Разработка web-приложения, обеспечивающего построение графовых моделей алгоритмов обработки данных

Место проведения: *МГТУ им. Баумана*
Продолжительность: *5 минут*
Научный руководитель: *Соколов Александр Павлович*

Журавлев Николай Вадимович, студент

zhuravlevnv@student.bmstu.ru

МГТУ им. Н.Э. Баумана

Россия, Москва, 2022.02.15 – 22 июня 2023 г.

Содержание доклада

- Часто реализация **алгоритмов обработки данных** предъявляет высокие требования к квалификации специалиста (в области разработки ПО и математического моделирования) и вычислительным ресурсам.
- Одним из известных подходов для преодоления указанных трудностей является **потоко-ориентированное программирование** (flow-based programming)[1].
- В работе предложено использовать графоориентированный подход[2], созданный изначально для реализации сложных вычислительных методов (СВМ).

[1] Morrison J. Paul. Flow-Based Programming, 2nd Edition: A New Approach to Application Development. CreateSpace Independent Publishing Platform, 2010. P. 370.

[2] Соколов А.П., Першин А.Ю. Графоориентированный программный каркас для реализации сложных вычислительных методов // Программирование. 2019. Т. 47, № 5. С. 43–55.

- Часто реализация алгоритмов обработки данных предъявляет высокие требования к квалификации специалиста (в области разработки ПО и математического моделирования) и вычислительным ресурсам.
- Одним из известных подходов для преодоления указанных трудностей является потоко-ориентированное программирование (flow-based programming)[1].
- В работе предложено использовать **графоориентированный подход**[2], созданный изначально для реализации сложных вычислительных методов (СВМ).

[1] Morrison J. Paul. Flow-Based Programming, 2nd Edition: A New Approach to Application Development. CreateSpace Independent Publishing Platform, 2010. P. 370.

[2] Соколов А.П., Першин А.Ю. Графоориентированный программный каркас для реализации сложных вычислительных методов // Программирование. 2019. Т. 47, № 5. С. 43–55.

Цель разработки

Создать web-приложение для построение, хранения и накопления графовых моделей описывающих процесс обработки данных

Задачи разработки

1. Провести аналитический обзор литературы в области алгоритмов визуализации и обхода графов.
2. Разработать архитектуру и программную реализацию web-приложения.
3. Провести тестирование работоспособности созданного web-приложения.

- **Вершины** – множество состояний данных СВМ S_i, каждому из которых соответствует момент успешного завершения очередной функции перехода (процедуры обработки данных).
- **Ребрам** сопоставлены функции перехода F_{ij} (Рис. 1).
- H - функция-селектор[3].

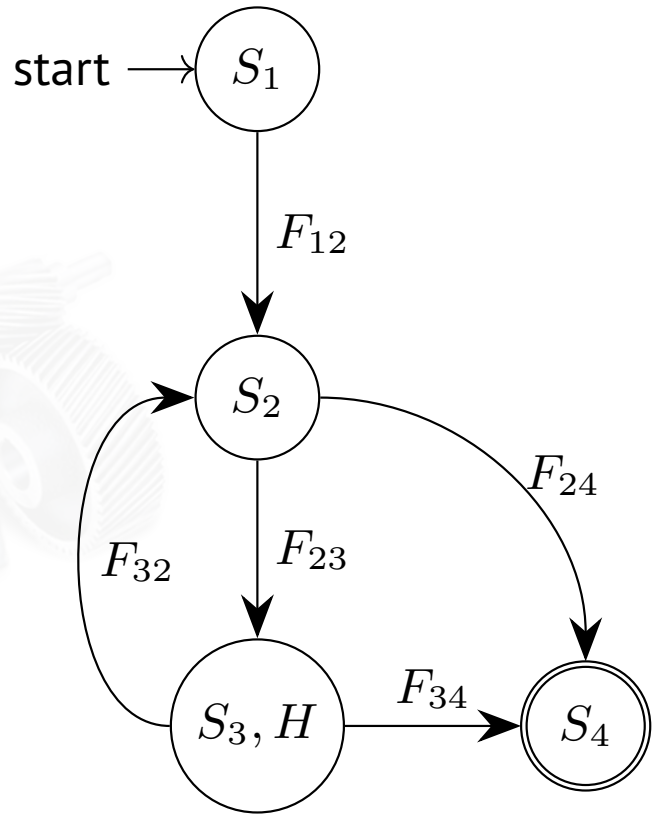

Рис. 1: Пример графовой модели

[3] Соколов А.П., Першин А.Ю. Система автоматизированного проектирования композиционных материалов. Часть 2: вычислительная под система, распределенные вычисления с применением графоориентированного подхода // Известия СПбГЭТУ «ЛЭТИ». 2020. № 10. С. 49–63.

