

Разработка и реализация веб-среды визуальной разработки фрагментированных программ

Автор: Ижицкий Р. Л.
Научный руководитель: Маркова В. П., ИВМиМГ СО РАН
Соруководитель: Киреев С. Е., ИВМиМГ СО РАН

Новосибирск

26.06.2019 г.

Введение

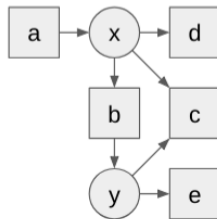
- Проблема повышения уровня программирования
 - Особо сложная для параллельного программирования
- Проект “Технология фрагментированного программирования”
 - Цель: повышение уровня параллельного программирования
 - Система фрагментированного программирования LuNA и язык LuNA

Язык LuNA

- Алгоритм, записанный в системе LuNA, называется фрагментированным
- Элементы алгоритма — множество переменных единственного присваивания и операций единственного срабатывания, связанных отношениями in и out
- Исполнительная семантика — data flow (по готовности данных)
- Можно представить графом информационных зависимостей

```
df x, y;
```

```
cf a: func1(out: x)  
cf b: func2(in: x, out: y)  
cf c: func3(in: x, y)  
cf d: func4(in: x)  
cf e: func4(in: y)
```



Проблема

Низкая доступность системы LuNA внешним пользователям

- Требуется установка системы LuNA
- Существует определенный порог вхождения в язык LuNA

Цель и задачи

Цель: создание веб-среды для визуальной разработки LuNA-программ

Задачи:

- Разработка визуального языка на базе языка LuNA
- Реализация веб-среды визуальной разработки (Web IDE)
- Реализация транслятора из визуального представления в текстовое

Проблемы разработки визуальных data flow языков

- Выбор компонентов языка
 - Представление циклов
 - Представление структур данных
- Представление большого программного кода

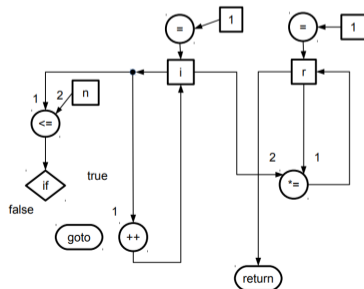
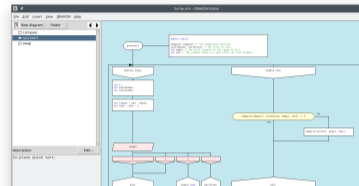
Родственные работы

Примеры существующих визуальных языков

- DRAKON (control flow)
- LabVIEW (data flow)
- Simulink (data flow)
- Pifagor (data flow)

Недостатки

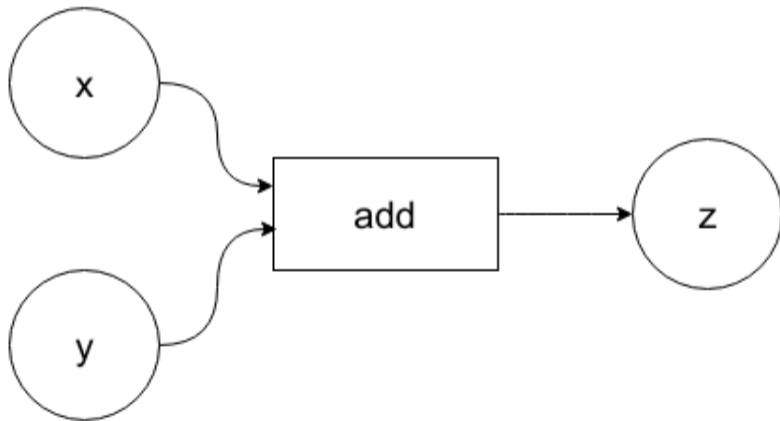
- В control flow нет элемента отражающего данные
- Семантика существующих data flow языков далека от семантики языка LuNA



