Московский Авиационный Институт
(Национальный исследовательский университет)

Факультет информационных технологий и прикладной математики
Кафедра №806 Вычислительная математика и программирование

Реферат

по курсам
«Архитектура компьютера», «Программные и аппаратные средства информатики»
I семестр

«Эдсгер Вибе Дейкстра»

Студент: Рамалданов Р. Р.

Группа: М8О-108Б-22

Номер по списку: 17

Руководитель: Сахарин Н. А.

Оценка: <…>

Дата: <…>

Подпись преподавателя:

Москва, 2023

СОДЕРЖАНИЕ

Введение

Целью данной курсовой работы является самостоятельное изучение биографии известного нидерландского ученого – Эдсгера Вибе Дейкстры. Его труды оказали влияние на развитие информатики и информационных технологий. Он является одним из разработчиков концепции структурного программирования, исследовал формальную верификацию и распределённые вычисления.

Биография

Эдсгер Вибе Дейкстра родился в Роттердаме (Нидерланды) в мае 1930 года в семье научных работников: отец будущего лауреата Тьюринговской премии был химиком, мать – математиком что, видимо, и предопределило выбор Дейкстры поступить на отделение математики и теоретической физики Лейденского университета. Еще учась в университете Дейкстра познакомился с первыми компьютерами и увлекся их программированием. Этому немало способствовало и то, что будучи еще студентом Дейкстра с 1952 года работал программистом в Математическом центре Амстердама. За год до окончания университета Дейкстра оказался перед дилеммой: продолжить научную карьеру по основной специальности – теоретической физике или все-таки продолжать заниматься программированием. Его научный руководитель все-таки убедил его выбрать программирование.

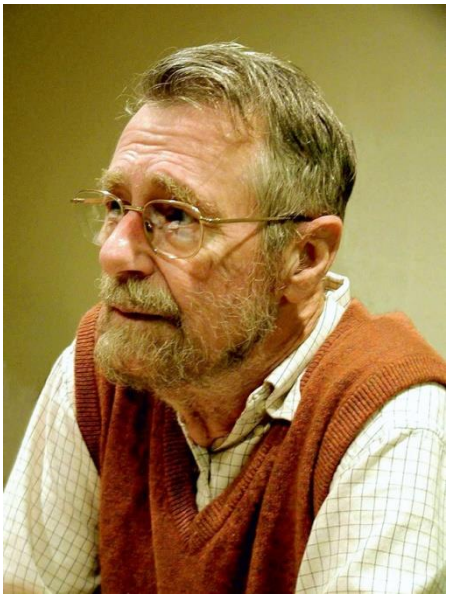

Рисунок 1. Эдсгер Дейкстра

В 1957 году женился, по собственным воспоминаниям, в графе «профессия» анкеты, которую положено заполнять при бракосочетании, написал «программист» — и его заставили переписывать документы, заявив, что такой профессии не существует, в результате пришлось указать «физик-теоретик».

В обычной жизни Э.В.Дейкстра был по-хорошему «чудаковат»: предпочитал простую ручку компьютеру, в его доме не было телевизора, он не пользовался мобильным телефоном, не смотрел кино. Когда его коллеги подготовили и издали к 60-летнему юбилею специальный сборник, Дейкстра ответил каждому из них личным благодарственным письмом, написанным от руки (а это, между прочим – 61 адресат). Ученому его уровня и положения полагался секретарь, но Дейкстра отказался от этой привилегии и все предпочитал делать сам. Любил музыку и был хорошим пианистом. В августе 2002 года Эдсгер Вибе Дейкстра скончался в своем доме в Нидерландах.

Алгоритм кратчайшего пути на графе

В 1959 году Дейкстра предложил алгоритм поиска кратчайшего пути от любой вершины ориентированного взвешенного графа до любой другой. Алгоритм появился в результате поиска решения оптимальной разводки соединений плат компьютера.

Алгоритм Дейкстры широко применяется и сегодня (например, при планировании автомобильных и авиа-маршрутов, при разводке электронных плат, в протоколах маршрутизации). Является примером классического «жадного» алгоритма.

ALGOL-60

Порядком намаявшись в начале своей программистской карьеры с машинными кодами и с тем, что для различных моделей компьютеров один и тот же алгоритм нужно было переписывать практически с нуля, Эдсгер Дейкстра не мог не «ухватиться» за языки программирования высокого уровня. FORTRAN, появившийся в конце 50-х годов Дейкстру не очень привлекал: в FOTRAN-е многое было принесено в жертву главной цели - во чтобы то ни стало реализовать высокоуровневый язык и избавить программиста от «проклятья» двоичных кодов. Появись FORTRAN сегодня, его шансы удержаться, думаю, были бы весьма сомнительными. Но тогда, безусловно, FORTRAN был великим шагом вперед. Однако, Дейкстре все равно FORTRAN не импонировал – ему в этом языке, по-видимому, не

хватало того изящества и логичности конструкций, которые Дейкстра привык видеть в математике и логике.

