

Абстракции и базовые операции специализированного языка описания алгоритмов решения задач структурного анализа и синтеза

12, декабрь 2013

DOI: 10.7463/1213.0656686

Иванова Г. С.

УДК 004.3 +519.1

Россия, МГТУ им. Н.Э. Баумана

gsivanova@bmstu.ru

Введение

Большая размерность и экспоненциальная вычислительная сложность решения большинства задач структурного анализа и синтеза предполагает, что программную реализацию этих алгоритмов необходимо осуществлять с использованием всех возможных способов сокращения времени их выполнения. Это требует глубокой проработки алгоритмов и текстов проектируемых программ.

Количество работ, в которых разрабатываются и анализируются новые алгоритмы решения новых и старых задач рассматриваемого класса, последние 25 лет не уменьшается. Среди последних публикаций на эту тему следует назвать работы [1-7].

Однако по-прежнему разработка, исследование и оптимизация, как самих алгоритмов решения задач структурного анализа и синтеза, так и написанных по ним программ – сложная и трудоемкая задача.

1 Обоснование необходимости разработки специализированного языка

Программы решения задач структурного анализа и синтеза можно реализовывать на различных языках. Как правило, для этого используют универсальные языки высокого уровня, предполагающие процедурную или объектную декомпозицию. В таких языках операции над графами реализуются операциями над сложными структурами данных, такими как массивы, матрицы, списки. Однако размеры и сложность исходных текстов программ указанного класса при подобном уровне детализации таковы, что не позволяют

разработчику сосредоточиться на поиске более эффективных с точки зрения вычислительной сложности алгоритмов и способов уменьшения времени выполнения реализующих их программ.

Использование же языков, основанных на иных принципах декомпозиции, сопряжено с дополнительными трудностями. Так, например, при реализации программы, выполняющей операции над графами, на языке функционального программирования Haskell [8], разработчик должен абстрагироваться от выполняемых над объектами задачи проектных операций и оперировать математическими абстракциями, напрямую не связанными с абстракциями, которые представляют объекты решаемых задач, т.е. графами. При этом также сложно концентрироваться на снижении времени выполнения программы.

Для сокращения сроков разработки и улучшения качества программ решений задач структурного анализа и синтеза на универсальных языках неоднократно предлагались и библиотеки подпрограмм, реализующие основные алгоритмы на графах, и объектные библиотеки для представления, изображения графов и реализации некоторых операций и алгоритмов над ними. Наибольший интерес из известных библиотек представляют: библиотека LEDA (Library of Efficient Data types and Algorithms) [9], разработанная в Институте Информатики Макса Планка (ФРГ), и библиотека GTL (Graph Template Library), созданная в Университете Паскаля (Франция) [10]. Однако на практике использование указанных и им подобных библиотек ограничено, поскольку они рассчитаны в основном на обыкновенные графы (не учитывают мульти-, гипер- и ультраграфов), и потому используют, как основное, представление графов через отношение смежности вершин.

В то же время большинство разработчиков алгоритмов решения задач структурного анализа и синтеза в своих работах используют псевдоязыки. Как правило, нотация этих псевдоязыков представляет собой комбинацию математической записи операций над множествами и Паскале-подобных управляющих конструкций [11-13]. Указанные нотации обеспечивают компактную запись сложных алгоритмов, понятны разработчикам алгоритмов и позволяют сосредоточиться на оптимизации последних.

В связи с вышесказанным целесообразным является создание специализированного языка с понятным разработчикам алгоритмов синтаксисом. Такой язык должен позволить:

- использовать для представления моделей структур не только отношение смежности вершина-вершина, но и отношение инцидентности вершина-ребро и/или ребро-вершина, а потому обеспечит преобразование не только графов, но и гипер- и ультраграфов;

- описывать алгоритмы в терминах абстракций, используемых при решении задач структурного анализа и синтеза математических моделей, что в десятки раз сократит объем кода исходной программы и время ее разработки, обеспечивая возможность быстрой реализации и сравнения нескольких программ для выбора варианта с лучшими временными характеристиками.

