N* Новосибирский государственный университет

*НАСТОЯЩАЯ НАУКА

Разработка алгоритмов генерации и создание генератора LuNA-программ

Автор: Синюков В.К.
Научный руководитель: Перепелкин В.А.

Всероссийская летняя XLI молодежная Школа-конференция по параллельному программированию
Новосибирск, 2023

*Проблема и Цель

Проблема: автоматическая генерация параллельных программ из последовательных - это в общем случае задача алгоритмически труднорешаемая.
Задача автоматической генерации программ изначально в LuNA не ставилась, тем не менее существует достаточно много готовых наработок и решений по автоматической генерации параллельных программ в конкретных предметных областях (например, Sapfor), которые можно использовать для генерации LuNA-программ.
Объемлющая цель: создание автоматического генератора LuNA-программ из последовательных программ численного моделирования, написанных на языках C/C++/Fortran.
Частная цель (текущая): создание такого автоматического генератора, который сможет сгенерировать данную программу, реализующую численный метод Якоби решения системы дифференциальных уравнений.

*Схема генерации и задачи на школу

*Примеры

```

int main()
{
    int a, b, c;
    a = 5;
    b = 6;
    c = a * b;
    return 0;
}

```

JSON

Генератор

```
import init_c(int, int, name);
import init_a(name);
import init_b(name);

sub main()
{
    df a, b, c;

    init_a(a);
    init_b(b);
    init_c(a, b, c);

}

```

```
extern "C"
{
    void init_c(int a, int b, OutputDF& c)
    {
        c = a * b;
    }
    void init_a(OutputDF& a)
    {
        a = 5;
    }
    void init_b(OutputDF& b)
    {
        b = 6;
    }
}
```

```
#define N 1000000

int main()
{
    int A[N];
    for (int i = 0; i < N; ++i)
        A[i] = i;
    return 0;
}
```

```
import calc_A(int, int, name);

sub main(int nf)
{

    df A;

    for f0=0..nf-1 {
        calc_A(f0, nf, A[f0]);
    }

}
```

```
#define N 1000000

#include "ucenv.h"

int get_size(int f, int nf, int init_size)
{
    return (f * init_size / nf - (f - 1) * init_size / nf);
}

int get_start(int f, int nf, int init_size)
{
    return ((f - 1) * init_size / nf);
}


extern "C"
{
    void calc_A(int f0, int nf0, OutputDF& A)
    {
        int *A_arr = A.create<int>(N);
        int start0 = get_start(f0, nf0, N);
        int size0 = get_size(f0, nf0, N);
        for (int i = 0; i < size0; ++i)
        {
            A_arr[i] = (i + start0);
        }

    }
}
```

```cpp
#define N 1000
#define M 1000

int main()
{
    int A[N * M];
    for (int j = 0; j < M; ++j)
        for (int i = 0; i < N; ++i)
            A[j * N + i] = j * N + i;
    return 0;
}
```

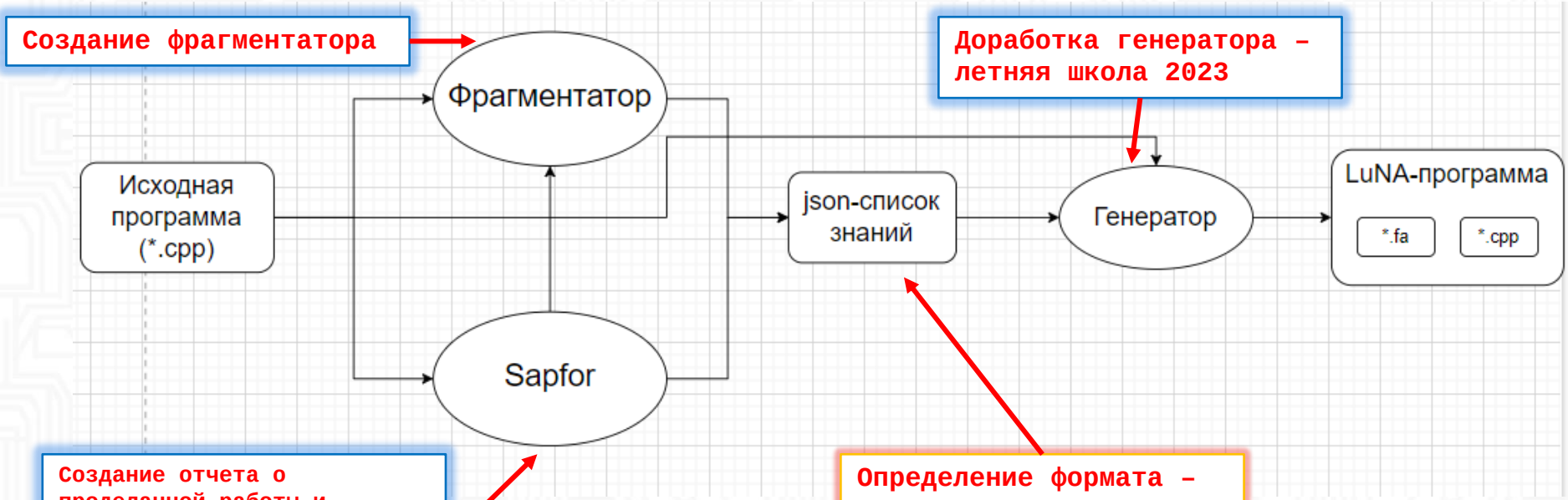
```
import calc_A(int, int, int, int, name);

sub main(int nf)
{

    df A;

    for f0=0..nf-1 {
        for f1=0..nf-1 {
            calc_A(f0, f1, nf, nf, A[f1][f0]);
        }
    }

}
```

```cpp
#define N 1000
#define M 1000

#include "uceny.h"

int get_size(int f, int nf, int init_size)
{
    return (f * init_size / nf - (f - 1) * init_size / nf);
}

int get_start(int f, int nf, int init_size)
{
    return ((f - 1) * init_size / nf);
}


extern "C"
{
    void calc_A(int f0, int f1, int nf0, int nf1, OutputDF& A)
    {
        int *A_arr = A.create<int>(N * M);
        int start0 = get_start(f0, nf0, M);
        int size0 = get_size(f0, nf0, M);
        int start1 = get_start(f1, nf1, N);
        int size1 = get_size(f1, nf1, N);
        for (int j = 0; j < size0; ++j)
        {
            for (int i = 0; i < size1; ++i)
            {
                A_arr[j * N + i] = (j + start0) * N + (i + start1);
            }
        }

    }
}
```

*

*Схема генерации и задачи на школу

*Схема генерации и дальнейшая работа



*Заключение

Итоги проделанной работы:

- ✓ Были разработаны и реализованы
  - алгоритмы генерации примитивных LuNA-программ
  - алгоритмы генерации LuNA-программ с разбиением циклов на фрагменты вычислений
- ✓ Были сделаны продвижения в создании отчета для разработчиков системы Sapfor

*Заключение

Дальнейшие планы:

- Завершение отчета и его представление разработчикам Sapfor
- Разработка фрагментатора
- Доработка генератора

Novosibirsk State University
*THE REAL SCIENCE

RUSSIA, 630090, NOVOSIBIRSK, PIROGOVA STR., 2

nsuniversity.official

nsu24

@nsuniversity

WWW.NSU.RU/N/