

Академик А. П. Ершов и графы в программировании

В. Н. Касьянов

**Институт систем информатики СО РАН
имени А.П. Ершова
Email: kvn@iis.nsk.su**

План доклада

- А.П. Ершов и графы
- Первые работы
- Теория схем программ
- Теория формальных преобразований программ
- Первая монография по графам в программировании
- Инструментальные и прикладные системы
- Сибирская школа программирования
- Заключение



А.П. Ершов
(1931-1988)

А.П. Ершов и графы

- Академик Андрей Петрович Ершов (19 апреля 1931- 8 декабря 1988) был одним из советских пионеров в области системного и теоретического программирования, основателем Сибирской школы программирования. Его существенный вклад в установление информатики в качестве нового направления науки и нового феномена социальной жизни широко признан как в России, так и за рубежом.
- Новаторские работы Ершова по теории графов в программировании, рассмотренные в данном докладе, несколько менее известны, но не менее важны. Он называл графы основной конструкцией для программиста и говорил, что графы обладают огромной, неисчерпаемой изобразительной силой, соразмерной масштабу задачи программирования. А.П. Ершов внес фундаментальный вклад в теорию графов, главным образом в области программирования.

Первые работы

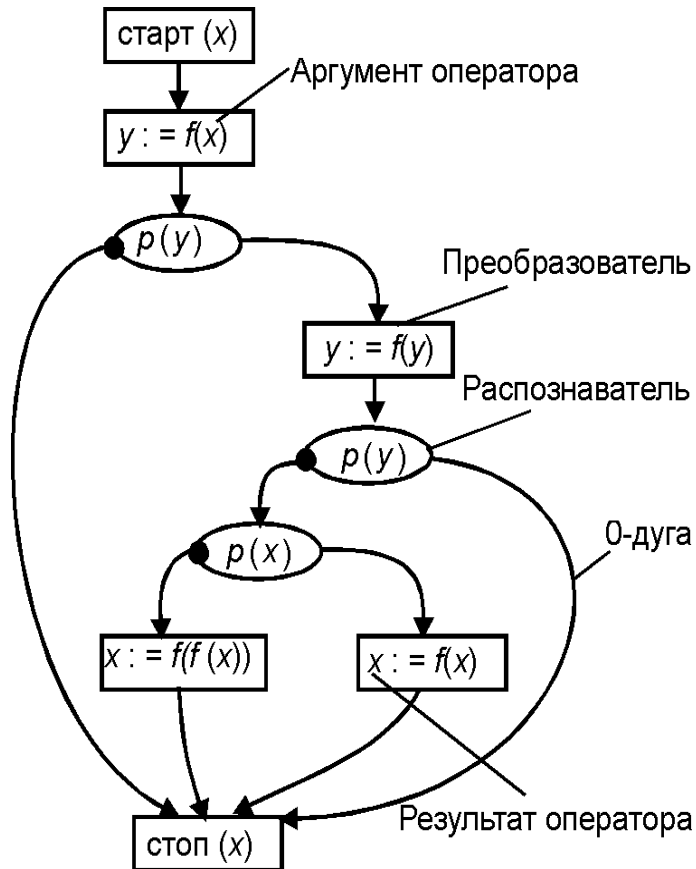
Среди первых работ, существенно использующих теоретико-графовые методы в решении задач программирования, можно отметить широко известные работы А. П. Ершова по

- организации вычисления арифметических выражений (1958 г.),
- граф-схемной модели для императивных программ в виде операторных алгоритмов (1958–1962 гг.),
- теории схем Янова с использованием их графового представления и концепции так называемой разметки (1963–1966 гг.) и
- граф-схемной теории экономии памяти (1961–1966, 1972 гг.).

Первые работы (2)

- В 1958 г. Ершов описал ставший классическим простой алгоритм определения оптимального порядка, в котором следует вычислять операторы в линейном участке, когда графовое представление луча является бинарным деревом.
- Содержательно алгоритм, обрабатывая два операнда бинарной операции, сначала работает над вычислением того из них, который требует больше регистров (является более трудным операндом). Если потребность в регистрах у обоих операндов совпадает, то каждый из операндов может быть обработан первым.
- Этот алгоритм, соответствующим образом модифицированный для учета пар регистров и другой специфики объектной ЭВМ, использовался в ряде трансляторов с языков Альфа, Алгол, Блисс и Си и стал основой известного алгоритма оптимальной генерации выражений Сети-Ульмана (1970).