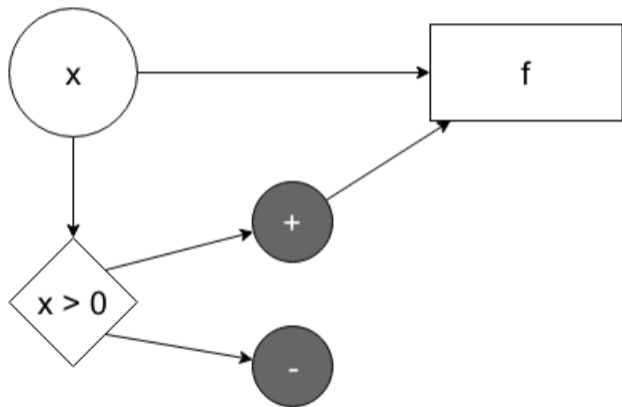
Компоненты визуального языка

- Базовые операции
 - Переменные
 - Операции
 - Информационные связи
- Условия
 - Блок условия
 - Индикаторы условия
- Циклы
 - Блок начала цикла
 - Блок границы цикла
 - Отношение связности
- Структуры данных
 - Операции сборки и разборки структур данных

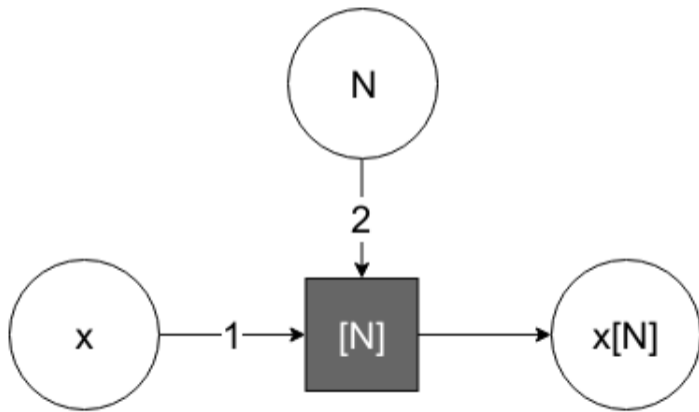
Переменные и операции



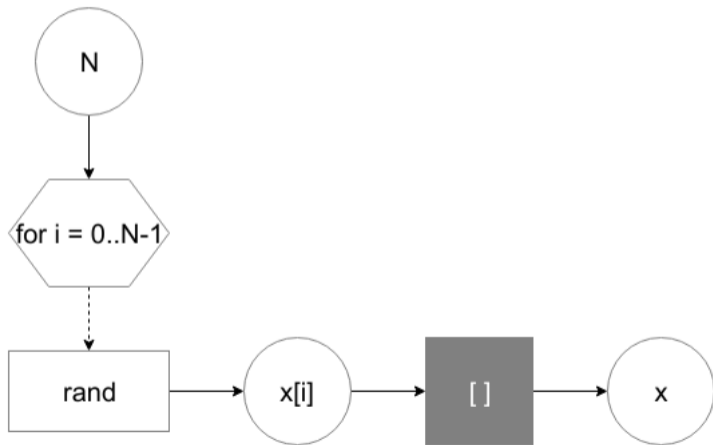
Условные операторы



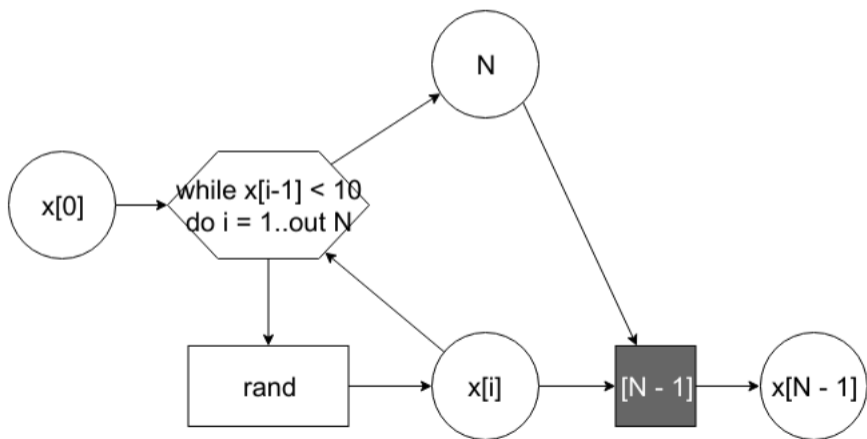
Структуры данных



Циклы for



Циклы while



Выбор средств разработки веб-среды

- Язык: JavaScript
- Выбор библиотеки для рисования графов. Критерии:
 - Возможность рисовать заданное графическое представление фрагментированного алгоритма, спроектированное на основе языка LuNA
 - Бесплатность
 - Возможность динамического редактирования графов
 - Наличие развитой документации библиотеки