Создание нового специализированного языка с традиционной схемой компиляции «исходный код на разрабатываемом языке – объектный код на машинном языке», особенно с учетом требования максимально возможной оптимизации кода – сложная задача, решение которой потребует много сил и времени. Однако трудоемкость разработки компилятора можно снизить на несколько порядков, если выполнять трансляцию не в машинный код, а на один из существующих объектных универсальных языков программирования высокого уровня, компилятор которого уже реализует известные способы оптимизации получаемого машинного кода. Тогда схема преобразования будет выглядеть так: «исходный код на разрабатываемом языке – исходный код на универсальном языке программирования – объектный код на машинном языке».

Анализ уже существующих алгоритмов решения задач структурного анализа и синтеза, показывает, что помимо увеличения степени наглядности программы, а также ускорения ее разработки, специализированный язык должен обеспечивать:

- выбор представления графовой модели (простые и взвешенные матрицы смежности, матрицы инцидентности, простые и взвешенные отображения по соответствующим отношениям и т.д.), поскольку для различных задач и размерностей исследуемых структур применяют различные варианты представлений;
- замену структур данных для хранения представлений графов без изменения текста программы, что позволит выбрать вариант, обеспечивающий минимальное время выполнения программы [14];
- автоматическое или автоматизированное применение оптимизирующих преобразований уровня операций над множествами, в том числе указанных в [15].

Кроме того следует создать средства автоматической оценки вычислительной и емкостной сложности программы по ее тексту на специализированном языке и обеспечить наглядное сравнение временных и емкостных характеристик разных программ, решающих одну и ту же задачу.

2 Выявление базовых абстракций языка описания алгоритмов решения задач структурного анализа и синтеза

К первоочередным задачам, решаемым при разработке формального языка программирования, относят построение спецификации и обоснование синтаксиса языка, а также определение его семантики. И синтаксис и семантика языка при этом должны базироваться на основных абстракциях языка и совокупности операций над ними.

Анализ алгоритмов решения задач структурного анализа и синтеза [1-7, 11-13] показывает, что основной абстракцией языка должно служить «множество». Данная абстракция в рассматриваемой предметной области широко используется для представления графовых моделей структур: ориентированных и неориентированных графов, мультиграфов, гиперграфов и ультраграфов. Соответственно почти все преобразования графовых моделей реализуются как преобразование соответствующих множеств вершин и ребер и/или их отображений.

Поскольку разрабатываемый язык описания алгоритмов является специализированным и предназначен для решения задач структурного анализа и синтеза, для множеств вершин и ребер всегда существуют универсумы, т.е. все возможные элементы этих множеств известны и их количество ограничено. Элементом множества в этом случае является число (номер узла, вершины, дуги или ребра), соответствующее номеру описания этого элемента в универсуме.

Помимо множеств совокупность абстракций, используемых для представления графовых моделей, должна включать [16]: мультимножества – для представления отображений «вершины-вершины» мультиграфов, а также гипер- и ультраграфов с повторением вершин в ребре, кортежи – для задания ребер с отношением порядка в ориентированном гиперграфе и ультраграфе, а также путей и цепей, вектора – для хранения весов элементов графовых моделей и матрицы – для задания матриц смежности и инцидентности.

Причем в отличие от принятого в программировании представления перечисленных выше абстракций, как структурных типов данных, в данном случае абстракции целесообразно рассматривать, как базовые типы, поскольку это позволит организовать их эффективную реализацию [14].

Остальные используемые при описании алгоритмов абстракции, такие как множество множеств, вектор векторов и т. п., будем рассматривать как сложные, т.е. построенные из базовых. Это позволит применять для их обработки те же операции, что и для базовых абстракций, но с учетом того, что элементами сложных абстракций являются другие абстракции.