Теория схем программ

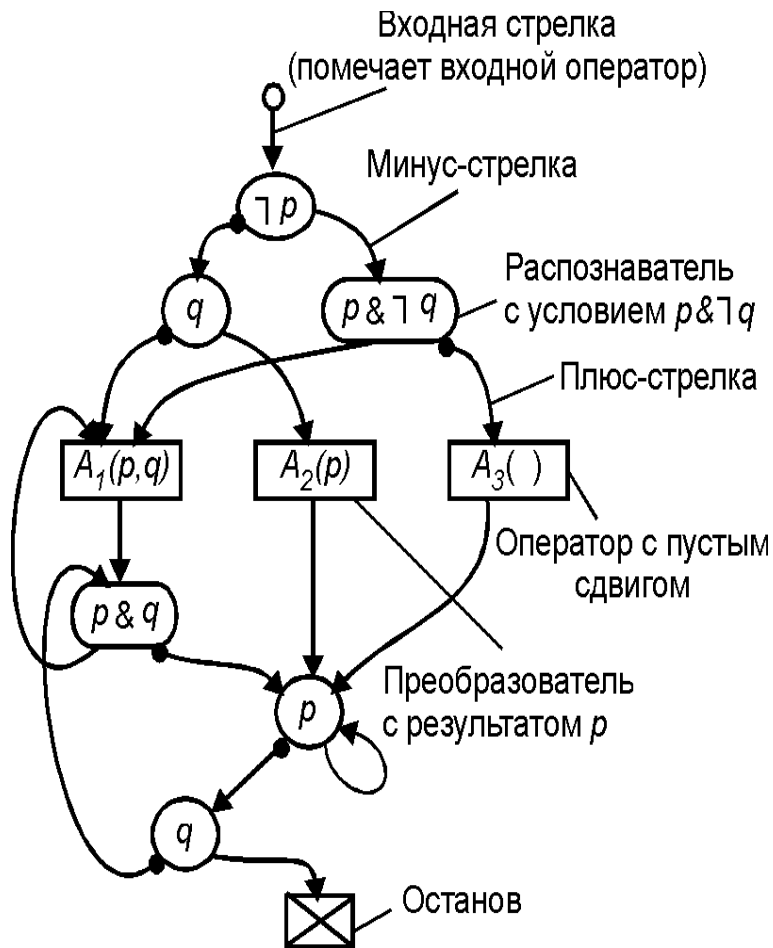


- Понятие схемы программ принадлежит А.А. Ляпунову и было введено им в 1953 г., исходя из общей концепции необходимости и возможности формализации процесса программирования.
- В опубликованных в 1958 и 1959 гг. статьях Ершова об операторных алгоритмах предлагалась модель программы, которая была основой такой известной модели, как стандартные схемы.

Теория схем программ (2)

- В работах 1958 и 1959 гг. Ершовым была установлена связь предложенного им формализма операторных алгоритмов с такими известными понятиями алгоритма, как частично-рекурсивные функции и нормальные алгоритмы Маркова.
- В последующей работе, опубликованной в 1962 г., Ершовым были рассмотрены возможности операторных алгоритмов для представления программ и логических схем программ, показана возможность построения инвариантов, общих как для некоторого машинного языка, так и для фрагмента проблемно-ориентированного языка, и позволивших строго определить отношение эквивалентности в обоих языках.
- Эти работы Ершова были одним из основных источников современной теории схем программ.

Теория формальных преобразований программ



Первой работой, посвященной общей теории преобразований программ, явилась ставшая классической статья Ю.И.Янова «О логических схемах алгоритмов» (1958), в которой для моделей программ, введенных в литературу Ляпуновым и Яновым, был найден алгоритм распознавания эквивалентности двух схем и построена полная система преобразований.

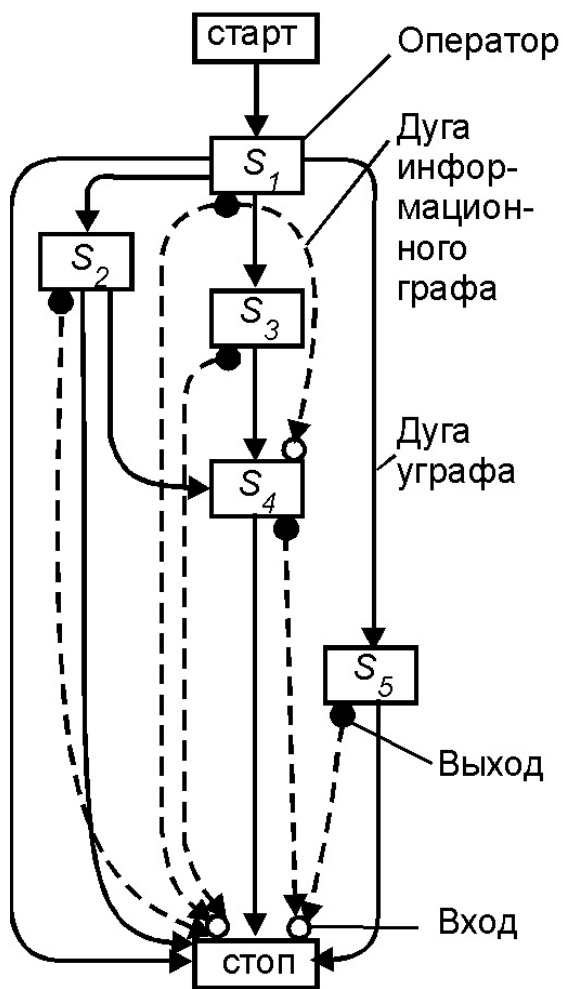
Теория формальных преобразований программ (2)

- В 1968 г. Ершов публикует статью, в которой использует для схем Янова графовое представление. Оказалось, что графовое определение является более адекватным рассматриваемой проблеме. Это нашло свое отражение в упрощении аксиоматики преобразований (14 аксиом заменились на 6), а также в эффективизации правила вывода, использующего понятие логической подчиненности, которое вместо сложного алгоритма преобразований средствами, лежащими вне аксиоматики, стало опираться на четыре аксиомы, задающие стационарную разметку дуг управляющего графа логическими функциями.
- Введенная Ершовым методика разметки активно использовалась во многих последующих работах по схемам Янова; она оказалась удобным средством для формализации преобразований, требующих для своего применения предварительного сбора некоторой информации о схеме в целом.