Выбор библиотеки для рисования графов

JointJS	Rappid	GoJS	Mindfusion Diagram	JsPlumb
jsUML2	Nomnoml	State.js	Cytoscape.js	Mermaid.js
Raphael	Fabric.js	Paper.js	p5.js	dagre-d3
D3	vis.js	Draw2D	Diagram.js	MxGraph

Не свободно распространяемые

Невозможность рисовать заданное графическое представление

Невозможность динамического редактирования графов

Векторный редактор графических примитивов

Неудобный интерфейс для пользователя (Создание вершин и ребер)

Плохая документация

Выбранная библиотека - MxGraph

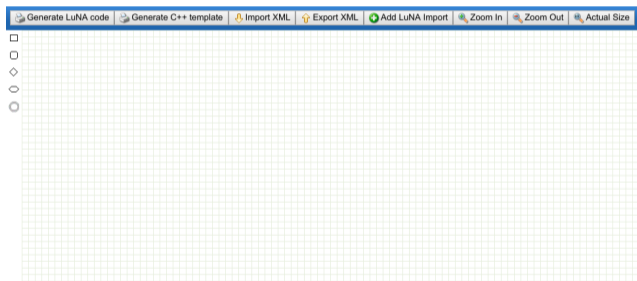
Внутреннее представление визуальных компонентов

- Разработано собственное представление компонентов визуального языка
 - XML-элементы: Variable, Operation, Condition, SubConditionPositive, SubConditionNegative, LoopIn, LoopOut
 - Атрибуты XML-элементов
- Пример XML-элемента:

```
<Operation name="sum" addName="xPlusY" id="4">  
  <mxCell style="" vertex="1" parent="1">  
    <mxGeometry x="5" y="8" width="8" height="4" as="geometry"/>  
  </mxCell>  
</Operation>
```

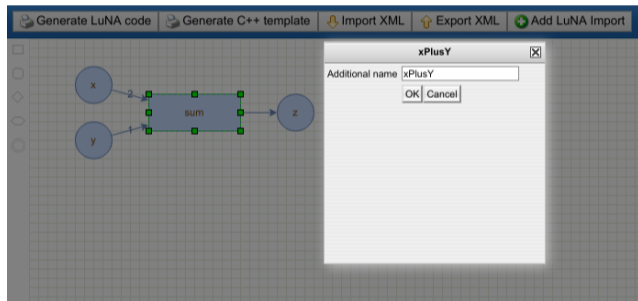

Компоненты IDE

- Меню с командами
- Панель компонентов языка
- Холст для редактирования алгоритма
- Окно с атрибутами компонентов
- Окно со списком внешних подпрограмм



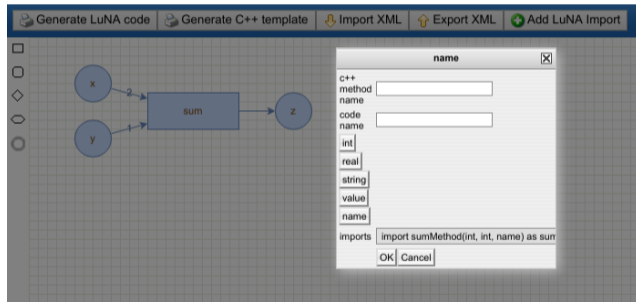
Компоненты IDE

- Меню с командами
- Панель компонентов языка
- Холст для редактирования алгоритма
- **Окно с атрибутами компонентов**
- Окно со списком внешних подпрограмм



Элементы IDE

- Меню с командами
- Панель компонентов языка
- Холст для редактирования алгоритма
- Окно с атрибутами компонентов
- Окно со списком внешних подпрограмм