Следует также иметь в виду, что элементы любой поддерживаемой на данном уровне абстракции в конкретной задаче могут иметь один или более весов различных типов (целые, вещественные и булевские).

Для представления базовых абстраций языка, включающих элементарные компоненты, такие как элементы множества, в программе должен использоваться определяемый программистом тип данных. При описании этого типа должны быть заданы следующие атрибуты.

1. *Количество* компонентов, которое определяет размерность входа задачи и используется при выделении необходимой памяти и на этапе анализа для получения оценок эффективности алгоритма.

2. *Тип* каждого компонента, определяющий объем памяти, который отводится под компонент, и множество возможных операций над ним.

3. *Имена*, используемые для доступа к конкретным компонентам.

4. Для всех абстракций кроме универсума следует также указывать *отношение к другим абстракциям*, что позволит проверять совместимость типов для выполнения операций над ними. Для универсума эта информация неактуальна, поскольку физически универсум – это последовательность целых чисел, которая непосредственно в программе, как правило, не обрабатывается. Однако отношение к одному универсуму, например универсуму вершин – позволит проверять корректность выполнения над множествами операции объединения, пересечения и т.п.

Синтаксис представления абстракции, включающий указанные атрибуты, следует строить с учетом принятого в математике описания тех же или аналогичных абстракций. Это позволит сделать сложные конструкции описаний более понятными и компактными.

Так универсум предлагается описывать конструкцией вида:

$$Universe \ X = \{x[i] / i = 1, n\}, x[i] \in Int,$$

а множество – конструкцией вида:

$$Set \ Z = \{z[i] / i = 1, k\}, Z \subseteq X.$$

Описание конструкций в БНФ при этом выглядит следующим образом:

$\langle \text{Описание универсума} \rangle ::= \text{Universe } \langle \text{Имя} \rangle = \langle \text{Описание размерности множества} \rangle,$
 $\langle \text{Описание элемента универсума} \rangle$
 $\langle \text{Описание множества} \rangle ::= \text{Set } \langle \text{Имя} \rangle = \langle \text{Описание размерности множества} \rangle,$
 $\langle \text{Описание элемента множества} \rangle$
 $\langle \text{Описание размерности множества} \rangle ::= \{ \langle \text{Элемент} \rangle / \langle \text{Размерность} \rangle \}$
 $\langle \text{Элемент} \rangle ::= \langle \text{Имя} \rangle [\langle \text{Индекс} \rangle]$
 $\langle \text{Размерность} \rangle ::= \langle \text{Индекс} \rangle = 1, \langle \text{Мощность множества} \rangle$
 $\langle \text{Описание элемента универсума} \rangle ::= \langle \text{Элемент} \rangle \in \text{Int}$
 $\langle \text{Описание элемента множества} \rangle ::= \langle \text{Имя} \rangle \subseteq \langle \text{Имя} \rangle$

Для каждой абстракции необходимо определить множество элементарных и сложных операций. Элементарные операции должны быть реализованы в языке непосредственно, а сложные – через элементарные (возможно – в виде подпрограмм).

3 Определение совокупности операций над абстракциями языка решения задач структурного анализа и синтеза

3.1 Математические операции

Из совокупности операций на множествах, определенных в математике, в алгоритмах рассматриваемой предметной области используются:

- проверка вхождения элемента во множество;
- определение мощности множества;
- проверка эквивалентности (и не эквивалентности) двух множеств;
- проверка вхождения (строгого и нестрогого) одного множества в другое;
- пересечение множеств;
- объединение множеств (объединение множества и элемента);
- конкатенация множеств (конкатенация множества и элемента);
- дополнение одного множества до другого (дополнение множества до элемента);
- симметрическая разность множеств;
- абсолютное дополнение множества до универсума.

Операции над мультимножествами в математике встречаются существенно реже, но обычно предполагают, что операции над множествами применимы к мультимножествам. Однако, элементы мультимножества, помимо n -арного отношения вхождения R_n , связаны еще и n -арным отношением кратности Q_n . Следовательно, помимо множественных операций, над мультимножествами в языке должны быть определены специфические операции, связанные с отношением кратности:

- определение базового множества мультимножества;

- определение количества повторений элемента мультимножества.