Постановка задачи

↳ Анализ алгоритмов визуализации графов

1. Силовые алгоритмы визуализации графов[4].

   Вершины размещают на плоскости случайно, затем их представляют как частицы, на которые действуют «силы» от других частиц и они перемещаются за счёт этого.

2. Послойное рисование графа (**использовался в настоящей работе**).

   Выделяют слои и на каждый слой помещают вершину, которая соседствует с вершиной на предыдущем слое.

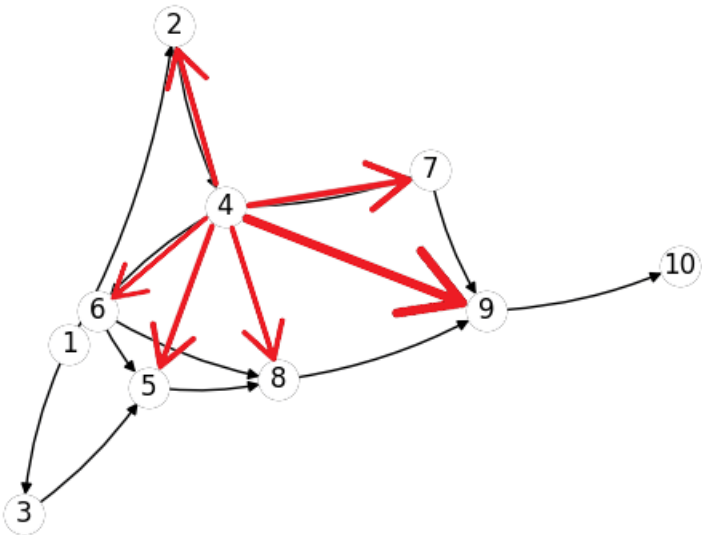

Рис. 2: Применение силового алгоритма

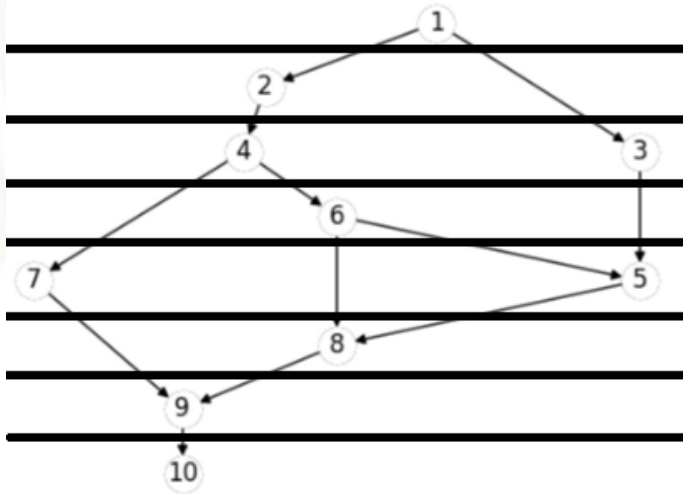

Рис. 3: Применение алгоритма послойного рисования

[4] Пупырев С.Н., Тихонов А.В. Визуализация динамических графов для анализа сложных сетей. Моделирование и анализ информационных систем. 2010;17(1):117-135.

Постановка задачи
↳ Анализ алгоритмов обхода графов

1. Общий алгоритм обхода графа с запоминанием дуг [5].

   Берут начальную вершину, помечают как пройденную. Затем для каждой выходящей из неё дуги проверяют, если вершина конца дуги не помечена, то обход по дугам приостанавливается, вершина помечается пройденной и для неё процесс повторяется.
2. Общий алгоритм обхода графа с запоминанием вершин.

   Кладут в стек начальную вершину и помечают как пройденную. Затем вершина берётся из стека и для каждой вершины, выходящей из неё, проверяют, если она не помечена, то заносится в стек. Затем процесс повторяется, пока стек не опустеет.