Генерация LuNA-программы

- Генератор LuNA-кода
 - Вход: внутреннее библиотечное представление алгоритма
 - Выход: LuNA-код (структура разработанного алгоритма)
- Генератор кода на C++
 - Вход: список подпрограмм, задаваемых пользователем веб-среды
 - Выход: шаблон программы на C++

Алгоритм генерации LuNA-программы

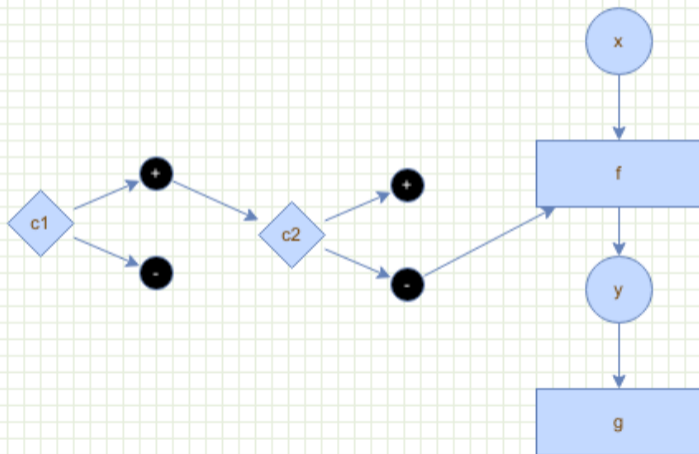
1. Получение списка всех элементов (вершин и ребер)
2. Для каждого стартового элемента цикла генерация кода этого цикла (с пометкой посещенных элементов)
 - Обход каждого цикла в глубину по прямым дугам, обрабатывая только операции
3. Генерация кода для непомеченных операций

Генерация кода для операций:

- 1 Для операции ищутся все условные операторы, от которых она зависит, и при их наличии формируется код условия
- 2 Генерация кода по атрибутам, входным и выходным переменным

Для всех переменных генерируется их объявление

Пример: Условные операторы



```

sub main() {
  df y;
  if (!c2 && c1) {
    f(#in x, #out y);
  }
  g(#in y);
}
    
```


Пример: скалярное произведение

```
sub main() {  
  df ab, s[0], s[N], N, a, b;  
  for i = 0..N-1{  
    initB(#in i, #out b[i]);  
    mul(#in i, a[i], b[i], #out ab[i]);  
    initA(#in i, #out a[i]);  
  }  
  for i = 0..N-1{  
    sum(#in s[i], ab[i], i, #out s[i+1]);  
  }  
  init0(#in #out s[0]);  
  print(#in s[N]);  
}
```

```
#include "ucenv/ucenv.h"  
extern "C"  
void c_initA(int i0,  
             OutputDF& dfo0) {  
  dfo0.setValue<int>(0);  
}  
  
extern "C"  
void c_sum(int i0, int i1,  
           int i2, OutputDF& dfo0) {  
  dfo0.setValue<int>(0);  
}
```


Результаты

- Создан прототип визуального языка, позволяющий разрабатывать ограниченный подкласс фрагментированных алгоритмов
- Создан прототип веб-среды разработки, позволяющий:
 - Рисовать и редактировать фрагментированные алгоритмы на разработанном языке
 - Транслировать графическое представление фрагментированного алгоритма в текст программы на языке LuNA и в шаблон модуля на языке C++
 - Сохранять и загружать визуальные программы на разработанном визуальном языке

Планы

- Реализация вложенных циклов
- Реализация while-циклов
- Проверка на ошибки
- Возможность запуска кода из IDE
- Повышение удобства интерфейса

Апробация работы

Работа была представлена на 57-й Международной научной студенческой конференции «Студент и научно-технический прогресс» (МНСК-2019) в виде устного доклада

Спасибо за внимание

- Репозиторий проекта: <https://github.com/ksilobait/Graph-LUNA>
- Контакты:
 - ksilobait@gmail.com (Ижицкий Руслан Леонидович)
 - kireev@ssd.sccc.ru (Киреев Сергей Евгеньевич)