Кортеж в математике определяют как «слово над множеством», т.е. как последовательность или строку, состоящую из элементов множества.

Помимо того, что элементы кортежа могут повторяться, как в мультимножествах, они по определению упорядочены, т.е. на них дополнительно наложено отношение порядка или бинарное отношение следования. Соответственно для кортежей должны быть введены операции определения номера элемента в кортеже, замены элемента с указанным номером, добавления (вставки) элемента в заданное место, удаления элемента с заданного места со сдвигом остальных элементов, а также описанные выше операции, связанные с реализующими абстракциями.

Операции пересечения, объединения, дополнения, симметрической разности, абсолютного дополнения до универсума, проверки эквивалентности (неэквивалентности), проверки вхождения (строгого и нестрогого) определены для пар множеств (мультимножеств) и, соответственно, их целесообразно рассматривать как сложные, т. е. представимые через элементарные операции над множествами. Эти операции широко используются при описании операций над графовыми моделями, а потому они должны быть определены в языке в виде подпрограмм.

В связи с тем, что в программах решения задач структурного анализа и синтеза вектора и матрицы используют в основном для хранения весов вершин и ребер, специфические операции над ними, как правило, не выполняют. Таким образом необходимость реализовывать, например умножение матриц и векторов, отсутствует.

3.2 Операции, связанные с реализацией абстракций

Основной набор операций над множествами, мультимножествами и кортежами в математике разрабатывался для формулирования условий, а не действий, поэтому для языка описания алгоритмов этих операций недостаточно.

Помимо математических операций над указанными абстракциями в языке необходимо также иметь операции, связанные с реализацией этих абстракций в памяти, при которой на элементы множеств и мультимножеств неявно накладывается отношение порядка. Так при представлении линейными структурами данных, элементы множественных абстракций (множества, мультимножества, кортежи) доступны в порядке добавления элементов к абстракции (при использовании стека – обратном порядку добавления элементов). При представлении множественных абстракций характеристическими векторами порядок элементов определяется последовательностью

их задания в универсуме. При представлении деревьями – реализованным алгоритмом просмотра вершин дерева (обычно «слева направо»).

Необходимость описания действий, проявляется, например, в том, что часто встречающаяся операция поиска элемента с определенными свойствами не может быть заменена операцией проверки принадлежности, потому что результатом операции проверки принадлежности является значение «истина» или «ложь», а результатом поиска элемента должен быть его номер (или адрес) в памяти, который необходим для выполнения дальнейшей обработки этого элемента.

Аналогично операция сортировки элементов меняет порядок их просмотра, связанный с физическим размещением этих элементов в памяти. Заменить операцию сортировки описанием того, что элементы множества (мультимножества) должны быть упорядочены также невозможно.

Фактически перечисленные выше операции при описании алгоритма выполняются не над множествами или мультимножествами и их элементами, а над абстракциями, реализующими эти понятия в памяти ЭВМ. Соответственно, помимо обычно употребляемых операций над множествами и мультимножествами, необходимо предусмотреть операции чтения и записи элементов по номеру в неявно заданном порядке просмотра, операции поиска заданного элемента по значению, поиска максимального или минимального элемента, сортировки множеств и мультимножеств, а также другие операции, подразумевающие действия с конкретными элементами. Кроме того, необходимо определить операции для взвешенных множественных абстракций, например, операции определения веса конкретного (минимального, максимального или заданного) элемента множества и сортировки множеств в соответствии с указанным весом.

