Российская академия наук
Сибирское отделение
Институт систем информатики
имени А. П. Ершова

На правах рукописи

Стасенко Александр Павлович

МОДЕЛИ И РЕАЛИЗАЦИЯ ТРАНСЛИРУЮЩИХ КОМПОНЕНТОВ СИСТЕМЫ ФУНКЦИОНАЛЬНОГО ПРОГРАММИРОВАНИЯ

05.13.11 – математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

Новосибирск 2009

Работа выполнена в Институте систем информатики
имени А. П. Ершова СО РАН

Научный руководитель: Касьянов Виктор Николаевич,
доктор физико-математических наук, профессор

Официальные оппоненты: Малышкин Виктор Эммануилович,
доктор технических наук, профессор

Скопин Игорь Николаевич,
кандидат физико-математических наук

Ведущая организация: Сибирский федеральный университет

Защита состоится «5» июня 2009 г. в 16 ч. 00 мин. на заседании диссертационного совета ДМ 003.032.01 в Институте систем информатики имени А. П. Ершова Сибирского отделения РАН по адресу:

630090, г. Новосибирск, пр. ак. Лаврентьева, 6.

С диссертацией можно ознакомиться в читальном зале библиотеки ИСИ СО РАН (пр. ак. Лаврентьева, 6)

Автореферат разослан «4» мая 2009 г.

Ученый секретарь
диссертационного совета,
к. ф.-м. н. Мурзин Ф. А.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. В настоящий момент намечается существенное замедление роста производительности вычислительных систем за счет увеличения тактовой частоты вычислительных устройств, связанное с ограниченностью современных технологических возможностей. Поэтому для сохранения существующих темпов роста скорости вычислений, косвенно декларируемых законом «Moore's Law», возобновляется интерес к параллельным вычислениям. Ярким тому подтверждением является появление процессоров с двумя независимыми вычислительными ядрами, ориентированных на массовый рынок. Прослеживается общая тенденция увеличения значимости роли многих процессорных ядер, что в обозримом будущем может привести к значительному росту возможностей вычислительных систем по параллельной обработке данных.

К сожалению, подобного прогресса в области разработки программного обеспечения не наблюдается. Данное обстоятельство, прежде всего, связано с преобладанием императивных языков программирования, жестко фиксирующих порядок вычислений. Автоматическое выделение параллелизма в последовательной программе на этапе компиляции в общем случае обычно малоэффективно, а в распараллеливании на уровне машинных инструкций возможности современных процессоров практически исчерпаны. Возможности императивных языков по явному описанию фиксированного числа параллельных вычислений также не решают вопрос масштабирования на большее число процессоров (или их ядер). К тому же такой подход обычно снижает надежность программы и может даже вызывать её замедление при наличии единственного процессора (ядра) или большого числа пересекающихся критических секций.

Решение описанных проблем видится в применении функциональных языков программирования, задающих непосредственно сами вычисления в виде применений функций в их аргументам, а не то, как эти вычисления

должны быть исполнены. Такое неявное описание позволяет автоматизировать планирование порядка вычислений, избавляя программиста от решения этой задачи на этапе подготовки программы к исполнению, причём, с учётом архитектуры конкретной вычислительной системы. Таким образом, текст функциональной программы и её информационный граф задают машинно-независимый параллелизм.

Однако и в функциональных языках существуют ограничивающие параллелизм факторы. Одной из таких причин является набор языковых операций, специально ориентированный на последовательные вычисления или ограниченный параллелизм. В частности, в функциональном языке Лисп (Lisp) операторы для работы с данными обеспечивают выборку только одного компонента, а функциональный язык FP не предусматривает распараллеливание на уровне независимых операторов. Также в модели вычислений, присущей функциональному языку, могут присутствовать элементы управления, тем или иным образом приводящие к разделению процессами общих вычислительных ресурсов. Например, в схемах потока данных Дениса таким ресурсом является узел слияния потоков данных.

Указанные недостатки обычно обходятся в потоковых языках программирования, явно описывающих вычисления в виде графа потока данных и обычно являющихся также и функциональными языками. Эти языки, как правило, содержат операторы размножения информационных потоков, их группирования в списки данных различного уровня вложенности и одновременного выделения из списков нескольких независимых и разнородных по составу групп. Потоковые языки программирования изначально ориентировались на потоковые архитектуры вычислительных систем, имевших распространение в 80-90-х гг. XX века, и с тех пор практически не развивались.