Теория формальных преобразований программ (3)

- Принципиальный вклад в решение задачи об экономии памяти внес С.С.Лавров в 1961 г. Он ввел понятие операторной схемы, моделирующей программы со скалярными величинами, рассмотрел различные варианты распределения памяти как эквивалентные преобразования, состоящие в переобозначении величин, ввел понятие маршрута, канонического распределения памяти и графа несовместимости. Схемы Лаврова нашли свое применение для решения широкого класса прикладных задач теории программирования — главным образом, для алгоритмов оптимизации программ.
- Сведение задачи распределения памяти для схем Лаврова к известной задаче раскраски графа, опубликованное Ершовым в 1962 г., привлекло внимание программистов к классическим задачам теории графов. В результате появилась совместная работа Ершова и Г.И. Кожухина об оценках хроматического числа связных графов (1962 г.), ставшая основой разработанного ими эвристического алгоритма близкой к оптимальной раскраски графа.

Теория формальных преобразований программ (4)



В 1968 г. Ершов публикует статью об операторных схемах над общей и распределенной памятью. В ней он вводит понятие информационного графа и вариант схем Лаврова (так называемые схемы с распределенной памятью), в которых элементами памяти являются не переменные, а входы операторов, и информационные связи между выходами и входами операторов задаются не с помощью переменных, составляющих общую память, а явно — в виде дуг информационного графа.

Теория формальных преобразований программ (5)

- Изучение схем с распределенной памятью позволило построить теорию, не зависящую от различных вариантов распределения памяти и получившую широкое применение в различных приложениях, таких как оптимизирующая трансляция и распараллеливание программ.
- Например, в последние годы в теории и практике оптимизирующей трансляции активно используется так называемая SSA-форма (Static Single-Assignment) представления программы, существенно упрощающая алгоритмы анализа и преобразования программ. По существу, SSA-форма является таким представлением программы в виде схемы над распределенной памятью, в котором каждому входу любого оператора сопоставлен ровно один выход.

Теория формальных преобразований программ (6)

- Создание общей теории распределения памяти было завершено работой по аксиоматике распределения памяти, опубликованной Ершовым в 1972 г. В ней методика разметки, впервые примененная Ершовым для схем Янова, была использована им для построения корректной и полной системы преобразований, позволяющей для любой схемы программы систематически строить любые допустимые распределения памяти для аргументов и результатов ее операторов.
- В дальнейшем данная методика в общей постановке глобальной оптимизации программ была описана Килделлом и получила существенное развитие как отдельное направление в системном и теоретическом программировании по анализу потока данных в работах Кэма и Ульмана, Грэхама и Вегмана и многих других.

Первая монография по графам в программировании



о В книге, написанной в виде беседы с читателем, Ершов демонстрирует применение графовых методов к решению задач программирования в действии, начиная с элементарной постановки задачи и кончая полным решением проблемы во всей её сложности.

о Рассмотрены две классические задачи теоретического программирования, решения которых и развитые на этих решениях методы привели к созданию теоретического программирования как самостоятельной математической дисциплины.

Первая монография по графам в программировании (2)

Часть I ЭКОНОМИЯ ПАМЯТИ В ОПЕРАТОРНЫХ СХЕМАХ

Глава 1. Содержательный анализ задачи

§ 1.1. Краткое повторение программирования § 1.2. Накопление фактов. Линейные программы § 1.3. Накопление фактов. Программы общего вида § 1.4. Накопление фактов. Подведение итогов

Глава 2. Постановка задачи и общая теория

§ 2.1. Краткое повторение математических основ
§ 2.2. Исходные определения § 2.3. Общая теория

Глава 3. Алгоритмизация

§ 3.1. Информационный граф § 3.2. Граф несовместимости
§ 3.3. Раскраска вершин графа. Общее исследование
§ 3.1. Раскраска вершин графа. Поиск алгоритма

Первая монография по графам в программировании (3)

Г л а в а 4. Реализация

§ 4.1. Вступление

§ 4.2. Структурированное программирование

§ 4.3. Общая организация экономии памяти

§ 4.4. Каноническое распределение памяти

§ 4.5. Получение графа несовместимости

§ 4.6. Раскраска вершин графа

Г л а в а 5. Заключительный анализ

§ 5.1. Связь с теорией и практикой

§ 5.2. Исторический обзор

Инструментальные и прикладные системы

- Под руководством Ершова и по его идейным проектам был создан широкий набор инструментальных и прикладных программных систем, в том числе целый ряд трансляторов и языковых процессоров.
- Созданная под руководством Ершова система Альфа (1960-1964) была первой в мировой практике оптимизирующей системой программирования, практически доказавшей возможность создания трансляторов с приемлемой эффективностью рабочих программ для языков, более сложных, чем язык Фортран.