Вектора при решении задач структурного анализа и синтеза в отличие от множеств, мультимножеств и кортежей могут содержать не номера элементов графов в соответствующих универсумах, а их веса различных типов или номера и веса одновременно. Элементы вектора находятся только в отношении порядка R , в соответствии с которым каждому из них сопоставлен номер (индекс). Кроме основной операции обращения по номеру для векторов должны быть определены операции поиска элемента с заданными свойствами (максимально, минимального и с заданным значением) и поиска весов соответствующих элементов. Именно наличие указанных операций согласно теории языков программирования позволит говорить о наличии абстракции вектор в языке.

Операции вставки и удаления элементов с указанием места обрабатываемого элемента для векторов предусматривать не будем, поскольку при необходимости сопоставлять по номеру элементы множеств, мультимножеств, кортежей и их веса в векторе целесообразно использовать взвешенное множество (мультимножество, кортеж).

Элементы матриц, также как и элементы векторов могут хранить номера элементов или веса, а также то и другое вместе. Над ними определены два отношения порядка элементов в строке R_i и в столбце R_j . Поэтому для них также должны быть определены операции обращения по соответствующим номерам, поиска элемента с заданными свойствами (максимально, минимального и с конкретным значением), а также поиска весов указанных элементов.

К элементарным операциям модификации для множеств, мультимножеств и кортежей из определенных в теории множеств относятся операции конкатенации и дополнения для случая, когда второе множество состоит из одного элемента, которые осуществляют операции добавления элемента (в конец – для кортежа) и удаления элемента по значению.

В эту же группу включим операции записи элемента и веса элемента по номеру. Для множества и мультимножества этих операций достаточно. Для кортежей, элементы которых упорядочены, в качестве элементарных также необходимы операции вставки элемента в указанное место и удаления элемента со сдвигом оставшихся на его место, что обеспечит необходимый порядок элементов при модификации этой структуры.

Вектор и матрицу в отличие от кортежа будем считать абстракциями статическими, структура которых в процессе обработки меняться не может, что соответствует обычному использованию указанных абстракций. Следовательно, для них определены только операции записи элемента и веса по номеру.

Множество операций модификации для абстракций представлено в таблице 1.

Таблица 1 - Операции модификации, определенные для базовых абстракций

Операции	Множество	Мульти- множество	Кортеж	Вектор	Матрица
Конкатенация элемента	+	+	+	–	–
Дополнение множества до элемента	+	+	+	–	–
Вставка элемента по номеру	–	–	+	–	–
Удаление элемента по номеру	–	–	+	–	–
Запись элемента по номеру	+	+	+	+	+
Запись веса по номеру	+	+	+	+	+

К элементарным операциям просмотра соответственно относятся операции:

- проверки вхождения элемента во множество (базовое множество – для мультимножеств и кортежей);
- определения множества элементов, совпадающих с заданным – для мультимножества и кортежа;
- определение следующего и предыдущего элементов – в кортеже (см. таблицу 2).

Таблица 2 - Операции просмотра, определенные для множественных абстракций

Операции	Множество	Мультимножество	Кортеж
Проверка вхождения элемента	+	+	+
Определение множества элементов, совпадающих с заданным	–	+	+
Определение следующего и предыдущего элементов	–	–	+

К операциям определения элементарных характеристик отнесем операции:

- определения мощности множества (базового множества для мультимножества и кортежа), длины кортежа (мультимножества);
- количества вхождения элемента в мультимножество;
- определения номера элемента по его значению, значения по номеру и веса по номеру – для всех абстракций (см. таблицу 3)

Таблица 3 - Операции определения элементарных характеристик абстракций уровня множеств

Операции	Множе- ство	Мульти- множество	Кортеж	Вектор	Матри- ца
Определение мощности множества (базового множества)	+	+	+	—	—
Определение количества элементов в структуре (длины)	—	+	+	—	—
Определение количества вхождения элемента в мультимножество или кортеж	—	+	+	—	—
Поиск номера(-ов) элемента по значению	+	+	+	+	+
Определение значения элемента по номеру	+	+	+	+	+
Определение веса элемента по номеру	+	+	+	+	+

* — для базового множества;

** — ближайшего по порядку просмотра;

*** — в неявной последовательности элементов;

**** — для обращения к элементу необходимы два номера.

Таким образом, совокупность элементарных операций над множествами, мультимножествами и кортежами включает 15 операций.

