И. В. Поттосин

ИСТОРИЯ АЛЬФА-ПРОЕКТА

Роль Альфа-проекта

Альфа-проект занимает особое место в истории новосибирской школы программирования. Именно с него начинается всесоюзная и мировая известность этой школы. Это была первая работа, которой мы заявили о себе как о сильном коллективе, умеющем решать сложные и нужные задачи. Это был первый в мире оптимизирующий транслятор с языков типа Алгол.

Конечно, и до этого существовали оптимизирующие трансляторы: и отечественные ПП (программирующие программы), и зарубежные Фортран-трансляторы, но они оптимизировали программы, написанные на куда более простых и приспособленных для такой оптимизации языках. Алгол 60 разрабатывался как "естественный математический" язык, без внимания к получению эффективных программ. Мало того, некоторые молчаливо предполагали, что за естественность формулирования алгоритмов следует платить недостаточной эквивалентностью. Это давало возможность некоторым отечественным программистам говорить на программистских форумах, что Алгол 60 — это западная провокация, предназначенная для того, чтобы завлечь нас в тупик: "Они-то будут писать на Фортране или мнемокодах, а мы, поддавшись влиянию Запада, будем создавать неэффективные программы". Так действительно было.

Алгол 60 создавал ряд новых, неизведанных проблем для трансляции, и решению этих, собственно трансляционных, проблем и было посвящено большинство западных и отечественных реализаций Алгола 60. Многие считали, что создание Алгол-трансляторов, дающих достаточно эффективный код, невозможно, и в процессе нашей работы мы неоднократно и на разных уровнях слышали, что взялись за неразрешимую задачу. Надо честно сказать, что сложность задачи мы недооценили, сроки, которые мы намечали вначале, не выдерживались, и наше непосредственное начальство, считая проект нереализуемым, пыталось его прекратить. Только личная поддержка С. Л. Соболева, нашего директора, спасла нас. Ситуация усугублялась тем, что мы реализовывали

не Алгол 60, а его расширение, которое включало конструкции, существенно усложнявшие как оптимизацию, так и собственно трансляцию.

Успешная, хоть и затянувшаяся, реализация проекта была конструктивным доказательством того, что с языков типа Алгол возможна эффективная реализация, что естественность конструкций не противоречит порождению эффективного кода, но требует изощренной техники оптимизации. Начало создания такой техники и было положено в Альфа-проекте, хотя она и опиралась на предыдущие работы по ПП и Фортран-трансляции.

Альфа-язык

В 1958 году международная группа опубликовала начальную версию нового языка программирования, основной идеей которого было дать как можно более естественную форму для выражения алгоритмов (прежде всего, вычислительной математики). Опираясь на эту версию, Андрей Петрович Ершов вместе с Геной Кожухиным и Димой Янковым решили разработать свой язык, внеся в него то, что было бы удобно вычислителям, в частности, возможность оперировать с векторами, матрицами и многомерными объектами. Когда своя версия была готова, появилась окончательная версия международного языка Алгол 60. Ряд решений в Алголе 60 совпадали с принятыми нашей группой, но ряд существенных конструкций, которые были разработаны в "сибирском" языке, отсутствовали в Алголе 60. Было принято решение удалить все несущественные различия из "сибирского" языка, но существенные — оставить, сделав из него правильное расширение Алгола 60. Дима Янков к этому времени ушел из института, и работу по созданию окончательной версии языка выполнила группа в составе Ершова, Кожухина (автора первого перевода Алгола 60 на русский язык) и вновь пришедшего в институт Юлика Волошина.

Новый язык в разное время носил разные названия. Сначала он носил название "входной" (в английском переводе Input Language), затем — скорее обиходное, чем официальное — "сибирский" и, наконец, уже установившееся и официальное имя — язык Альфа.