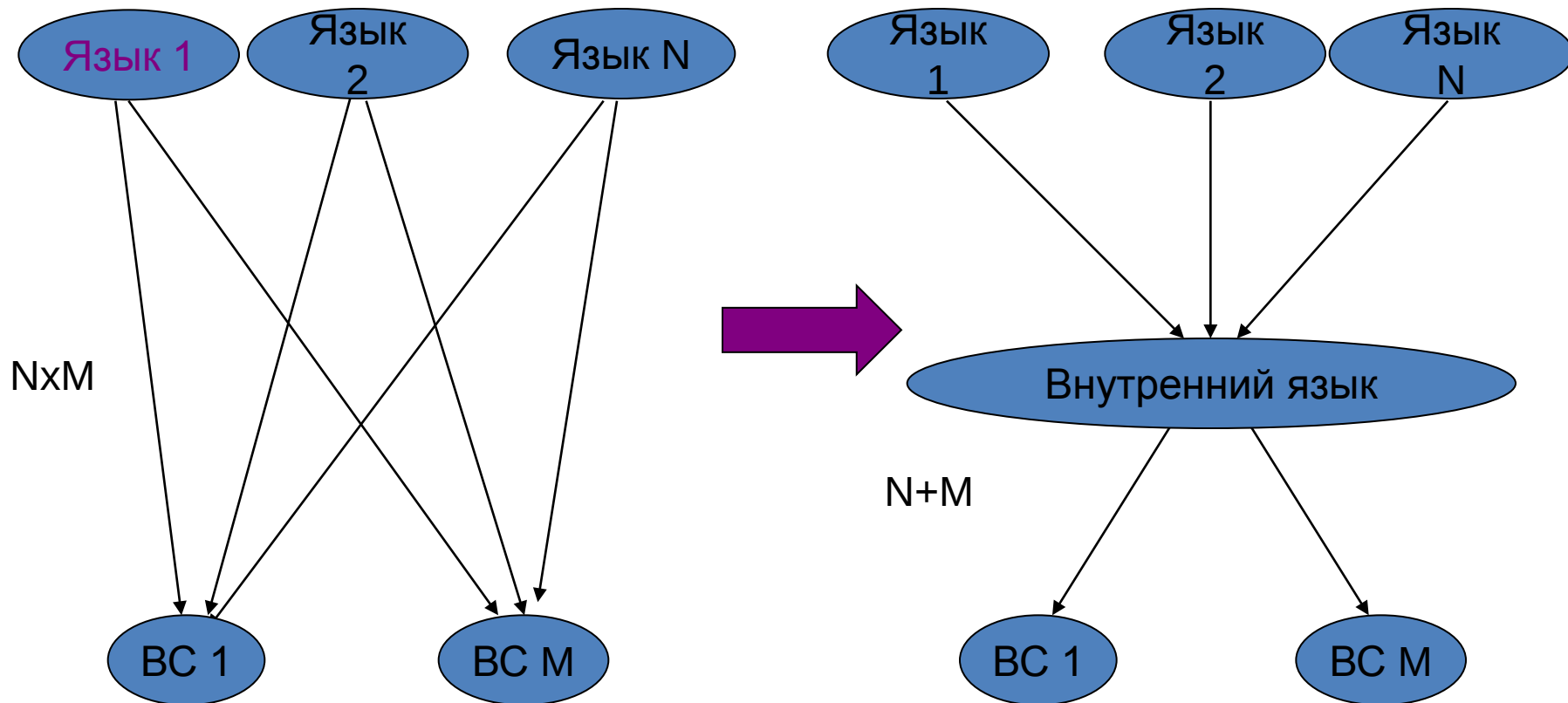
Инструментальные и прикладные системы (2)

- В системе Альфа впервые в практике трансляции были введены промежуточные схемные представления транслируемых программ, ориентированные на оптимизацию, в частности, в Альфа-трансляторе существовал специальный просмотр, на котором по программе строилась схема Лаврова для анализа информационных связей в интересах глобальной экономии памяти.
- Работы по системе Альфа с ее многопроходной схемой трансляции и оптимизирующими преобразованиями промежуточных представлений программ внесли крупный вклад в методологию оптимизирующей трансляции.
- Сам Ершов называл Альфа-транслятор «грандиозным 24-проходным прокатным станом, протягивающим транслируемую программу и перековывающим ее в эффективный объектный код через игольное ушко всего 4К слов памяти».

Инструментальные и прикладные системы (3)

- В 1971 г. Ершов публикует статьи по универсальному программирующему процессору и многоязыковой системе программирования, ориентированной на описание языков и универсальную оптимизацию, которые положили начало работы по созданию первой оптимизирующей многоязыковой транслирующей системы (проект Бета).
- Работы по проекту включали в себя исследования и эксперименты (они осуществлялись в 70-х гг.), а также собственно создание многоязыковой транслирующей системы (оно заняло первую половину 80-х гг.).
- Одной из центральных концепций типовой схемы трансляции, разработанной и реализованной в рамках проекта Бета, является концепция внутреннего (или промежуточного) языка, позволяющего с помощью универсальных алгоритмов оптимизации, выполняемых на уровне внутреннего языка, обеспечить получение качественной рабочей программы безотносительно к ее происхождению.

Инструментальные и прикладные системы (4)



Инструментальные и прикладные системы (5)

- Внутренний язык служит ядром системы Бета, экранируя фазу генерации от конкретных входных языков, а фазу декомпозиции от конкретных выходных.
- Так как все оптимизирующие преобразования осуществляются на едином языковом представлении, возможны изменения как набора преобразований, так и порядка их выполнения. Возможно полное отключение оптимизации, а также ее повторное выполнение.
- При проектировании фазы оптимизации предпочтение было отдано глобальным и универсальным преобразованиям.