Заключение

На основании выполненных исследований можно утверждать, что целесообразно разработать специализированный язык описания алгоритмов решения задач структурного анализа и синтеза. Язык должен оперировать абстракциями: «множество с заданным универсумом», «мультимножество», «кортеж», «вектор», «матрица». Над этими абстракциями в языке должны быть реализованы как традиционные операции, такие как пересечение множеств или проверка вхождения элемента во множество или мультимножество, так и специальные операции, связанные с использованием перечисленных абстракций в алгоритмах, например, обращение к элементу кортежа по номеру.

Применение языка с указанным выше набором абстракций и операций над ними приведет к сокращению времени разработки и реализации алгоритма и позволит

реализовать автоматический анализ емкостной и вычислительной сложности по тексту программы на предлагаемом языке и автоматизировать применение оптимизирующих преобразований, позволяющих существенно снизить емкостную и/или вычислительную сложность разрабатываемых программ решения задач структурного анализа и синтеза.

Список литературы

1. Скороходов В.А., Чеботарева А.С. Максимальный поток в сети с циклической зависимостью длительностей прохождения по дугам от времени // Известия высших учебных заведений. Северо-Кавказский регион. Естественные науки. 2011. № 5. С. 23-27.
2. Евстигнеев В.А., Турсунбай К.Ы. [О раскраске графов в классе параллельных локальных алгоритмов](#) // [Сибирский журнал вычислительной математики](#). 2011. Т. 14, № 3. С. 231-243.
3. Алгоритмы и методы решения задач составления расписаний и других экстремальных задач на графах больших размерностей / Е.В. Панкратьев, А.М. Чеповский, Е.А. Черепанов, С.В. Чернышев // [Фундаментальная и прикладная математика](#). 2003. Т. 9, № 1. С. 235-251.
4. Лузгачёв М.В., Самуйлов К.Е. Задача маршрутизации трафика на графе сети [MPLS](#) с одноадресными соединениями // [Вестник Российского университета дружбы народов. Сер. Математика, информатика, физика](#). 2009. № 1. С. 23-33.
5. Ейбоженко Д.А. Алгоритм К-кластерной оптимизации для задачи Штейнера на ориентированных графах // [Вестник Санкт-Петербургского университета. Сер. 10. Прикладная математика. Информатика. Процессы управления](#). 2011. № 2. С. 29-39.
6. Соколов Г.В. Анализ алгоритмов автоматической укладки графов на плоскости в рамках задачи визуализации моделей на графах // [Вестник Пермского национального исследовательского политехнического университета. Прикладная математика и механика](#). 2010. № 15. С. 162-171.
7. Титов Ю.П. Применение декомпозиции графа на связные составляющие при работе алгоритмов выбора маршрутов транспортных перевозок // [Сборник научных трудов Sworld](#). 2012. Вып. 2, т. 1. С. 8-12.
8. Бёрд Р. Жемчужины проектирования алгоритмов: функциональный подход. М.: ДМК Пресс, 2013. 330 с. [Bird R. Perls of Function Algorithm Design. Cambridge University Press, 2010].
9. Mehlhorn K., Naher St. The LEDA Platform of Combinatorial and Geometric Computing. Cambridge University Press, 1999.

