', m, 'level = ', m
26                            + 2,
27                            '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:  $(N - 1) + 1 + [\text{if } (M \geq 2) \text{ then } 1 \text{ else } 0 \text{ end}] +$   
 $+ [\text{if } (M \geq 3) \text{ then } 1 \text{ else } 0 \text{ end}] + [\text{if } (N \geq 3) \text{ then } 1 \text{ else } 0 \text{ end}]$ .  
 Первое слагаемое отвечает за дуги в массиве А, второе слагаемое отвечает за дуги из массива С в А, третье слагаемое отвечает за дуги из массива В в А (появляются при  $M \geq 2$ ), четвертое слагаемое за дуги внутри массива В (появляются при  $M \geq 3$ ), последнее слагаемое отвечает за дуги внутри массива С (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 В в А, так как они идут из  $V[i][M]$  и становятся длиннее с увеличением  $M$ . Для нарисованного графа количество длинных дуг 5.
7. Количество областей регулярности в информационном графе:  
 $2 + [\text{if } (M \geq 2) \text{ then } 1 \text{ else } 0 \text{ end}] + [\text{if } (M \geq 3) \text{ then } 1 \text{ else } 0 \text{ end}] +$   
 $+ [\text{if } (N \geq 3) \text{ then } 1 \text{ else } 0 \text{ end}]$ . То есть максимальное количество областей регулярности 5. Первые две области регулярности - это в массиве С (для вершин из массива С в А) и А, они есть всегда. Далее при  $M \geq 2$  появляются дуги из массива В в А и это еще одна область регулярности. При  $M \geq 3$  появляются дуги внутри массива В и это еще одна область регулярности. При  $N \geq 3$  в массиве С две области регулярности, одна область - вершины, из которых идут дуги в массив А и внутри массива С, другая область регулярности - вершины из которых идут дуги только к массиву А. 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

```
1 for(i = 2; i <= n+1; ++i)
2   C[i] = C[i-2] * e;
3
4 for(i = 2; i <= n+1; ++i)
5   for(j = 2; j <= m+1; ++j)
6     B[i][j] = B[i-1][j-2];
7
8 #pragma omp parallel for
9 for(i = 2; i <= n+1; ++i){
10  A[i][1][1]=B[i][m] + C[i];
11  for(j = 2; j <= m+1; ++j)
12    #pragma omp parallel for
13    for(k = 1; k <= n-1; ++k)
14      A[i][j][k] = A[i][j][k] + A[i][j-1][1];
15 }
```