Тем самым, актуальность работы обуславливается:

1) технологическими трудностями роста производительности последовательных машинных архитектур и, как следствие, ростом интереса к параллельным вычислениям;
2) высокой изменчивостью параллельных машинных архитектур и, как следствие, необходимостью в машинно-независимом представлении параллелизма;
3) естественной пригодностью потоковых языков программирования для описания машинно-независимого параллелизма;
4) малой распространенностью и устарелостью существующих потоковых языков и систем программирования на их основе.

Язык программирования Sisal является одним из самых известных потоковых языков промышленного уровня и позиционируется как замена языка Fortran для научных применений. Язык Sisal имеет следующие особенности, облегчающие переход с популярных императивных языков программирования:

- приближенный к языку Паскаль синтаксис;
- развитая система типов;
- явно выделенные циклические выражения.

Последняя спецификация языка Sisal версии 2.0 датируется 1991 г., а последнее обновление транслятора OSC, работающего только с языком Sisal версии 1.2 от 1985 г., было в 1995 г. В 1995 г. также появилось пользовательское описание языка Sisal 90, не содержащее, однако, точных спецификаций языка.

В связи с этим актуальна задача разработки и спецификации новой версии языка Sisal, включающей некоторые возможности современных языков программирования. Не меньшей актуальностью обладает создание компилятора для новой версии языка Sisal, что не является тривиальной задачей ввиду богатого синтаксиса и семантики языка Sisal.

Цель работы – разработка и реализация моделей транслирующих компонентов системы функционального программирования. Достижение цели работы связывается с решением следующих **задач**.

1. Разработка языка Sisal версии 3.1, развивающего базовую, функциональную часть языка Sisal 3.0, являющегося входным языком системы SFP и основанную на языке Sisal 90.
   1.1. Уточнение описания языка Sisal 90.
   1.2. Расширение языка Sisal 90 средствами модульного программирования (Sisal 3.0) и пользовательскими типами.
   1.3. Улучшение синтаксиса и семантики языка Sisal 90.
2. Разработка машинно-независимого ВП IR1 программ языка Sisal 3.1, основанного на графах информационных зависимостей.
   2.1. Разработка модели внутреннего представления (ВП) IR1 для машинно-независимого описания функциональных программ и её специализация для языка Sisal 3.1.
   2.2. Разработка и реализация вспомогательных компонентов ВП IR1 для его сохранения, восстановления и визуализации.
3. Исследование методов трансляции и создание компилятора переднего плана из текста программ языка Sisal 3.1 в ВП IR1.
   3.1. Разработка модели построения компоненты компилятора переднего плана, основанной на теории автоматов, и её графического представления.
   3.2. Разработка компоненты поддержки трансляции из текстового представления во ВП IR1 (для поддержки нескольких входных языков системы SFP).

Методы исследования. В диссертационной работе использовались понятия и методы теории графов, теории алгоритмов и элементы теории множеств. Также применялась теория синтаксического анализа и построения трансляторов. В качестве сравнения применялись различные классы

потоковых схем. Для описания синтаксиса языка программирования использовались расширенные формы Бэкуса-Наура (БНФ).

Научная новизна. В диссертационной работе получены новые научные результаты.

1. Спроектирована и формально описана новая версия языка Sisal 3.1, развивающая функциональную часть предыдущих версий языка Sisal и приближающая её к уровню современных языков программирования.
2. Разработаны модели для удобного построения, хранения, анализа, преобразования и использования машинно-независимого внутреннего представления программ функциональных языков программирования.
3. Создана модель построения компоненты компилятора переднего плана, разделяющая его реализацию на графическую, текстовую и интерпретирующую части и позволяющая использовать преимущества каждой из них для упрощения построения и сопровождения компиляторов переднего плана, написанных вручную.
4. Формализована модель описания графической составляющей компилятора переднего плана, основанная на упрощении классической модели магазинного автомата с расширениями, делающими возможным её практическое применение.

Практическая ценность. Разработанные язык программирования Sisal 3.1, его ВП IR1 и транслятор из первого во второе являются неотъемлемой частью системы функционального программирования (SFP), разрабатываемой в рамках проекта ПРОГРЕСС. Данные разработки будут использоваться другими участниками проекта SFP в качестве основы для создания своих частей проекта и будут внедрены в учебный процесс в Новосибирском госуниверситете.

