МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатика и вычислительная техника и прикладная математика

КУРСОВАЯ РАБОТА

по дисциплине Теории языков программирования и методам трансляции

на тему: «Программная реализация транслятора»

Выполнил ст. гр. ИВТ-18
Втулкин В.В.

Проверил доцент кафедры ИВТ и ПМ
Коган Е.С.

Чита
2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатика и вычислительная техника и прикладная математика

ЗАДАНИЕ

на курсовую работу

По дисциплине: Теория языков программирования и методы трансляции
Студенту Втулкину Владиславу Викторовичу

специальности 09.03.01 Информатика и вычислительная техника

1. Тема курсовой работы: Программная реализация транслятора
2. Срок подачи студентом законченной работы: 23 декабря 2021 г.
3. Исходные данные к работе: литературные источники, источники в сети интернет
4. Перечень подлежащих в курсовой работе вопросов:
   а) Постановка и анализ задачи;
   б) Анализ данных;
   в) Программная реализация;
5. Перечень графического материала (если имеется): -

Дата выдачи задания «9» сентября 2021г.

Руководитель курсовой работы_____/Коган.Е.С./
(подпись, расшифровка подписи)

Задание принял к исполнению

«__» _____ 20__ г.

Подпись студента _____ / Втулкин В.В./

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатика и вычислительная техника и прикладная математика

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по 09.03.01 Информатика и вычислительная техника

на тему: «Программная реализация транслятора»

Выполнил студент группы ИВТ-18 Втулкин В.В.

Руководитель работы: доцент кафедры ИВТ и ПМ Коган Евгения Семёновна

РЕФЕРАТ

Пояснительная записка 18 – страниц, 1 - приложений.

ТРАНСЛЯТОР, C#, C++, Go, .NET, ЛЕКСИЧЕСКИЙ АНАЛИЗАТОР, СИНТАКСИЧЕСКИЙ АНАЛИЗАТОР, ГРАММАТИКА, ЛЕКСЕМА.

В работе описывается создание транслятора с языка программирования C++ на язык Go. Разработка транслятора производится на .NET с использованием шаблона консольного приложения. В работе рассматриваются понятие транслятора, принципы построения транслятора.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время искусственные языки, использующие для описания предметной области текстовое представление, широко применяются не только в программировании, но и в других областях. С их помощью описывается структура всевозможных документов, трехмерных виртуальных миров, графических интерфейсов пользователя и многих других объектов, используемых в моделях и в реальном мире. Для того, чтобы эти текстовые описания были корректно составлены, а затем правильно распознаны и интерпретированы, используется специальный вид программ, именуемый трансляторами. Разработка трансляторов на сегодняшний момент является одной из самых актуальных областей в разработке программного обеспечения.

В данной курсовой работе рассматривается процесс разработки транслятора с языка блок-схем в язык программирования Pascal.

1 Постановка и анализ задачи

1.1 Описание предметной области

Транслятор — программа или техническое средство, выполняющее трансляцию программы.

Трансляция программы— преобразование программы, представленной на одном из языков программирования, в программу на другом языке. Транслятор обычно выполняет также диагностику ошибок, формирует словари идентификаторов, выдаёт для печати текст программы и т. д.

Язык, на котором представлена входная программа, называется исходным языком, а сама программа — исходным кодом. Выходной язык называется целевым языком.

Одна из важных ролей транслятора состоит в сообщении об ошибках в исходной программе, обнаруженных во время трансляции.

Процесс отображение входной программы в эквивалентную ей выходную программу состоит из двух частей: анализ и синтез.

Анализ разбивает входную программу на составные части и накладывает на них грамматическую структуру. Затем использует эту структуру для создания промежуточного представления входной программы. Если анализ обнаруживает, что входная программа неверно составлена синтаксически или семантически, он должен выдать информативные сообщения об этом, чтобы пользователь мог исправить обнаруженные ошибки. Анализ также собирает информацию об исходной программе и сохраняет её в структуре данных, называемой таблицей идентификаторов (символов), которая передаётся вместе с промежуточным представлением синтезу.

Синтез строит требуемую целевую программу на основе промежуточного представления и информации из таблицы идентификаторов.

Анализ часто называют начальной стадией, а синтез – заключительной.

Если посмотреть на процесс трансляции более детально, можно увидеть, что он представляет собой последовательность фаз, каждая из которых преобразует одно из представлений входной программы в другое. На практике некоторые фазы могут быть объединены, а межфазное промежуточное представление может не строиться явно. Таблица идентификаторов, в которой хранится информация обо всей входной программе, используется всеми фазами транслятора.