Как уже говорилось, Альфа-язык был расширением Алгола 60, более того, расширением некоторого его подмножества. Основные ограничения этого подмножества были связаны с главной задачей — обеспечить высокоэффективный код. Все, что мешало эффективной генерации кода или требовало сложного анализа при проведении оптимиза-

ций, удалялось. Так, требовалась обязательная спецификация параметров-меток, запрещались именующие выражения и идентификаторы стандартных функций как фактические параметры, ограничивалось использование переменных с индексом в качестве параметров циклов, при неопределенном значении указателя переключателя переход был неопределен и вводились другие тому подобные ограничения, которые отсекали всякую трудно реализуемую и трудно понимаемую экзотику. По-настоящему серьезных ограничений было два: были запрещены рекурсивные процедуры, так как мы не нашли хорошего метода реализации таких процедур, а реализовать каждую процедуру как потенциально рекурсивную (что необходимо, если не проводить сложного анализа) было бы накладно; побочный эффект вызовов функций игнорировался: мы хотели избавиться от сложного анализа, когда он влияет на корректность оптимизаций, а когда не влияет.

Наряду с этими, не так уж серьезными (кроме отказа от рекурсивности), ограничениями, Альфа-язык содержал ряд принципиальных новшеств. Большинство из них увеличивали выразительную силу языка для вычислительных задач.

Впервые для языков программирования арифметические типы были дополнены типом комплексный с введением всей комплексной арифметики. Также впервые было введено понятие многомерного значения (массива значений) и многомерной переменной, обладающей этим значением целиком — до сих пор в языках действия с массивами должны были трактоваться как некоторая совокупность действий над их компонентами, в Альфа-языке появилась крайне удобная и широко использовавшаяся возможность выполнять естественные операции над всем многомерным значением целиком и целиком же присваивать это значение многомерной переменной. Нововведение было явно выигрышным и в ряде последующих языков — Алголе 68, ПЛ/1, Аде — были введены подобные же средства. В языке появились средства создания многомерных объектов — значений и переменных: образования подмассивов таких объектов (“вырезок” в терминах последующих языков), объединения объектов в новый многомерный объект (с повышением размерности или без). Последние средства, так называемые геометрические операции, несмотря на свою полезность, так по-настоящему и не прижились в последующих языках — исключением, пожалуй, тут является язык Мучника и Шафаренко Eval. По-видимому, причина здесь в том, что для полной и естественной реализации подобных операций объеди-

нения (геометрических операций) надо глубокое проникновение семантики, связанной с многомерными значениями с некоторой структурой, в язык, что было характерно и для Альфа, и для Eval.

В вычислительных задачах зачастую используются такие механизмы, как рекуррентность и итерации. Эти механизмы характерны тем, что надо иметь несколько последовательных (упорядоченных, так сказать, по времени) экземпляров объекта и при каждом получении нового объекта сдвигать эту последовательность. Для естественного выражения итерации и рекуррентности в Альфа-язык было введено ни в каком языке более не повторенное понятие объектов с верхним (временным) индексом, для которых операции по заведению последовательности и ее сдвигу реализовывались по умолчанию.

Впервые в Альфа-языке были введены циклы без параметра и описания начальных значений, что сейчас присутствует во многих языках. Из Фортрана было позаимствовано такое наглядное средство, как функции-выражения (тот простой случай, когда значение функции определяется единственным выражением). В качестве способов подстановки параметров процедуры была добавлена подстановка результатом (out-параметр Ады). Логические выражения могли включать цепочки равенств или неравенств (вида a<b<=c<d=e).

Еще одно нововведение в Альфа-языке — это перечисления. В целом ряде конструкций языка — цепочке неравенств, цепочке присваиваний, цепочке описания начальных значений, геометрических операциях — допускались перечисления, причем во всей их полноте (например, A[i+1,j] := ... := a [i+1,m] := 0).

Интересно, что для задания синтаксиса перечислений пришлось прибегнуть к контекстно-зависимым правилам грамматик — это был первый пример использования контекстно-зависимой грамматики для описания синтаксиса языка (Алгол 68, конечно, в большей мере использовал механизмы контекстно-зависимых грамматик, но он был позже).

Конкретный входной язык системы Альфа содержал еще ряд дополнений Альфа-языка, учитывавших, в том числе, и конкретную архитектуру объектной машины (М-20 или БЭСМ-6), и сложившийся состав программного обеспечения. Для систем Альфа и Альфа-6 эти дополнения несколько различаются, в дальнейшем я буду говорить только о входном языке системы Альфа.