10. Himsolt M. GML: A Portable Graph File Format. Technical Report. Universitat Passau, 1997.
11. Овчинников В.А. Алгоритмизация комбинаторно-оптимизационных задач при проектировании ЭВМ и систем. М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. 286 с.
12. Кормен Т., Лейзерсон Ч., Риверст Р. Алгоритмы: построение и анализ: пер. с англ. М.: Издательский дом Вильямс, 2006. 955 с. [Cormen T.H., Leiserson C.E., Rivest R.L. Introduction to Algorithms. The MIT Press, 1999.]
13. Ахо А.В., Хопкрофт Д., Ульман Д.Д. Структуры данных и алгоритмы: пер. с англ. М.: Издательский дом Вильямс, 2003. 382 с. [Aho A.V., Hopcroft J.E., Ullman J.D. Data structures and Algorithms. Addison-Wesley Publishing Company, 2000.]
14. Пасечников К.А., Иванова Г.С. Синтез оптимальных структур данных для решения задач на графах // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2008. № 4 (73). С. 29-38.
15. Овчинников В.А., Иванова Г.С., Павлов А.Е. Оценка эффективности оптимизирующих преобразований алгоритмов операций над ультраграфами // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2013. № 1. DOI: [10.7463/0113.0547731](https://doi.org/10.7463/0113.0547731)
16. Иванова Г.С. Модели объектов задач структурного синтеза. // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2006. № 12. Режим доступа: <http://technomag.edu.ru/doc/62361.html> (дата обращения 15.09.2013).

Abstractions and base operations of a specialized description language of algorithms for solving problems of structural analysis and synthesis

12, December 2013

DOI: 10.7463/1213.0656686

Ivanova G., S.

Bauman Moscow State Technical University, 105005, Moscow, Russian Federation
gsivanova@bmstu.ru

This article justifies the need for a specialized programming language that would allow to describe solving procedures for tasks of structural analysis and synthesis in terms of mathematical models being used. Compared with the existing languages and subroutine libraries such language would not only allow to reduce program development time significantly but also provide an opportunity to develop more efficient programs in terms of computational complexity of algorithms and runtime. According to the analysis of algorithms and representations of graph models a set of the necessary abstractions and operations on them was determined. The abstractions such as "set", "multi-set", "sequence", "vector" and "matrix" were proposed for basic abstractions. Operations on these abstractions were divided into mathematical and related to specific implementation features of the algorithm descriptions. A set of elementary operations of the indicated types was defined for each abstraction.

Publications with keywords: [structure synthesis](#), [specialized language](#), [language abstractions](#), [operations with abstractions](#)

Publications with words: [structure synthesis](#), [specialized language](#), [language abstractions](#), [operations with abstractions](#)

References

1. Skorokhodov V.A., Chebotareva A.S. Maksimal'nyy potok v seti s tsiklicheskoj zavisimost'yu dlitel'nostey prokhozhdeniya po dugam ot vremeni [Maximum flow in a network with a cyclic duration dependence passing through the arcs of the time]. *Izvestiya vuzov. Severo-kavkazskii region. Estestvennye nauki* [Izvestiya vuzov. Severo-kavkazskii region. Natural sciences], 2011, no. 5, pp. 23-27.
2. Evstigneev V.A., Tursunbay K.Y. O raskraske grafov v klasse parallel'nykh lokal'nykh algoritmov [On graph coloring in a class of parallel local algorithms]. *Sibirskiy zhurnal vychislitel'noy matematiki* [Siberian Journal of Numerical Mathematics], 2011, vol. 14, no. 3, pp. 231-243.