1.2 Постановка задачи

Разрабатываемый транслятор будет иметь возможность выполнять трансляцию с языка программирования C++ на Go.

При трансляции транслятор будет выполнять фазы лексического анализа, синтаксического анализа, а также генерации целевого исходного кода. Важно отметить, что на каждой из фаз транслятор должен осуществлять жёсткий контроль ошибок в анализируемом им исходном коде и по мере их нахождения уведомлять об этом пользователя.

1.3 Средства реализации

Реализация выполнена на программной платформе .NET с помощью консольного шаблона на языке программирования C#.

1.4 Описание языка программирования C++

C++ (читается си-плюс-плюс) — компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое

программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков.[4][5] В сравнении с его предшественником — языком C — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.[5]

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования.

1.5 Структура языка C++

В C ++ точка с запятой является терминатором утверждения. То есть каждое отдельное утверждение должно заканчиваться точкой с запятой. Он указывает конец одного логического объекта. Блок представляет собой набор логически связанных операторов, которые окружены открывающимися и закрывающимися фигурными скобками.

Идентификатор C ++ - это имя, используемое для идентификации переменной, функции, класса, модуля или любого другого пользовательского элемента. Идентификатор начинается с буквы от A до Z или от a до z или символа подчеркивания (_), за которым следует ноль или несколько букв, символов подчеркивания и цифр (от 0 до 9).

C ++ не допускает знаков препинания, таких как @, $ и% в идентификаторах. C ++ - это язык программирования с учетом регистра. Таким образом, Manpower и рабочая сила являются двумя разными идентификаторами на C ++.

1.6 Описание языка программирования Go

Go представляет компилируемый статически типизированный язык программирования от компании Google. Язык Go предназначен для создания различного рода приложений, но прежде всего это веб-сервисы и клиент-серверные приложения. Хотя также язык обладает возможностями по работе с графикой, низкоуровневыми возможностями и т.д.

Работа над языком Go началась в 2007 в недрах компании Google. Одним из авторов является Кен Томпсон, который, к слову, является и одним из авторов языка Си (наряду с Денисом Ритчи). 10 ноября 2009 года язык был анонсирован, а в марте 2012 года вышла версия 1.0. При этом язык продолжает развиваться. Текущей версией на момент написания данной статьи является версия 1.17, которая вышла в августе 2021 года.

Язык Go развивается как open source, то есть представляет проект с открытым исходным кодом, и все его коды и компилятор можно найти и использовать бесплатно. Официальный сайт проекта - https://go.dev/, где можно много полезной информации о языке.

Go является кроссплатформенным, он позволяет создавать программы под различные операционные системы - Windows, Mac OS, Linux, FreeBSD. Код обладает переносимостью: программы, написанные для одной из этих операционных систем, могут быть легко с перекомпиляцией перенесены на другую ОС.

Основные особенности языка Go:

- компилируемый - компилятор транслирует программу на Go в машинный код, понятный для определенной платформы
- статически типизированный
- присутствует сборщик мусора, который автоматически очищает память
- поддержка работы с сетевыми протоколами
- поддержка многопоточности и параллельного программирования

В настоящее время Go находит широкое применение в различных сферах. В частности, среди известных проектов, которые применяют Go, можно найти следующие: Google, Dropbox, Netflix, Kubernetes, Docker, Twitch, Uber, CloudFlare и ряд других.

2 Анализ данных

Данные, с которыми работает транслятор можно разделить на несколько групп. Входные данные – текст программы на C++. Промежуточные данные – это данные, используемые транслятором во время работы. Выходные данные – текст программы на Go.

2.1 Промежуточные данные

Трансляция одного языка в другого происходит постепенно. Сначала текст программы на C++ поступает на вход лексическому анализатору. После чего на выходе появляются таблицы идентификаторов и лексем. Данные таблицы поступают на вход синтаксическому анализатору, который генерирует дерево разбора.

Лексема - это структурная единица языка, состоящая из элементарных символов языка и не содержащая других структурных единиц языка. Лексемы языка программирования - это идентификаторы, константы, ключевые слова языка, знаки операций и т. п.

Таблица лексем содержит весь текст исходной программы, обработанный лексическим анализатором. В неё входят все возможные типы лексем, при этом, любая лексема может в ней встречаться любое число раз.

Таблица идентификаторов содержит только следующие типы лексем: идентификаторы и константы. В неё не попадают ключевые слова входного языка, знаки операций и разделители. Каждая лексема в таблице идентификаторов может встречаться только один раз.