Одним из важных дополнений было введение библиотеки процедур. Насколько мне известно, в системе Альфа впервые было введено такое

понятие, которое в современных системах выражается библиотеками модулей, пакетов, классов. Состав библиотек никак не стандартизировался, он мог быть определен при установке системы, для этого существовала своя технология. Те процедуры из библиотеки, вызов которых встречался в программе, подключались, как это сейчас принято, к ней как объекты самого внешнего контекста.

Надо заметить, что нам было легко пойти на это новшество: процедуры реализовывались Альфа-системой максимально эффективным образом, а влияние их вызовов на реализацию вызывающей программы учитывалось настолько адекватно, насколько возможно. Разумеется, это достигалось не только за счет изощренного программирования процедур, но и за счет тех ограничений Алгола 60 (рекурсивность и побочный эффект), на которые мы сознательно пошли. В результате пользователи системы могли вовсю пользоваться таким могучим изобразительным средством, как процедуры, не опасаясь, что это ухудшит им эффективность рабочих программ. В качестве контраста вспоминается, что в руководстве по пользованию другой системой программирования для Алгола 60, где процедуры реализовывались во всей своей мощности, в рекомендациях пользователям указывалось, что во избежание существенного ухудшения эффективности пользоваться процедурами не рекомендуется. Что поделаешь, техника программирования процедур тогда еще делала первые шаги, а сами процедуры в Алголе 60 несли ряд неконструктивных, плохо реализуемых черт (например, подстановка именем), от которых последующие языки избавились.

Учет архитектуры и программного обеспечения реализовывался такими средствами входного языка, как использование библиотеки ИС-2 (набор подпрограмм в машинном коде, динамически вызываемых и связываемых с рабочей программой) и машинных команд. Синтаксически обращение к стандартной подпрограмме из библиотеки носило вид вызова процедуры, а так как состав библиотеки был фиксирован, то компилятор знал свойства данной подпрограммы, нужные для потокового анализа. Позволялось использовать единичные машинные команды как операторы входного языка. Так как эти операторы включались в общий потоковый анализ и не должны были мешать оптимальной генерации кода, то их употребление должно было подчиняться некоторой дисциплине.

Так как оперативная память ЭВМ М-20 была маленькой (сейчас это выглядит совсем удивительно — всего 4К 45-разрядных слов), то надо

было ввести механизмы, поддерживающие работу программ с довольно большими требованиями на память. В системе программирования ТА-2 (тоже для М-20) решением стало создание виртуальной памяти, что не требовало никаких усилий от пользователей, но влекло накладные расходы. Мы пошли по другому пути: заставили пользователя "укладываться" в оперативную память, самостоятельно фрагментируя программу и данные с выделением внешних (располагаемых во внешней памяти — для М-20 это барабан и ленты) массивов и блоков. Внешние блоки скачивались с внешней памяти и стирались по мере необходимости автоматически (run-time системой), а внешние массивы надо было явно копировать в буферные массивы в оперативной памяти (или из них).

Последним по времени новшеством во входном языке было введение средств комплексации. Комплексировались такие программы, каждая из которых занимала всю оперативную память, а все информационные связи между ними могли осуществляться только через внешние массивы. Все программы в комплексе нумеровались, и этот фиксированный номер использовался для явного указания динамического преемника данной программы комплекса. Комплексирование позволяло создавать сложные (по тем временам) программные системы, и его необходимость мы осознали уже после того, как система достаточно широко пошла по стране. Основным толчком к введению комплексирования была здоровая критика вычислителей из Отдела прикладной математики (теперь ИПМ РАН им. М. В. Келдыша) — Г. Лимана, Л. Майорова, М. Степановой и Л. Фрязиновой. Они приехали из Москвы со своеобразной инспекцией — посмотреть, насколько Альфа-система подходит для их больших задач (а у них в ОПМ тогда действительно были очень большие по тем временам вычислительные задачи). Свои эксперименты с системой они проводили в очень дружественной атмосфере, качество получаемого кода им весьма понравилось, а вот в то Прокрустово ложе, связанное с ограниченной памятью, их серьезные задачи не вкладывались. Такие живые примеры нас убедили, и (чего не сделаешь для друзей) мы пошли на нужные расширения языка и развитие системы.