Апробация работы. Основные положения диссертации обсуждались на следующих конференциях и семинарах.

1. 9th International Conference on Parallel Computing Technologies, Pereslavl-Zalessky, 2007 г.
2. Европейская конференция по вычислениям (ECC-2007), г. Афины, 25-27 сентября, 2007.
3. V Всероссийская конференция молодых ученых по математическому моделированию и информационным технологиям (с участием иностранных ученых), Новосибирск, ИВТ СО РАН, 2004 г.
4. Конференция-конкурс «Технологии Microsoft в информатике и программировании», Новосибирск, 2005 г.
5. Конференция-конкурс «Технологии Microsoft в теории и практике программирования», Новосибирск, 2006 г.
6. XLIV Международная научная студенческая конференция «Студент и научно-технический прогресс», Новосибирск, 2006 г.
7. XII Международная научно-методическая конференции «Новые информационные технологии в университетском образовании», 2007 г.
8. Семинар спецкурса «Введение в функциональное программирование», 2007 г.
9. Семинары «Конструирование и оптимизация программ», Новосибирск, ИСИ СО РАН, 2002-2008 гг.

Публикации. Основные результаты диссертационной работы опубликованы в 19 печатных работах, из которых 2 препринта, 12 статей и 6 тезисов докладов. Исследования поддерживались грантами УР.04.01.027 и УР.04.01.0201 научной программы «Университеты России» Министерства науки и образования Российской Федерации. Исследования выполнялись в соответствии с планами научно-исследовательских работ ИСИ СО РАН по проекту 3.15 «Методы и средства трансляции и конструирования программ». Проект проводился в рамках программы 3.1 СО РАН «Информационное и математическое моделирование в различных областях знаний, задачи

поддержки принятия решений, экспертные системы, системное и теоретическое программирование».

Структура диссертации. Диссертация состоит из введения, четырёх глав, заключения, списка литературы и девяти приложений. Работа содержит 109 страниц машинописного текста (без приложений и библиографии), 78 рисунков и 22 таблиц. Список литературы содержит 99 наименований.

СОДЕРЖАНИЕ РАБОТЫ

Во введении обосновывается актуальность темы диссертации, формулируются её цели, характеризуется научная новизна и практическая ценность работы.

В главе 1 содержится краткий обзор потоковых языков программирования, историю развития языка Sisal и описание нововведений и особенностей его последней версии Sisal 3.1, представляемой в данной работе.

Раздел 1.1 рассматривает общие определяющие качества потоковых языков программирования и их преимущества над императивными языками программирования.

Раздел 1.2 содержит историю и краткий анализ развития версий языка Sisal, начиная с языка Val, являющегося прародителем языка Sisal, и заканчивая версией Sisal 3.1. Приводятся примеры основных нововведений каждой версии языка. Нововведения языка Sisal 3.0 заключаются в возможности задавать отдельные части программы на императивном языке Си, расширенной поддержке модульности программ, возможности их предварительной обработки (preprocessing) и аннотирования для упрощения оптимизирующих преобразований.

Раздел 1.3 представляет новую версию языка Sisal 3.1, являющуюся развитием языка Sisal 90 с помощью идей улучшенной поддержки модульности и механизмами предварительной обработки, предложенными в языке Sisal 3.0. Описание мультиязыкового программирования и

аннотирования программ оставлены для внедрения в последующих версиях языка. В разделе 1.3 также описываются нововведения языка Sisal 3.1, среди которых выделяются средства для задания пользовательских типов. Введение пользовательские типов в язык Sisal 3.1 потребовало реализации пользовательских операций и статического полиморфизма.

Пользовательский тип определяется через указание его базового типа. Детали реализации базового типа скрываются запретом на неявное наследование пользовательским типом операций его базового типа, что требует явного определения всех свойственных новому типу операций и способствует ясному разграничению операций определяемых иерархией типов.

Отделение внутреннего представления данных от внешнего возможно через переопределение операции неявного преобразования типов. Переопределение операции точки также позволяет задавать «мнимые» поля данных пользовательского типа. Ввиду функциональной природы языка Sisal, возможен естественный синтаксис вызова метода пользовательского типа, путём переопределения его операции точки, возвращающей значение функционального типа.