Инструментальные и прикладные системы (6)

- Чтобы сохранить приемлемую сложность выполнения глобальных оптимизаций, фаза оптимизации была разбита на два последовательно работающих этапа — этап (потокowego) анализа и этап преобразований.
- На первом этапе строится такое промежуточное представление программы, которое в дополнение к конструкциям внутреннего языка содержит специальные средства (так называемые тени), ориентированные на явное описание схемных свойств транслируемой программы и позволяющие придать преобразованиям более локальный и направленный характер.
- Второй этап состоит из преобразований, изменяющих как внутреннюю программу, так и ее тени (т. е. алгоритмы преобразований корректируют те схемные свойства, которые меняются при их выполнении).

Сибирская школа программирования

- В отличие от Москвы, где, начиная с работ Ю. И. Янова, в большей степени развивался логический подход к программированию, или Киева, где в работах В. М. Глушкова и его учеников явно прослеживается приоритет алгебраических методов, Новосибирск стал центром применения теоретико-графовых моделей и методов в программировании.
- Созданная в Новосибирске академиком А. П. Ершовым школа программирования внесла значительный вклад в становление и развитие теоретического и системного программирования.

Сибирская школа программирования (2)

- Теория схем программ — одно из наиболее крупных достижений научной школы Ершова.
- На ее базе разработаны методы оптимизирующей трансляции, значительно повышающие эффективность и надежность решения задач на ЭВМ с использованием языков высокого уровня.
- Внесен существенный вклад в теорию и методологию структурного программирования и параллельной обработки, включая автоматическое распараллеливание программ.
- Разработаны эффективные алгоритмы анализа, верификации и преобразования программ и систем на базе теоретико-графовых и сетевых моделей.
- Получены крупные результаты в разработке теории и методов конструирования качественного программного обеспечения на основе смешанных вычислений, конкретизирующих преобразований, аннотированного программирования и языков спецификаций.

Сибирская школа программирования (3)

- Органическое объединение теоретических исследований с созданием экспериментальных и прикладных программных систем, воплощающих и практически проверяющих разработанные идеи и подходы, — характерная черта работ школы Ершова.
- Созданы трансляторы и транслирующие системы (АЛЬФА, АЛГИБР, АЛЬФА-6 и др.), языки и системы программирования (Эпсилон, БЕТА, Лисп, Сетл, БАРС, Поляр и др.), операционные системы и системное наполнение прикладных систем (АИСТ-0, СОФИСТ, ЭКСЕЛЬСИОР и др.), системы анализа и преобразования программ (ТМ, ТРАП, АС, СКАТ, СПЕКТР и др.), инструменты визуализации и визуальной обработки (bCAD, VEGRAS, ALT и др.), системы искусственного интеллекта (УНИКАЛЬК, НЕМО+, СИМП, ТАО и др.), инструментальные окружения программирования (СОКРАТ) и т.д.
- Особенностью реализованных систем, помимо производственных возможностей, является их принципиальная новизна. Ряд созданных систем закладывал новые направления системного программирования.

Сибирская школа программирования (4)

- Проблема терминологии, без сомнения, является одной из основных проблем в применении теоретико-графовых методов в программировании и информатике.
- Терминология в прикладной теории графов далеко не устоялась и постоянно развивается, в том числе, в связи с активным развитием теоретико-графовых методов решения задач программирования, а также из-за постоянного расширения их применения на новые предметные области, такие как Web-графы, социальные сети, семантический Web, базы знаний, сети белок-белковых взаимодействий, библиографические сети и т. д.
- Евстигнеев В.А., Касьянов В.Н. «Толковый словарь по теории графов в информатике и программировании», Новосибирск: Наука, 1999, 288 С.
- Евстигнеев В.А., Касьянов В.Н. «Словарь по графам в информатике», Новосибирск: Сибирское Научное Издательство, 2009, 300 С.

Сибирская школа программирования (5)

Т-Нумерация — WikiGrapp

статья | обсуждение | просмотр | история

Т-Нумерация

Т-Нумерация (T-Numbering) — такая нумерация вершин уграфа, что для некоторой фиксированной его *обратной нумерации* N справедливы следующие свойства:

- (1) для любых *бывершин* p и q : $T(p) < T(q)$ тогда и только тогда, когда $N(p) < N(q)$;
- (2) T -номера вершин N -области $N[p]$ вершины p образуют отрезок $[T(p), T(p) + |N[p]| - 1]$.

М-номер Т-номер N-номер

Литература

- Касьянов В. Н., Евстигнеев В. А. Графы в программировании: обработка, визуализация и применение. — СПб.: БХВ-Петербург, 2003.
- Касьянов В. Н. Оптимизирующие преобразования программ. — М.: Наука, 1988.

Последнее изменение этой страницы: 07:48, 27 ноября 2013. К этой странице обращались 374 раза. Содержимое доступно в соответствии с GNU Free Documentation License 1.2. Политика конфиденциальности Описание Wiki GRAPP Отказ от ответственности

Powered By MediaWiki

Сибирская школа программирования (6)

- Несмотря на наличие обширной специальной литературы по решению задач на графах, широкое применение в практике программирования полученных математических результатов затруднено в силу отсутствия систематического их описания, ориентированного на программистов. Поэтому значительный класс практических задач, по существу сводящихся к простому выбору подходящего способа решения и к построению конкретных формулировок абстрактных алгоритмов, для многих программистов все еще остается полем для интеллектуальной деятельности по «переоткрытию» методов.
- Выполнен цикл работ по изучению и систематизации алгоритмов обработки, визуализации и применения графовых моделей в программировании.
- Касьянов В.Н., Евстигнеев В.А. «Графы в программировании: обработка, визуализация и применение», СПб.: БХВ-Петербург, 2003, 1104 С.