Разработка Альфа-транслятора

Создание Альфа-транслятора началось с того, что А. П. Ершовым был разработан общий проект транслятора и выделены проблемы, которые надо было решить. В первой половине 1961 года такой проект был готов.

Затем надо было сформировать коллектив разработчиков. Тут учитывалось и желание, и возможности. Не все из отдела хотели браться за эту неподъемную работу, не все реально могли ее выполнять. У отдела были и другие задачи, нельзя было их оголять. В результате сформировался совершенно молодежный и практически не имеющий опыта (только трое имели дело с трансляцией до этого) коллектив — это были или сравнительно недавние выпускники университетов или только что кончившие университет: Ершов, Змиевская, Трохан (Московский университет), Бабецкий, Кожухин, Поттосин (Томский университет), Кожухина, Мишкович (Ленинградский университет), Волошин (Рижский университет), Бежанова (Горьковский университет), Загацкий (Саратовский университет), Михалевич (курсы программистов СО АН СССР, потом Новосибирский университет). Компания разноплеменная и не очень искушенная в программировании, особенно для такого сложного, пионерского проекта (средний стаж в программировании в среднем был три с половиной года: от 8 лет до года, большинство — 2-3 года). Все искупало два фактора: хорошая организация работ, которую проводил Андрей Петрович, и общее желание сделать "большое дело".

Все создание Альфа-транслятора было разбито на следующие этапы.

- Создание проектного задания. Это было сделано Ершовым при участии Кожухина.
- Продумывание алгоритмов трансляции и оптимизации. В этом участвовали практически все разработчики. За каждым был закреплен какой-либо "участок", в котором он должен найти решение, обсуждавшееся затем сообща. В результате была выработана последовательность действий (блоков транслятора) и представление об общих структурах данных, которые связывали блоки. Блок обычно поручался тому, кто продумывал соответствующие алгоритмы.
- Разработка блоков. Алгоритм блока специфицировался на специальном неформальном, но достаточно точном языке.
- "Ручная" прокрутка блоков. Это было специальное действо. Разработчик блока и кто-либо прикрепленный (из числа других разработчиков) садились рядом и "исполняли" спецификацию на некотором примере. Результат исполнения использовался при прокрутке следующего блока. Прокрутка шла последовательно по блокам транслятора, в соответствии с их следованием.

- Программирование. По разработанной спецификации писалась машинная программа.
- Автономная отладка блоков. Она шла параллельно для всех блоков, использовались примеры прокрутки.
- Комплексная отладка. Здесь возникали проблемы, которые приводили к перепроектированию некоторых блоков, — прежде всего, при обнаружении слишком больших временных затрат.

Общая структура организации была такой — руководитель Ершов, два координатора — Кожухин (1-я часть транслятора) и Поттосин (2-я часть транслятора). Руководитель отвечал за принятие всех основных решений. Предполагалось, что он свободен от собственно разработки и принятия технических решений, но практически он участвовал в разработке блоков экономии памяти, а кроме того, когда выяснилось, что нужен специальный блок превращения промежуточного представления программы во внутренний язык, а все разработчики уже "разобраны" по блокам, то Ершову пришлось самому писать этот блок. Координаторы отвечали за проблемы, возникавшие у разработчиков блоков и следили за согласованием работ. Особую роль играл Волошин. Он зачастую в ущерб своей работе (в результате его блоки задержались) участвовал в обсуждении многих возникавших вопросов и часто помогал неожиданным взглядом на проблему. Польза такого "блуждающего дискутанта" в проекте несомненна.

Состав Альфа-транслятора и персоналия

Общая структура транслятора состоит из четырех этапов: получение промежуточного представления, превращение его во внутренний язык и оптимизации на внутреннем языке, построение машинных команд, экономия памяти и формирование машинной программы. Все эти этапы разбивались на блоки. Разбиение определялось как функциональными действиями, так и необходимостью "вписаться" в малую (4K слов) память компьютера.

Окончательная структура блоков такова:

- Ввод и первичный синтаксический контроль (блок 1), разработчик и реализатор — Загацкий.
- Обработка описаний (блок 2) — Кожухин.
- Завершение идентификации (блок 3) — Бабецкий.
- Декомпозиция выражений (блоки 4 и 5), разработчик — Кожухин, реализатор — Кожухина.