Введение пользовательских операций, потребовало введение их статического полиморфизма, который было решено распространить на вызовы функций и редукций. В связи с потоковой природой языка Sisal 3.1, имеющей ограничения на набор базовых операций потоковых машинных архитектур, было решено разрешать только статический полиморфизм, обрабатываемый во время трансляции.

Раздел 1.4 посвящен обоснованию ограничений языка Sisal 3.1 по сравнению с языком Sisal 90, таких как отказ от глобальных констант и ограничение полиморфизма множества типов.

Раздел 1.5 содержит анализ изменений синтаксиса и семантики языка Sisal 90 в языке Sisal 3.1, направленных на улучшение межъязыкового взаимодействия с языком Си, повышение читаемости программ, улучшение

синтаксиса, упрощение синтаксического разбора и устранение его неоднозначностей.

В главе 2 описывается модель внутреннего представления (МВП) IR1 (Internal Representation 1). Раздел 2.1 состоит из требований к разрабатываемой МВП IR1:

1) машинная независимость представления параллелизма, без явного выделения нескольких потоков вычислений;
2) машинная независимость значений простых типов данных, независимость от разрядности машинной архитектуры;
3) полнота, позволяющая произвести ретрансляцию в программу на исходном языке, семантически эквивалентную исходной программе;
4) ретрансляция в синтаксически корректную программу после преобразований, корректных для исходного языка (приложение 6);
5) интерпретируемость без дополнительных преобразований;
6) простота преобразований, в том числе и оптимизирующих;
7) иерархичность сущностей ВП, задающих естественную вложенность конструкций функционального языка программирования;
8) представление всех неявных операций явным образом;
9) представление циклических конструкций в явном виде;
10) расширяемость – лёгкое введение новых сущностей ВП, задающих новые конструкции языков программирования и типы данных;
11) переносимость – применение сущностей в разных языковых средах.

В разделе 2.2 находится краткий обзор распространенных промежуточных представлений программ. В частности, рассматриваются известные схемы потока данных.

Раздел 2.3 содержит описание языка промежуточного представления программ IF1 (Intermediate Form 1), основывающегося на ациклических иерархических орграфах и являющегося основой разработанной МВП IR1. Приводятся способы представления последовательного исполнения, альтернативы и итерации в терминах языка IF1.

Программа на языке IF1 состоит из множества графов специального вида, среди которых выделено подмножество графов, соответствующих функциям исходной Sisal-программы. Граф в IF1 состоит из объектов трёх видов: вершин (node в английской терминологии IF1), дуг и рамки графа. Причём вершинам и рамке графа приписаны упорядоченные множества входов не более одной дуги и выходов одной или более дуг. Входы рамки графа рассматриваются как выходы (результаты вычислений) графа, а выходы рамки графа как входы (параметры) графа. Далее под термином *порт*, как и в английской терминологии IF1, подразумевается вход и выход.

Каждой дуге графа приписан тип пересылаемого ею значения, если IF1 задаёт строго типизированный язык. Существует не исходящая ни откуда дуга специального вида, обозначающая литерал некоторого типа и передающая его значение в какой-либо вход.

Вершины обозначают операции над своими входами (аргументами), результаты которых находятся на выходах вершины. Вершины бывают простыми и составными. Простые вершины (или просто вершины) не содержат дополнительной информации помимо ассоциированной с ними операции. Составные вершины дополнительно содержат упорядоченное множество графов. Их число и все связи между входами (выходами) составной вершины и входами (выходами) этих графов неявно задаются типом составной вершины. Структурированность вычислений, задаваемых IF1-графом, основывается на иерархичности IF1-графов, заданной посредством графов составной вершины. Показывается, что IF1-граф является простым иерархическим графом.

Раздел 2.4 состоит из описания МВП IR1, как набора сущностей, реализующих понятия основные понятия языка IF1. Особенности представления программ языка Sisal 3.1, заданные набором числовых и строковых свойств МВП IR1, вынесены в приложения 4, 5 и 6. Для МВП IR1 в этом разделе описываются разработанные алгоритмы удаления неиспользуемых строк и типов, слияния эквивалентных типов.