Оба алгоритма не используются в настоящей работе, так как в них отсутствует возможность зайти в вершину дважды.

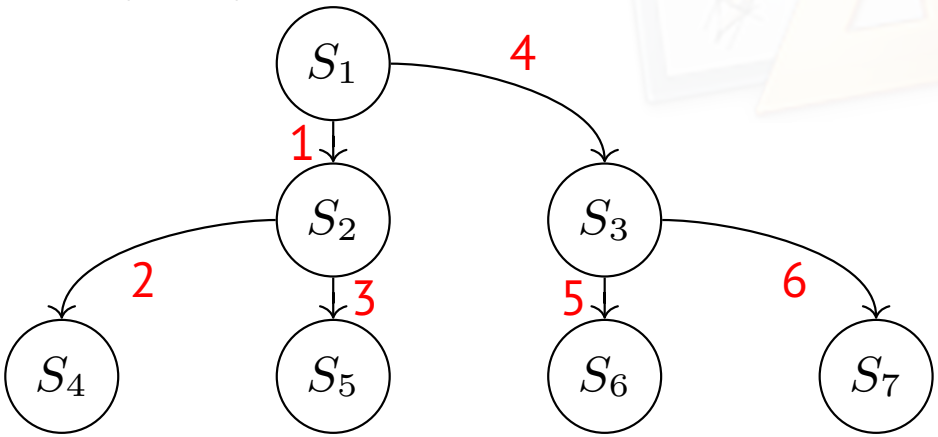

Рис. 4: Обход графа алгоритмом с запоминанием дуг

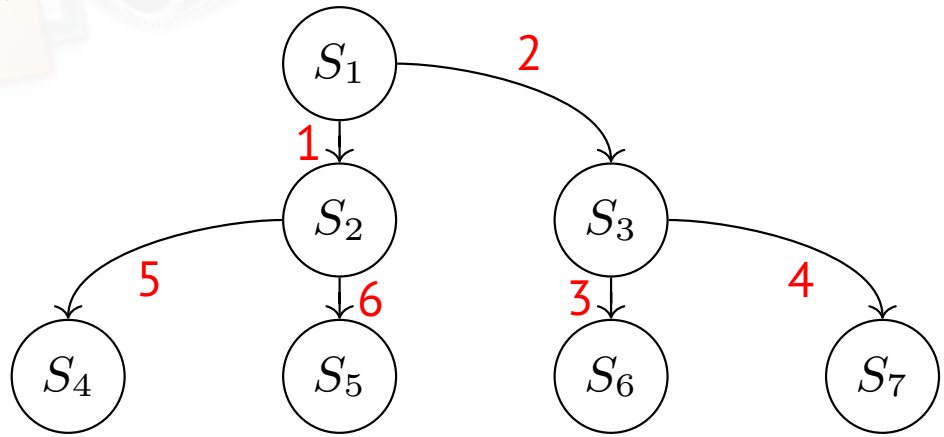

Рис. 5: Обход графа алгоритмом с запоминанием вершин

[5] Евстигнеев В.А., Касьянов В.Н. Графы в программировании. 2003. С. 100 – 125.

Должны быть обеспечены следующие функциональные возможности.

1. Создание графовой модели с «нуля».
2. Загрузка графовой модели из файла в формате aDOT (Листинг 1).
3. Сохранение графовой модели в файл в формате aDOT.
4. Обход графовой модели.

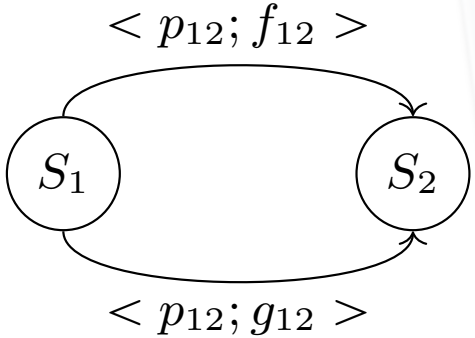

Рис. 6: Простая графовая модель

Листинг 1: aDOT-представление

```
digraph graph_model
{
 F [module="Lib", function="f_12"]
 G [module="Lib", function="g_12"]
 PRD [module="Lib", function="p_12"]
 F_12 [predicate=PRD, function=F]
 G_12 [predicate=PRD, function=G]
 BEGIN -> S_1
 S_1 -> S_2 [morphism=F_12]
 S_1 -> S_2 [morphism=G_12]
 S_2 -> END
}
```

Программная реализация
↳ Взаимодействие модулей

Модуль - это программная реализация одного из требований описанных на предыдущем слайде.