3. Pankrat'ev E.V., Chepovskiy A.M., Cherepanov E.A., Chernyshev S.V. Algoritmy i metody resheniya zadach sostavleniya raspisaniy i drugih ekstremal'nykh zadach na grafakh bol'shikh razmernostey [Algorithms and methods for solving scheduling problems and other extremum problems on large-scale graphs]. *Fundamental'naya i prikladnaya matematika* [Fundamental and Applied Mathematics], 2003, vol. 9, no. 1, pp. 235-251.
4. Luzgachev M.V., Samuylov K.E. Zadacha marshrutizatsii trafika na grafe seti MPLS s odnoadresnymi soedineniyami [Traffic Routing Problem on MPLS Network with Unicast Connections]. *Vestnik RUDN. Ser. Matematika, informatika, fizika*, 2009, no. 1, pp. 23-33.
5. Eybozhenko D.A. Algoritm K-klasternoy optimizatsii dlya zadachi Shteynera na orientirovannykh grafakh [*k*-Clustering optimization algorithm for Steiner problem in directed graphs]. *Vestnik Sankt-Peterburgskogo universiteta. Ser. 10. Prikladnaya matematika. Informatika. Protsessy upravleniya*, 2011, no. 2, pp. 29-39.
6. Sokolov G.V. Analiz algoritmov avtomaticheskoy ukladki grafov na ploskosti v ramkakh zadachi vizualizatsii modeley na grafakh [Analysis of Algorithms for automatic stacking graphs on a plane within the models on graphs visualization problem]. *Vestnik Permskogo natsional'nogo issledovatel'skogo politekhnicheskogo universiteta. Prikladnaya matematika i mekhanika*, 2010, no. 15, pp. 162-171.
7. Titov Yu.P. Primenenie dekompozitsii grafa na svyaznye sostavlyayushchie pri rabote algoritmov vybora marshrutov transportnykh perevozok [The Application of graph decomposition into connected components when transport routing algorithm working]. *Sbornik nauchnykh trudov Sworld*, 2012, iss. 2, vol. 1, pp. 8-12.
8. Bird R. *Perls of Function Algorithm Design*. Cambridge University Press, 2010. (Russ. ed.: Bird R. *Zhemchuzhiny proektirovaniya algoritmov: funktsional'nyy podkhod*. Moscow, DMK Press, 2013. 330 p.).
9. Mehlhorn K., Naher St. *The LEDA Platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.
10. Himsolt M. *GML: A Portable Graph File Format. Technical Report*. Universitat Passau, 1997.
11. Ovchinnikov V.A. *Algoritmizatsiya kombinatorno-optimizatsionnykh zadach pri proektirovanii EVM i system* [The algorithmization of combinatorial optimization problems in the design of computers and computer systems]. Moscow, Bauman MSTU Publ., 2001. 286 p.
12. Cormen T.H., Leiserson C.E., Rivest R.L. *Introduction to Algorithms*. The MIT Press, 1999. (Russ. ed.: Cormen T.H., Leiserson C.E., Rivest R.L. *Algoritmy: postroyeniye i analiz*. Moscow, Vil'yams, 2006. 955 p.).
13. Aho A.V., Hopcroft J.E., Ullman J.D. *Data Structures and Algorithms*. Addison-Wesley Publishing Company, 2000. (Russ. ed.: Aho A.V., Hopcroft J.E., Ullman J.D. *Struktury dannykh i algoritmy*. Moscow, Vil'yams, 2003. 382 p.).
14. Pasechnikov K.A., Ivanova G.S. Sintez optimal'nykh struktur dannykh dlya resheniya zadach na grafakh [Synthesis of optimal data structures for solving problems on graphs]. *Vestnik MGTU im. N.E. Baumana. Ser. Priborostroeniye* [Bulletin of the Bauman MSTU. Ser. Instrument Engineering], 2008, no. 4 (73), pp. 29-38.

15. Ovchinnikov V.A., Ivanova G.S., Pavlov A.E. Otsenka effektivnosti optimiziruyushchikh preobrazovaniy algoritmov operatsiy nad ul'tragrafami [Evaluation of efficiency of optimizing transformation for algorithms of ultra-graph operations]. *Nauka i obrazovanie MGTU im. N.E. Baumana* [Science and Education of the Bauman MSTU], 2013, no. 1. DOI: : [10.7463/0113.0547731](https://doi.org/10.7463/0113.0547731)

16. Ivanova G.S. Modeli ob"ektov zadach strukturnogo sinteza [The models of objects for structural creation tasks]. *Nauka i obrazovanie MGTU im. N.E. Baumana* [Science and Education of the Bauman MSTU], 2006, no. 12. Available at: <http://technomag.edu.ru/doc/62361.html> , accessed 15.09.2013.