Другое дело ALGOL-60. Язык описывался стройной и вполне строгой нотацией (т.н. называемая «форма Бэкуса-Наура»), его разработка велась едва ли не в академической среде с присущими последней требованиями четкости, ясности и доказуемости. Критерии были строгими и язык поэтому получился изящный (не случайно такие языки программирования как PL/1, PASCAL и ADA носят явные следы влияния ALGOL-a). Не мешкая Дейкстра приступил к реализации компилятора и успех в этом направлении подтвердил его давнюю мысль – программировать надо на «нормальных» языках, приспособленных, насколько это возможно, к психологии человека. А машинный код – ну что же, раз без него все равно никуда не деться – его можно и нужно оставить машинам.

Одной из наиболее сложных задач при трансляции языков программирования в те годы была задача компиляции арифметических выражений с учетом приоритетов операций и скобок. Дейкстра убедительно обосновал и упростил предложенный в 1957г. Ф.Брауэром и К.Замельзоном алгоритм использования для этой цели двух стеков (тогда обычно говорили «магазин» по аналогии с магазином оружия). Арифметические выражения эффективно транслировались в обратную польскую запись (постфиксную) очень удобную для генерации объектного кода.

Борьба за ресурсы или взаимодействие процессов

Разработчики первых операционных систем, которые и операционными системами можно назвать с большой натяжкой, работавших в пакетном режиме (когда одному процессу полностью «принадлежали» все ресурсы компьютера) почти не сталкивались с задачей, которая к середине 60-х годов прошлого века стала чрезвычайно актуальной – обеспечение доступа нескольких процессов к общим ресурсам и разделение этих ресурсов между процессами. Без этого нельзя было решить важнейшую задачу – одновременное выполнение нескольких процессов на одном компьютере.

Очень простая по формулировке проблема, как это часто бывает, оказалась далеко не простой для решения. Были предложены несколько алгоритмов, позволявших отследить использование общих ресурсов (прежде всего – оперативной памяти и устройств ввода/вывода) различными процессами. Однако, их более внимательный анализ показал, что в операционных системах, использующих эти алгоритмы, никак не удавалось избежать блокирования процессами друг друга. В отдельных случаях, когда блокировок удавалось избежать, ресурсы оказывались не в том состоянии, в каком ожидалось (не удавалось обеспечить т.н. «атомарность» операций). Система либо намертво стопорила саму себя, либо начинала вести себя непредсказуемо (Дейкстра дал таким состояниям выразительное название - «дедлок»). Разумеется, такая нестабильность и непредсказуемость поведения основной программы – операционной системы – никого не могла устроить.

Наиболее полно свои мысли по этому поводу Дейкстра изложил в статье «Взаимодействие последовательных процессов». Для решения этой проблемы Дейкстра предложил концепцию «семафора» - специальных целочисленных общих переменных и двух примитивов «P-операция» и «V-операция». Вот как сам Дейкстра описывает эти примитивы:

«Эти две последнии операции всегда выполняются над семафорами и представляют единственный способ обращения к семафорам со стороны одновременно действующих процессов.

Семафоры являются по существу неотрицательными целыми величинами. Если они используются только для решения задачи взаимного исключения, то область их значений составляют лишь «0» и «1».»

V-операция состоит в увеличении значения аргумента (который является семафором) на 1. Такое увеличение обязано быть неделимой операцией. P-операция, напротив, будучи применной к семафору, уменьшает (также неделимым образом) значение последнего на 1. Поскольку семафор, согласно определению, число неотрицательное, то рано или поздно P-операция уменьшит его значение до 0. Процесс, инициировавший P-операцию вынужден будет ждать (т.е. фактически окажется в режиме задержки выполнения) до тех пор пока другой процесс не «сдвинет» значение этого семафора в положительную область, т.е. разблокирует семафор.

Эти концепции и их дальнейшее развитие – например, мьютексы – и сегодня применяются при проектировании и реализации операционных систем.

В эти же годы Эдсгер Дейкстра руководил созданием операционной системы THE (от «Technische Hogeschool Eindhoven») с поддержкой многозадачности. Операционная система была построена в виде иерархии из 6 уровней; при этом более низкие уровни (начиная с 0) служили базой для более высоких (вплоть до 5). На начальных уровнях были реализованы система обработки прерываний, семафоры, переключение контекстов процессов, система управления памятью, диспетчер. На средних уровнях были реализованы взаимодействие с консолью, ввод/вывод, взаимодействие с устройствами; самый верхний уровень предназначался для пользовательских программ. В дальнейшем такая модель организации операционной системы получила широкое распространение.