- Редактирование графа
- Открыть из файла
- Сохранение в файл
- Обход графа

После выполнения модуля всегда происходит возвращение на главную страницу.

Рис. 7: Взаимодействие модулей

- Обход идёт до встречи с первым разветвлением.
- **При обнаружении ветвления**, с помощью функции-селектора выбирается ребро и продолжается движение по нему, а остальные заносятся в стек.
- Обход продолжается, пока не встретится вершина, для которой **нужна синхронизация**, и тогда продвижение по этому ребру останавливается. Из стека берётся ребро и начинает обход по нему.

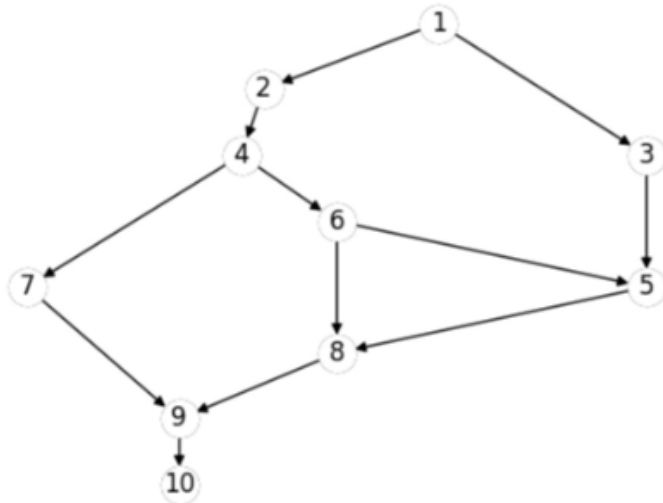

Рис. 8: Графовая модель, по которой выполняется обход

- Для тестирования обхода сделаем специальную модель данных, такую, что при каждом заходе в вершину, будем прибавлять 1 в переменную, соответствующую этой вершине.
- После обхода во всех вершинах должна быть 1 (Рис. 9).

```
2 = 1  3 = 1  4 = 1  1 = 1  5 = 1
```

Рис. 9: Результат полученный в ходе тестирования обхода графа

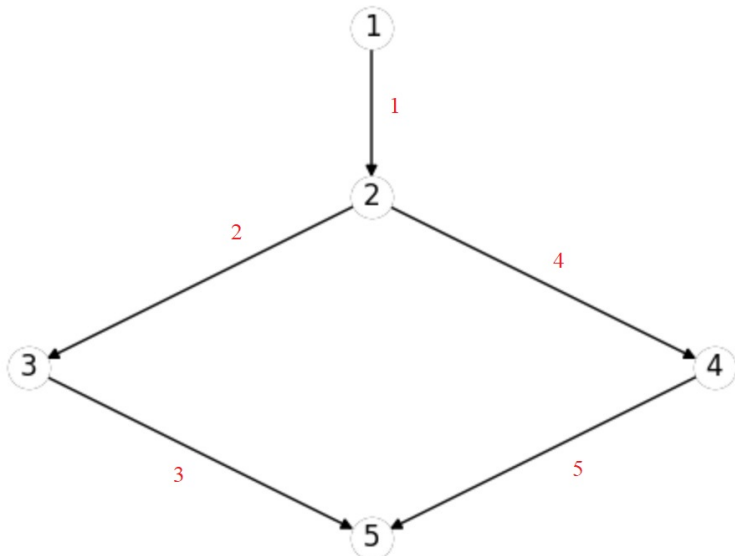

Рис. 10: Графовая модель, по которой выполняется обход

Заключение

- Был выполнен аналитический обзор литературы, по результатам которого был выбрал алгоритм визуализации и принято решение о разработке собственного алгоритма обхода.
- Было разработано web-приложения.
- Было проведено тестирование web-приложения, по результатам которого можно убедится в его работоспособности.

Спасибо за внимание!