Сибирская школа программирования (7)

- Объединение как чисто графовых алгоритмов решения чисто графовых задач, так и программистских задач вместе со специфическими алгоритмами для их решения, основанными или использующими теоретико-графовые методы и алгоритмы.
- Разделение базовых алгоритмов на подклассы по типу графов, используемых в качестве модели (деревья, бесконтурные графы, сводимые графы)
- Высокоуровневое описание алгоритмов, позволяющее понять алгоритм на содержательном уровне, оценить пригодность его для решения конкретной задачи и осуществить модификацию алгоритма, не снижая степень математической достоверности окончательного варианта программы.
- Статические и динамические иллюстрации (анимация алгоритмов).

Сибирская школа программирования (8)

Скриншот веб-страницы "Коды Прюфера" на сайте WEGA.

Навигация:

- Заглавная страница
- Все страницы
- Текущие события
- Свежие правки
- Портал сообщества
- Справка

Поиск:

Инструменты:

- Ссылки сюда
- Связанные правки
- Загрузить файл
- Спецстраницы
- Версия для печати
- Постоянная ссылка

Коды Прюфера

Пусть T - дерево с множеством вершин $\{v_1, v_2, \dots, v_n\}$. Будем считать, что номер вершины v_i равен i . Сопоставим дереву T последовательность $[a_1, a_2, \dots, a_{n-2}]$ называемую **кодом Прюфера**, по следующему правилу:

```
функ КОД_ПРЮФЕРА( $T$ : дерево) =
1. Пусть  $n$  - число вершин в  $T$ ,
   а  $A$  - целочисленный вектор длины  $n-2$ ;
2.  $B := [1 : n]$ ;
3. для  $i$  от 1 до  $n-2$  цикл
4.    $b := \min\{k \in B; k \text{ - номер висячей вершины}\}$ ;
5.    $A[i] := \text{номер вершины, смежной вершине с номером } b$ ;
6.    $B := B - \{b\}$ ;
7.   Удалить из  $T$  вершину с номером  $b$ 
8.   всё
9.   возврат  $A$ 
10. всё
```

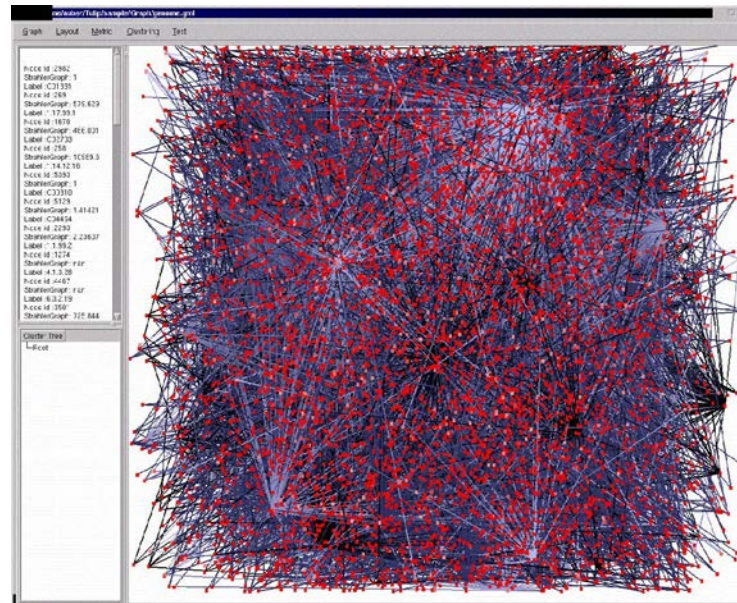
Рассмотрим седующий пример. Для дерева T (рис.) код Прюфера имеет вид:

$P_2(T) = [4, 4, 4, 5, 5, 7, 7]$.

```
graph TD
    4 --- 1
    4 --- 2
    4 --- 3
    4 --- 5
    5 --- 6
    5 --- 7
    7 --- 8
    7 --- 9
```

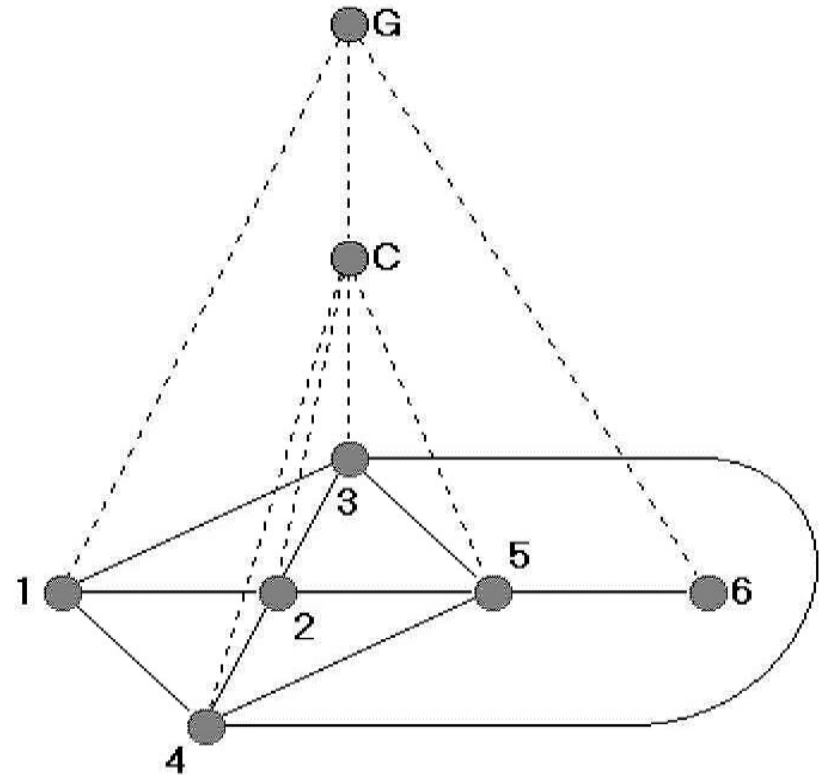
Сибирская школа программирования (9)