Во-первых, было принято решение отказаться от выделения дуги в качестве отдельной сущности (интерфейса) и переложить связи между портами на его интерфейс, поддерживающий типы передаваемых дугами значений. Такое решение позволило упростить реализацию обхода графа, так как для любого взятого порта можно сразу определить порт, поставляющий ему значения, и множество портов, принимающих, в свою очередь, от него значения. Это, например, позволило легко реализовать операцию удаления порта из ВП IR1. Далее термин дуга, однако, будет употребляться при описании взаимосвязей между портами.

Во-вторых, было принято решение не делать различных интерфейсов для реализации входов и выходов. Это позволило рассматривать единый интерфейс порта с этих двух сторон одновременно, что, в свою очередь, унифицировало сущности графа и рамки графа с точки зрения того, какую группу портов считать входными, а какую – выходными. Теперь один и тот же порт может служить выходом графа и в то же время являться входом для дуг, принадлежащих этому графу (и наоборот). Такая унификация сделала возможным рассматривать граф в качестве вершины в «объемлющем» графе, ничего не знающем о внутреннем устройстве такой вершины.

Раздел 2.5 содержит описание моделей описания вспомогательных компонент МВП IR1: компоненты преобразования в эквивалентную XML-форму и компоненты визуализации и навигации. Даются краткие сведения, связанные с используемыми при реализации этих интерфейсов алгоритмами.

В главе 3 поднимаются вопросы, связанные с графическим метаязыком для описания транслятора. Создание транслятора требует решения довольно трудоёмкой задачи построения распознавателя транслируемого языка по его формальному описанию. В разделе 3.1 рассматриваются существующие методы построения нисходящих и восходящих распознавателей. Обосновываются преимущества описания языка в виде непосредственно интерпретируемого, наглядного автомата.

В разделе 3.2 вводится модель упрощенного магазинного автомата Ψ, являющаяся классической моделью магазинного автомата с ограничениями на вид функции перехода, повышающими наглядность её графического представления. Доказывается, что подобное упрощение в случае недетерминированных автоматов не влияет на возможность задавать произвольные контекстно-свободные языки. Для детерминированных автоматов доказывается, что класс, задаваемых ими языков, равен классу LL_1 языков. Показывается, что в рамках модели детерминированного автомата Ψ, не сужая класс представимых им языков, можно устранить все ε-переходы первого и второго вида. Также вводится новый класс грамматик, в котором для каждого детерминированного автомата Ψ существует грамматика, задающая тот же язык, и наоборот.

Для визуального представления модели детерминированного автомата Ψ в разделе 3.3 описывается графический метаязык, программы на котором далее называются схемами Ψ. Схемы Ψ отображают модель автомата Ψ в виде ориентированного, размеченного графа, вершины которого, за исключением специального случая pop-вершин, соответствуют состояниям автомата, а связи между вершинами – переходам.

Раздел 3.4 содержит описание расширений графического метаязыка. Рассмотренная схема Ψ допускает эффективную реализацию, но ограничена узким классом LL_1 языков. Для исправления этой ситуации схемы Ψ были усилены за счёт добавления семантически зависимых переходов. В то же время неудобно и ненаглядно строить большие полностью определённые схемы Ψ, где реакция на ошибки разбора легко унифицируема. Для устранения этих затруднений в схемы Ψ был внедрён механизм иерархической обработки ошибок разбора, сходный с обработкой исключений языка Си++.

В разделе 3.5 приводится схема механизм применения схем Ψ для описания работы транслятора, показывающая разделение разрабатываемого транслятора на автоматную и генерирующую части. Для функционирования

транслятора состояниям и переходам автомата приписывается дополнительная разметка, обозначающая идентификаторы процедур. Процедуры сопоставляются своим идентификаторам во время исполнения автомата.

В конце главы, в разделе 3.6 описывается преобразование графического представления из графового формата .GraphML, удобного для построения и навигации, во внутренний формат (приведенный в приложении 9), приспособленный для непосредственного исполнения. Для этого была написана специальная утилита, дополнительно осуществляющая с помощью описанных в разделе алгоритмов устранение мнимых дуг, удаление недостижимых состояний, оптимизацию тестовых условий на дугах, минимизацию генерируемого автомата и разворачивание нерекурсивных зависимостей.