Для иллюстрации проблемы разделения/блокирования ресурсов и взаимодействия процессов Дейкстра предложил простую и остроумную модель; позже эта модель получила имя «задачи о пяти обедающих философах». Вот она в изложении Ч.Хоара:

«В давние времена один богатый филантроп пожертвовал свой капитал на основание некоего пансиона, чтобы дать пристанище пяти знаменитым философам. У каждого философа была своя комната, где он мог предаваться размышлениям. Была у них и общая столовая с круглым столом, вокруг которого стояли пять стульев, каждый помеченный именем того философа, которому он предназначался... Слева от каждого философа лежала золотая вилка, а в центре стояла большая миска спагетти, содержимое которой постоянно пополнялось.

Предполагалось, что большую часть времени философ проводил в размышлениях, а почувствовав голод, шел в столовую, садился на свой стул, брал вилку слева от себя и приступал к еде. Но такова уж сложная природа спагетти, что их не донести до рта без помощи второй вилки. Поэтому философу приходилось брать вилку и справа от себя... Если вилка требовалась другому философу, ему приходилось ждать, пока она освободится.»

Структурное программирование

Начну с цитаты Эдсгера Дейкстры из его знаменитой статьи «Заметки по структурному программированию»:

«Речь идет о построении больших программ, запись которых может оказаться, скажем, такого же размера, как весь текст этого раздела... Было бы прекрасно, если бы мне удалось проиллюстрировать различные методы на примерах малых программ, а затем закончить фразой: «... и столкнувшись с программой, в тысячу раз большей, вы построите ее точно таким же способом». Однако ... один из тезисов моей работы в том и состоит, что любые два объекта, которые отличаются в чем-то по меньшей мере в сто или более раз, становятся совершенно несопоставимыми.»

Проникновение компьютеров во все отрасли промышленности, бизнеса, образования и проч., а также сопутствующий этому проникновению рост объема разрабатываемого программного обеспечения вызывал обеспокоенность ведущих специалистов: программы становились все больше, все сложнее, все разнообразнее. Все это не смогло не сказаться на их качестве – оно стремительно падало.

Досадно, если ошибка была в программе начисления заработной платы, но это, в конце-концов дело поправимое. Страшно, если программная ошибка обнаруживалась в системе управления воздушным транспортом. И совсем уж катастрофической была бы ошибка в программе управления работой атомного реактора.

«Я считаю, что программа никогда не является самоцелью; программа предназначается для того, чтобы вызвать вычисления, а цель вычислений – получить нужный результат... Я утверждаю (хотя и не могу доказать), что легкость и гибкость таких наших суждений существенно зависит от простоты взаимосвязей между программой и вычислениями... Грубо говоря, можно считать желательным, чтобы структура программы отражалась в структуре вычислений.»

Для обеспечения указанных легкости и гибкости Дейкстра предлагает проектировать и кодировать программы в соответствии с определенной

дисциплиной, названной им структурным программированием. Известно (это математический факт известный как теорема Бойма-Якопини), что любую программу можно построить с использованием трех конструкций: последовательного выполнения и операторов выбора и цикла. Дейкстра предложил (одновременно, хотя и независимо с Никлаусом Виртом – еще одним лауреатом премии Тьюринга) представлять программу как иерархическую структуру блоков, каждый из которых выполняет небольшую, но завершенную задачу. Это достигается с помощью механизма процедур и функций. Т.о. программы приобретали модульность.

Идеи структурного программирования, поначалу встреченные с недоверием, довольно скоро завоевали признание (особенно, после создания Никлаусом Виртом языка программирования PASCAL). Более того, эта методика остается актуальной и сегодня (достаточно вспомнить такие популярные языки программирования C, PASCAL или BASIC).

Заключение

Благодаря данному реферату была изучена биография известного ученого, Эдсгара Вибе Дейкстры, и его достижения в области информатики. Ознакомление с трудами этого человека играет важную роль, именно благодаря нему информатика сейчас является такой, какая она есть.

Список используемых источников

1. Журнал системный администратор – Эдсгер Вибе Дейкстра. Смиренный гений программирования [Электронный ресурс] /. – Электрон. текстовые дан. – Режим доступа: http://samag.ru/uart/more/16 свободный. Дата посещения 21.01.2023
2. Habr - Эдсгер Дейкстра: в поисках «кратчайшего пути» к осознанному программированию [Электронный ресурс] /. – Электрон. текстовые дан. – Режим доступа: https://habr.com/ru/post/303712/ свободный. Дата посещения 21.01.2023