- Основные проблемы визуализации графов:
 - неприемлемая временная сложность,
 - ненаглядность (запутанность) получаемого изображения.
- Подходы к их решению:
 - иерархическое представление информации,
 - интерактивная визуализация.



Сибирская школа программирования (10)

- *Иерархический граф $H = (G, T)$, где G – некоторый граф, называемый *основным*, а T – корневое дерево *вложенности*, вершины которого соответствуют элементам некоторой иерархии фрагментов графа G , а дуги отражают отношение их непосредственной вложенности.*
- Графовая модель = класс атрибутированных иерархических графов + отношение эквивалентности

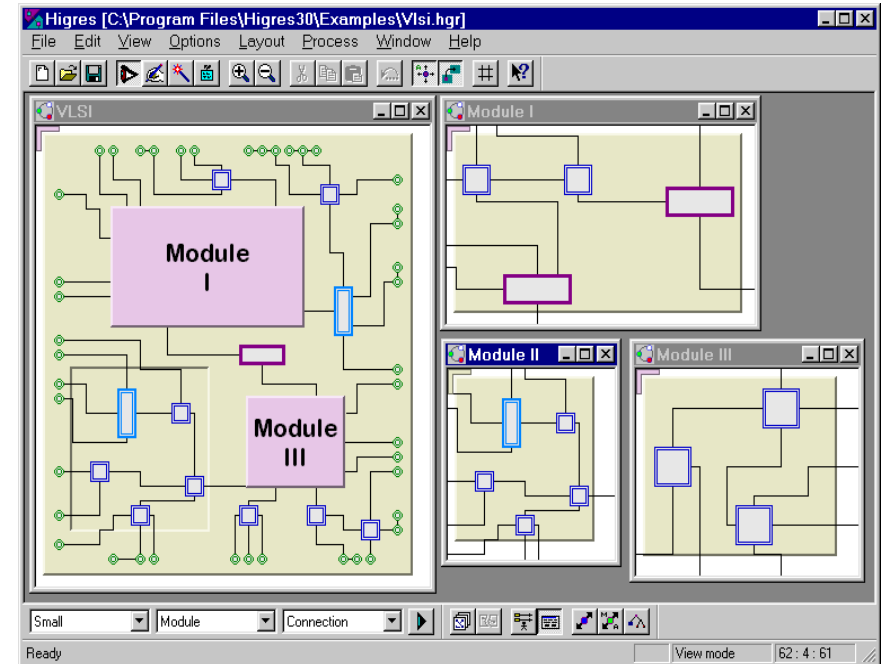
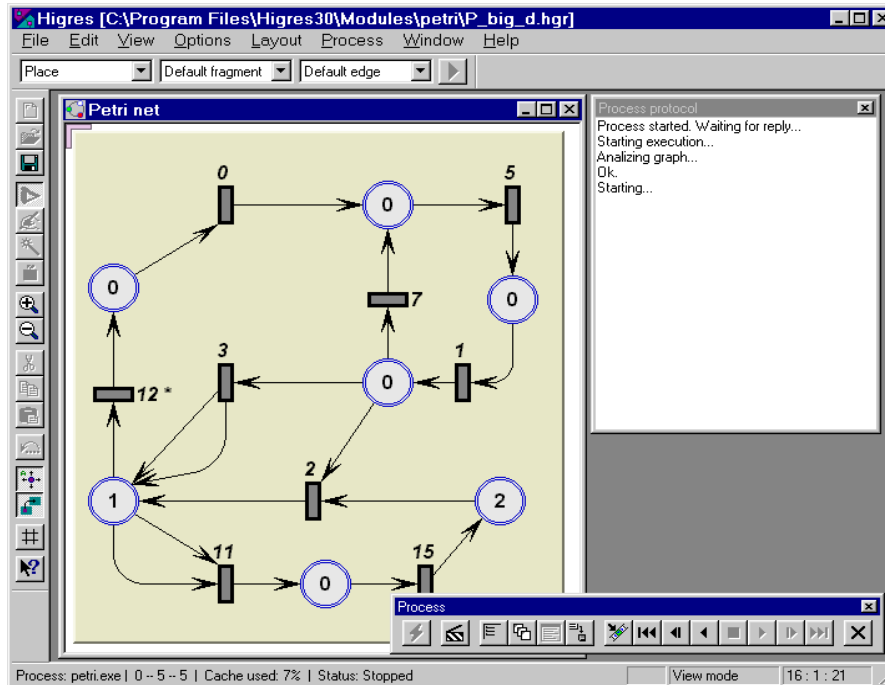