Представление автомата в виде данных позволяет проводить динамическую оптимизацию его дальнейшего исполнения, описанную в разделе 3.7. В частности, возможно вычисление вероятностей срабатывания переходов из взятого состояния и последующее переупорядочение последовательности их проверки.

В главе 4 находится описание транслятора программ на языке Sisal 3.1 во ВП IR1. Глава начинается с обзора в разделе 4.1 существующих трансляторов языка Sisal. Рассматривается «официальный» транслятор OSC и его развитие в рамках проекта «Sisal Lives». Также приводятся различные расширения OSC такие, как транслятор D-OSC, ориентированный на машины с распределённой памятью, и транслятор FSC, комбинирующий поддержку машин с общей и распределенной памятью.

В разделе 4.2 находится общая схема транслятора языка Sisal 3.1, написанного с использованием схем Ψ лексического и синтаксического анализа, совмещенного с семантическим разбором.

Раздел 4.3 содержит описание требований к модели транслятора из текстового представления во ВП IR1. Основное содержание раздела

составляет описание системы интерфейсов, составляющих основу разработанной модели.

Раздел 4.4 содержит общую информацию о подсистеме лексического анализа транслятора Sisal 3.1. Приводится описание лексем, задач лексического разбора, возможных лексических ошибок и предупреждений.

Раздел 4.5 занят общим описанием синтаксического и семантического анализа языка Sisal 3.1. Указываются причины объединения анализов синтаксиса и семантики в разрабатываемом трансляторе. Приводится общая структура разбора, задаваемого схемой Ψ, и общее описание разбора, включая используемые стеки. Рассматриваются особенности разбора отдельных конструкций языка. Перечисляются сообщения об ошибках и предупреждениях транслятора. Освещается механизм восстановления после ошибок синтаксического и семантического разбора.

Приложение 1 содержит семантику языка Sisal 3.1 в нестрогом виде пользовательского описания, которое, по замыслу, претендует на полноту и непротиворечивость.

Приложение 2 содержит полную грамматику языка Sisal 3.1 в расширенной форме Бэкуса-Наура (БНФ).

Приложение 3 состоит из текста интерфейса и реализации модуля языка Sisal 3.1, который описывает типы complex и double_complex, эквивалентные аналогичным типам языка Sisal 90.

Приложение 4 описывает синтаксис и семантику простых вершин МВП IR1, необходимых для представления программ языка Sisal 3.1.

Приложение 5 описывает синтаксис и семантику составных вершин МВП IR1, используемых для представления условных и циклических конструкций языка Sisal 3.1.

Приложение 6 содержит описание специализации МВП IR1 для языка Sisal 3.1.

Приложение 7 состоит из описания структура XML-представления ВП IR1 и примера XML-представления простой программы языка Sisal 3.1.

Приложение 8 содержит схемы Ψ, описывающие простую грамматику, практически весь лексический разбор языка Sisal 3.1 и показательную часть семантико-синтаксического разбора языка Sisal 3.1.

Приложение 9 посвящено спецификации формата файла, задающего интерпретируемый автомат анализатора, т. е. конечной форме преобразования программы графического метаязыка, описанного в главе 3.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ

В результате работы создан компилятор переднего плана новой версии языка Sisal 3.1, основывающийся на хорошо проработанных моделях интерфейса взаимодействия с пользователем и модели разделения реализации транслятора на графическую и текстовые части, связанные воедино частью интерпретирующей. Полученные научные и практические результаты работы далее сформулированы более точно.

1. Разработана и формально описана новая версия языка Sisal 3.1.

Новая версия языка Sisal 3.1 получена путём упрощения, улучшения, расширения и уточнения пользовательского описания языка Sisal 90, а также использования идей языка Sisal 3.0 и пользовательских типов. Формальное описание синтаксиса языка Sisal 3.1 является первым формальным описанием синтаксиса языка Sisal со времен языка Sisal 1.2. Семантика языка Sisal 3.1 приводится в нестрогом виде пользовательского описания, которое, по замыслу, претендует на полноту и непротиворечивость. Введение пользовательских типов в язык Sisal 3.1 повлекло введение средств переопределения операций и статического полиморфизма. Изменение языка Sisal 90 в языке Sisal 3.1 было обусловлено повышением удобства разбора и улучшением читаемости программ этого языка.

2. Разработаны модель внутреннего представления (МВП) IR1 для машинно-независимого описания функциональных программ и её специализация для языка Sisal 3.1.