- Анализ и программирование процедур (блоки 6 и 7) – Загацкий.
- Перевод из промежуточного представления во внутренний язык (блок 8) — Ершов.
- Обработка геометрических операций над массивами и реализация действий с многомерными массивами (блоки 9, 10) — Волошин.
- Анализ и чистка циклов (блоки 11-13) — Бежанова (при участии в разработке Поттосина).
- Экономия выражений (блок 14) — Поттосин.
- Построение машинных команд (блок 15) — Кожухина (при участии в разработке Бабецкого).
- Программирование индекс-регистров и циклов (блоки 16-18) — разработчик Поттосин, реализатор Кожухина.
- Экономия констант (блок 19) — Змиевская (при участии в разработке Ершова).
- Построение операторной схемы и графа несовместимости (блоки 20, 21) — Мишкович (при участии в разработке Ершова).
- Распределение памяти (блок 22) — Трохан (при участии в разработке Ершова).
- Чистка и компоновка программы (блоки 23, 24) — Трохан, Змиевская.

Доводка и внедрение

Когда Альфа-транслятор начал “дышать”, начали проявляться и не предвиденные ранее проблемы. Насколько помнится, особенно болезненными были три.

Вначале мы не предполагали никакой нейтрализации ошибок: транслятор останавливался на первой найденной (зачем транслировать далее неправильную программу?). Наши пользователи тут же застонали: надо было столько раз повторять трансляцию, сколько ошибок в программе. После этого была введена нейтрализация — пропуск до первого опорного символа (начало, конец, точка с запятой).

Обнаружилось, что при большом числе различных идентификаторов наш лексический анализ “жрет” большую часть времени трансляции, пришлось вводить функцию расстановки для отождествления имен.

Выяснилось, что при возрастании размера программы время на экономию памяти резко возрастает и на достаточно больших программах занимает львиную долю времени трансляции. Пришлось пересмотреть

алгоритм и перекомпоновать структуры данных.

Доводка транслятора шла в "боевом" режиме, на реальных задачах Н. Ершовой, В. Каткова, С. Ривина, Е. Каленковича. Особенно плодотворно (для нас, разработчиков) шла работа с задачами Владика Каткова (он же и был наиболее пострадавшим).

Сначала исправление ошибок шло в рабочем режиме: как только она обнаруживалась, так тут же исправлялась и шла в рабочую версию. Скоро выяснилось неудобство такого режима — якобы исправление ошибки приводило к неработоспособности транслятора. После этого был введен режим версионности: исправление ошибки обкатывалось на потоке тестов и задач, и только потом это исправление вносилось в рабочую версию транслятора. Хранителем версий и официальным (и единственным) исправителем был Юра Михалевич.

При отсутствии алфавитно-цифрового печатающего устройства вся информация об ошибках печаталась во внутренних цифровых кодировках транслятора. Возникла необходимость в посредничестве между транслятором и пользователями — перевода кодировок на "человеческий" язык и информировании пользователей о месте и характере ошибок. Такое посредничество осуществляла Альфа-группа, в которой главные роли играли Володя Минаев, Лиля Корнева, Маша Легостаева.

Отлаживать Альфа-программы было трудно — большая оптимизация не давала возможности хорошо установить соответствие между машинной программой и исходным тестом. Нужен был специальный инструмент. Такой инструмент — Альфа-отладчик — был создан Михалевичем. На основании указаний к отладке (какую историю исполнения надо выдать) он делал нужные вставки в терминах Альфа-языка в исходный текст. Отредактированный таким образом текст потом и транслировался.

Конечно, Альфа-система с точки зрения современных требований к программному продукту была несовершенной: тяжела в использовании, с плохим пользовательским интерфейсом, требовала специальных знатоков-посредников и т.п. Однако благодаря тому, что она давала код весьма близкий по эффективности к ручному программированию, Альфа-система использовалась везде в тех организациях, где нужна была высокая эффективность программ. Это были сотни организаций по всей стране. Наша Альфа-группа ведала рассылкой новых версий всем пользователям. В некоторых организациях (правда, немногочисленных) Альфа-систему даже модифицировали для своих нужд.