Простой иерархический граф
с двумя нетривиальными
фрагментами G и C

Сибирская школа программирования (11)

Система HIGRES

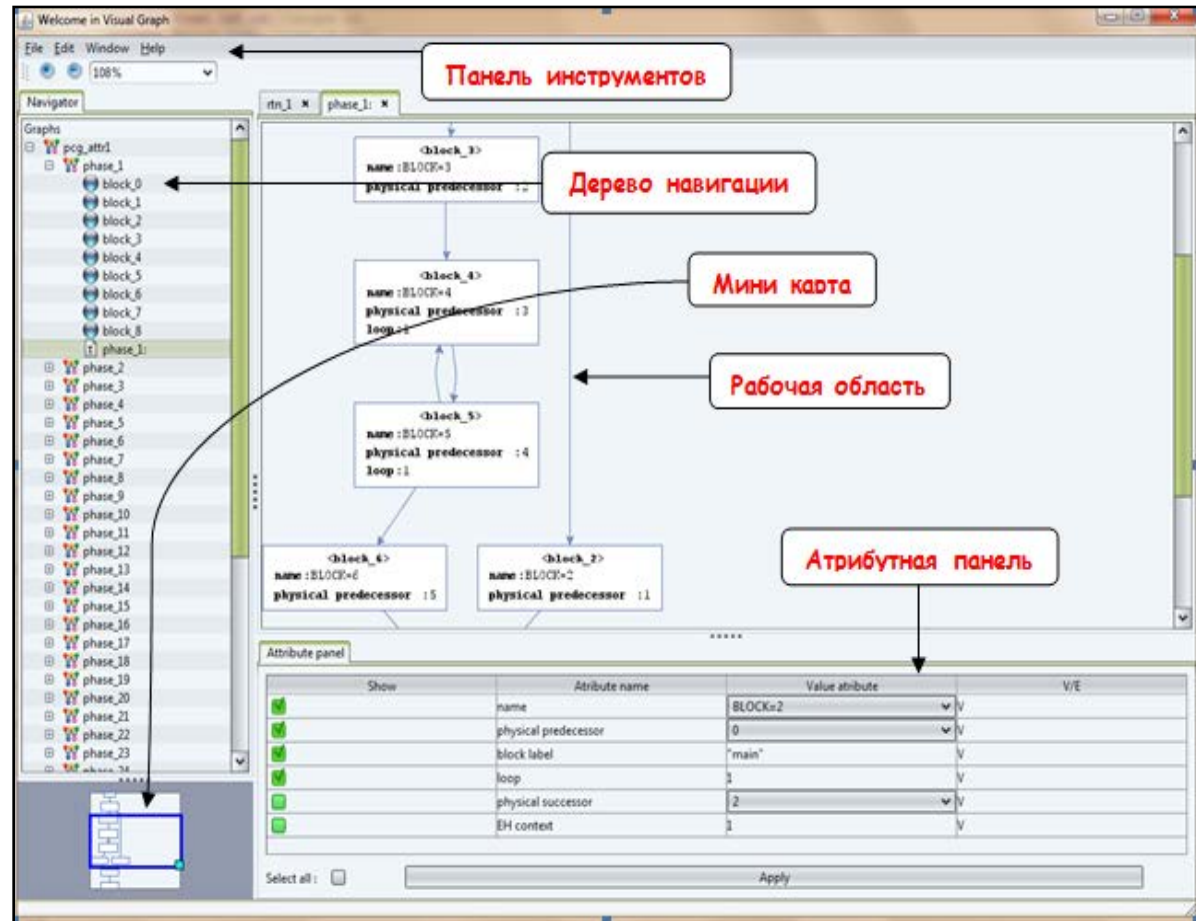
Визуализатор и редактор
простых иерархических графов
и графовых моделей



Платформа для исполнения и
анимации графовых алгоритмов

Сибирская школа программирования (12)

- Универсальная расширяемая система Visual Graph для визуализации атрибутированных иерархических графовых моделей большого размера
- Система Visual Graph поддерживает интерактивное управление визуализацией и удобную навигацию



Заключение

- Имя академика Андрея Петровича Ершова чаще всего ассоциируется со школьной информатикой. Его новаторские работы по теории графов в программировании, рассмотренные в данном докладе, несколько менее известны, но не менее важны.
- А.П.Ершов внес фундаментальный вклад в теорию графов, особенно в области программирования, и во многих аспектах определил современное состояние применения теоретико-графовых методов в программировании.
- Работы А.П.Ершова служат базой для продолжающихся в Институте систем информатики им. А. П. Ершова СО РАН исследований по теории графов в программировании.

Заключение (2)

А.П. Ершов:

«Программирование — это новый вид универсальной деятельности, при которой человек должен вложить в ЭВМ все, что видит, слышит, знает, и научить ее всему, что делает сам. Важнейшим свойством информационной модели или управляющей системы является ее структура, или говоря математическим языком, совокупность бинарных отношений на наборах элементарных единиц данных и действий. Эти структуры данных и структуры действий являются единственными ипостасями программ и обрабатываемой ими информации, в которых они могут существовать в воображении программиста во чреве компьютера. Вот почему, **графы являются основной конструкцией для программиста.**»

Спасибо за внимание !