Модель внутреннего представления (МВП) IR1 основывается на языке промежуточной формы IF1 и описывается системой взаимосвязанных интерфейсов и определяющей их взаимодействие реализацией COM-компонента. МВП IR1 задаёт общую основу специализации ВП IR1 для конкретного языка программирования, поэтому в терминах МВП IR1 выражаются лишь самые общие понятия языка IF1, не зависящие от конкретного языка. Для МВП IR1 разработаны и описаны алгоритмы удаления неиспользуемых строк и типов, слияния эквивалентных типов.

3. Разработаны и реализованы модели описания вспомогательных компонентов МВП IR1: компоненты преобразования в эквивалентную XML-форму и компоненты визуализации.

Модели вспомогательных компонентов ВП IR1 описываются системой интерфейсов, не зависят от специализации МВП IR1 и реализуются с помощью COM-компонента. Модель компонента XML-преобразования описывает эквивалентные преобразования между ВП IR1 и несколькими уровнями наглядности XML-текста. В модель заложена поддержка расширения и модификации ВП IR1 за счёт независимости XML-текста от порядка нумерации объектов ВП IR1. Модель компонента визуализации обеспечивает отображение и перемещение по ВП IR1 в терминах графического языка промежуточной формы IF1. Реализация модели основывается на тесно интегрированном исходном коде размещения графов свободно-распространяемой системы GraphViz и для визуализации использует интервальные деревья для эффективного отображения больших графов и проверки наличия их вершин и портов в данной точке.

4. Разработана и реализована модель описания компонента для поддержки трансляции из текстового представления во ВП IR1.

Модель компонента поддержки трансляции из текстового представления во ВП IR1 также описывается системой интерфейсов и реализуются с помощью COM-компонента. Модель описывает общий вид компилятора переднего плана в виде лексического и синтаксически-

семантического разборов. Моделью определяются понятия потока символа, потока лексем, средств сообщения о событиях трансляции и поддержки модульности. Реализация модели задаёт стандартные реализации её понятий, а также дополнительную функциональность, наподобие порождения потока символов файла в заданной кодировке, перечисление элементов которого вызывает дисковое чтение.

5. Разработана модель построения компонента компилятора переднего плана и её графическое представление.

Разработанная модель основывается на модели детерминированного магазинного автомата и упрощает общий вид функции перехода для более наглядного графического представления автомата. Доказывается, что такое упрощение не влияет на возможность недетерминированных автоматов задавать любые контекстно-свободные языки. Для детерминированных автоматов доказывается, что класс, задаваемых ими языков, равен классу LL_1 языков. Модель расширяется семантически-зависимыми переходами, реализующими «семантику отношений» (контекстные условия), и средствами иерархической обработки ошибок разбора. Разработаны и описаны алгоритмы преобразования наглядного описания автомата к его эффективно интерпретируемой форме представления.

6. Создан компонент компилятора переднего плана, позволяющий преобразовывать модуль программы на языке Sisal 3.1 во внутреннее представление IR1.

Разработанный в виде COM-компонента транслятор переднего плана является первой реализацией языка Sisal со времен языка Sisal 1.2 и успешно апробирует предложенные модели. К особенностям реализации транслятора относится его разделение на графическую, текстовую и интерпретирующую части. Графическая часть описывает автомат, распознающий синтаксис языка. Текстовая часть содержит раздельное описание «семантики отношений» языка и «операционной семантики», порождающей объекты