Результат грамматического анализа. Дерево разбора отличается от абстрактного синтаксического дерева наличием узлов для тех синтаксических правил, которые не влияют на семантику программы.

3 Программная реализация

Основная функция компилятора – отображение входной программы в эквивалентную ей исполняемую программу. Это отображение разделяется на две части: анализ и синтез.

Анализ разбивает входную программу на составные части и накладывает на них грамматическую структуру. Затем он использует эту структуру для создания промежуточного представления входной программы. Если анализ обнаруживает, что входная программа неверно составлена синтаксически либо семантически, он должен выдать информативные сообщения об этом, чтобы пользователь мог исправить обнаруженные ошибки. Анализ также собирает информацию об исходной программе и сохраняет её в структуре данных, именуемой таблицей идентификаторов (символов), которая передаётся вместе с промежуточным представлением синтезу.

Синтез строит требуемую целевую программу на основе промежуточного представления и информации из таблицы идентификаторов. Анализ часто называют начальной стадией, а синтез – заключительной.

Если рассмотреть процесс компиляции более детально, можно увидеть, что он представляет собой последовательность фаз, каждая из которых преобразует одно из представлений входной программы в другое. Типичное разложение компилятора на фазы приведено на рисунке 3. На практике некоторые фазы могут объединяться, а межфазное промежуточное представление может не строиться явно. Таблица идентификаторов, в которой хранится информация обо всей входной программе, используется всеми фазами компилятора. Первая фаза компиляции называется лексическим анализом или сканированием. Лексический анализатор читает поток символов, составляющих входную программу, и группирует эти символы в значащие последовательности, называемые лексемами. Лексемы передаются последующей фазе компилятора, синтаксическому анализу.

Вторая фаза компилятора – синтаксический анализ или разбор. Во время выполнения этой фазы используются лексемы, полученные от лексического анализатора, для создания древовидного промежуточного представления программы, которое описывает грамматическую структуру потока лексем. Типичным промежуточным представлением является синтаксическое дерево, в котором каждый внутренний узел представляет операцию, а дочерние узлы – аргументы этой операции.

Семантический анализатор использует синтаксическое дерево и информацию из таблицы идентификаторов для проверки входной программы на семантическую согласованность с определением языка программирования. Он также собирает информацию о типах и сохраняет её в синтаксическом дереве или в таблице идентификаторов для последующего использования в процессе генерации промежуточного кода.

После синтаксического и семантического анализа исходной программы компиляторы генерируют низкоуровневое промежуточное представление входной программы, которое можно рассматривать как программу для абстрактной вычислительной машины. Такое промежуточное представление должно обладать двумя важными свойствами: оно должно легко генерироваться и легко транслироваться в целевой машинный язык.

Некоторые компиляторы содержат фазу машинно-независимой оптимизации. Назначение этой оптимизации – улучшить промежуточный код, чтобы затем получить более качественный целевой код. Обычно «более качественный», «лучший» означает «более быстрый», но могут применяться и другие критерии сравнения, как, например, «более короткий код» или «код, использующий меньшее количество ресурсов». Генератор кода получает в качестве входных данных промежуточное представление исходной программы и отображает его в целевой язык. Важная функция компилятора состоит в том, чтобы записывать имена переменных входной программы и накапливать информацию о разных атрибутах каждого имени. Эти атрибуты могут

предоставлять информацию о выделенной памяти для данного имени, его типе, области видимости (где именно в программе может использоваться его значение) и, в случае имён процедур, такие сведения, как количество и типы их аргументов, метод передачи каждого аргумента (например, по значению или по ссылке), а также возвращаемый тип. Таблица идентификаторов представляет собой структуру данных, содержащую записи для каждого имени переменной с полями для атрибутов имени. Структура данных должна быть разработана таким образом, чтобы позволять компилятору быстро находить запись для каждого имени, а также быстро сохранять данные в записи и получать их из неё.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были получены навыки по проектированию программ, их отладке и документированию.

В результате работы была написана программа – транслятор с языка программирования C++ на язык программирования Go.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Wikipedia. Транслятор [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Транслятор свободный
2. Wikipedia. Лексический анализ [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Лексический_анализ свободный
3. Wikipedia. Синтаксический анализатор [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Синтаксический_анализатор свободный
4. Wikipedia. C++ [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/C%2B%2B свободный
5. Metanit. Руководство по языку Go [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://metanit.com/go/tutorial свободный
6. Молдованова О.В. Языки программирования и методы трансляции: учеб. пособие / О.В. Молдованова. – Новосибирск: ФГОБУ ВПО «СибГУТИ», 2012. – 39 с.

Приложение А

(обязательное)

Пример работы программы.