внутреннего представления IR1. Интерпретирующая часть транслятора связывает воедино его графическую и текстовую части.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. **Kasyanov V. N., Stasenko A. P., Gluhankov M. P., Dortman P. A., Pyjov K. A., Sinyakov A. I.** SFP – An interactive visual environment for supporting of functional programming and supercomputing // WSEAS Transactions on Computers. — Athens: WSEAS Press, 2006. — Vol. 5, N 9. — P. 2063–2070.
2. **Стасенко А. П.** Автоматная модель визуального описания синтаксического разбора // Вычислительные технологии. — Новосибирск: ИВТ СО РАН, 2008. — Т. 13, N. 5. — С. 70–87.
3. **Стасенко А. П., Пыжов К. А., Идрисов Р. И.** Компилятор в системе функционального программирования SFP // Вестник НГУ. Сер. Информационные технологии. — Новосибирск: Изд-во НГУ, 2008. — 11 с.
4. **Kasyanov V. N., Stasenko A. P.** Sisal 3.1 language structures decomposition // Parallel Computing Technologies. — Lecture Notes in Computer Science, 2007. — Vol. 4671/2007. — P. 62–73.
5. **Kasyanov V. N., Stasenko A. P.** A Functional Programming System SFP: Sisal 3.1 Language Structures Decomposition // Proceedings of 9th International Conference, PaCT 2007, Pereslavl-Zalessky, Russia, September 2007. — Berlin: Springer-Verlag, 2007. — P. 62¬–73.
6. **Kasyanov V. N., Stasenko A. P.** Sisal 3.1 language structures decomposition // European Computing Conference. Book of Abstracts. — Athens: WSEAS Press, 2007. — P. 92.
7. **Kasyanov V. N., Stasenko A. P.** Sisal 3.2 language structures decomposition // Lecture Notes in Electrical Engineering. — Berlin: Springer-Verlag, 2009. — Vol. 28. — P. 582–594.

8. **Stasenko A. P.** Sisal 3.1 language structures decomposition // Bull. Novosibirsk Comp. Center. Ser. Computer Science. — Novosibirsk, 2006. — Iss. 24. — 8 p.
9. **Глуханков М. П., Дортман П. А., Павлов А. А., Стасенко А. П.** Транслирующие компоненты системы функционального программирования SFP // Современные проблемы конструирования программ. — Новосибирск: Институт систем информатики имени А. П. Ершова СО РАН, 2002. — С. 69–87.
10. **Стасенко А. П.** Система интерфейсов транслятора во внутреннее представление IR1 // Методы и инструменты конструирования и оптимизации программ. — Новосибирск: Институт систем информатики имени А. П. Ершова СО РАН, 2005. — С. 229–238.
11. **Стасенко А. П.** Обзор потоковых языков программирования // Проблемы интеллектуализации и качества систем программирования. — Новосибирск: Институт систем информатики имени А. П. Ершова СО РАН, 2006. — 12 с.
12. **Стасенко А. П.** Автоматная модель визуального описания синтаксического разбора // Методы и инструменты конструирования программ. — Новосибирск: ИСИ СО РАН, 2007. — С. 186–209.
13. **Стасенко А. П.** Графический метаязык для описания транслятора // Сборник трудов аспирантов и молодых ученых «Молодая информатика». — Новосибирск: Институт систем информатики имени А. П. Ершова СО РАН, 2005. — С. 105–113.
14. **Стасенко А. П.** Графический метаязык для описания транслятора // Тез. докл. V Всеросс. конф. молодых ученых по математическому моделированию и информационным технологиям. — Новосибирск: ИВТ СО РАН, 2004. — С. 53.
15. **Стасенко А. П.** Совмещение достоинств Microsoft Dynamic HTML и W3C-совместимых HTML в системе HTML-справки // Конф.-конкурс

«Технологии Microsoft в информатике и программировании»: Тез. докл. — Новосибирск, 2005. — С. 45–47.

16. **Стасенко А. П.** Использование автоматного подхода для построения компилятора переднего плана // Конф.-конкурс «Технологии Microsoft в теории и практике программирования»: Тез. докл. — Новосибирск, 2006. — С. 37–39.
17. **Стасенко А. П.** Визуализация внутреннего представления системы функционального программирования в ActiveX компоненте // Тр. XLIV Междунар. науч. студенческой конф. «Студент и научно-технический прогресс»: Информационные технологии. — Новосибирск: НГУ, 2006. — С. 135–136.
18. **Стасенко А. П.** Система функционального программирования языка Sisal // XII Международная научно-методическая конференция «Новые информационные технологии в университетском образовании». — Новосибирск: НГУ, 2007. — С. 162–165.
19. **Стасенко А. П.** Внутреннее представление системы функционального программирования Sisal 3.0. — Новосибирск, 2004. — 54 с. — (Препр. / РАН. Сиб. отд-е. ИСИ; № 110).
20. **Стасенко А. П., Синяков А. И.** Базовые средства языка Sisal 3.1. — Новосибирск, 2006. — 60 с. — (Препр. / РАН. Сиб. отд-е. ИСИ; № 132).

Стасенко А. П.