МЕТА-МЕТА-ВЫЧИСЛЕНИЯ И СПЕЦИАЛИЗАЦИЯ ПРОГРАММ

С.А.Романенко

Пусть $Spec$ – функция от двух переменных, заданная программой и удовлетворяющая соотношению $F(s,d) = Spec(F,s)(d)$, где F – произвольная функция от двух переменных, заданная программой, а S и d – входные данные для F. Такую функцию $Spec$ мы будем в дальнейшем именовать специализатором. То, что специализаторы могут стать практически важным средством программирования, было осознано еще в шестидесятые годы [5]. К началу семидесятых годов относится, видимо, появление самого термина "специализатор" [2,10].

В 1971 году И.Футамура обнаружил, что компиляция программ может осуществляться вычислением Spec (Int, Prog), т.е. специализацией интерпретатора по отношению к программе, и что компиляторы могут быть получены вычислением Spec (Spec ,Int), т.е. специализацией специализатора по отношению к интерпретатору [9].

В дальнейшем [8] было обнаружено, что вычислением Spec (Spec , Spec) можно получить компилятор компиляторов, преобразующий интерпретаторы в компиляторы.

В 1985 году группе, работающей под руководством профессора копенгагенского университета Н.Джонса, удалось создать практически работающий самоприменимый специализатор, благодаря чему впервые удалось осуществить эксперименты по автоматическому преобразованию интерпретаторов в компиляторы по методу И.Мутамуры и даже вычислить нетривиальный генератор компиляторов Spec(Spec , Spec) [3,4,7].

Особенность копенгагенского специализатора в том, что в нем используются не только мета-вычисления (как в предшествующих специализаторах), но и мета-мета-вычисления. При этом специализатор имеет дело с тремя уровнями вычислений. Базовый Мета-уровень: мета-данные (термы семантического языка), мета-вычисления над мета-данными (частичные вычисления), мета-интерпретация. Мета-мета-уровень: мета-мета-вычисления над мета-мета-данными, мета-мета-интерпретация.

В то время как мета-вычисления изображают "в общем виде" обычные вычисления (примерно так, как это делается в элементарной алгебре), мета-мета вычисления изображают "в общем виде" мета вычисления.

Получив, благодаря любезности Н.Джонса, подробную информацию о структуре и принципах работы копенгагенского специализатора, автор попытался применить эти принципы к программам, написанным на языке Рефал (а не языке Лисп). В результате был разработан специализатор RLS, входным языком которого является RL - функциональный язык, который обеспечивает те же структуры данных, что и Рефал, но при этом является языком более низкого уровня, будучи похож на Лисп формой записи программ.

RLS разделен на две основные части: мета-специализатор и мета-интерпретатор [6].

Мета-специализатор получает на входе RL-программу и последовательность литер "S" и "D", описывающую, какие входные па-

раметры программы являются статическими (S), т.е. их значения будут известны во время мета-интерпретации, а какие - динамическими (D), т.е. их значения могут быть неизвестны во время мета-интерпретации. Во время мета-мета-интерпретации все переменные в программе и все операции над данными классифицируются как статические или динамические. Затем эта информация заносится в программу - программа аннотируется.

Мета-интерпретатор получает на входе аннотированную RL-программу и значения S-параметров. Результат работы - остаточная программа, у которой элиминированы входные S-параметры.

Затем остаточная программа попадает в пост-процессор, который выполняет переименование функций и переменных, а после этого - в оптимизатор, одна из задач которого - расщепление параметров функций, чтобы избежать построения промежуточных структур данных при передаче информации от одних функций к другим.

Информационные связи между различными частями специализатора можно изобразить следующим образом:

$$Spec(Prog, Pars\text{-}Cl, S\text{-}Vals) =$$

$$Opt(Post(M\text{-}Int(M\text{-}Spec(Prog, Pars\text{-}Cl), S\text{-}Vals)))$$

Отсюда можно видеть, что если мы захотим подвергнуть специализации сам специализатор, объявив $Prog$ и $Pars\text{-}Cl$ статическими параметрами, а $S\text{-}Vals$ - динамическим параметром, оказывается, что все параметры мета-специализатора - статические, поэтому обращение к нему можно просто вычислить, а все параметры пост-процессора и оптимизатора - динамические, поэтому нет возможности их существенно специализировать. Таким образом, единственная часть специализатора, которую можно (и нужно) подвергнуть нетривиальной специализации - это мета-интерпретатор. Это существенно облегчает задачу порождения генератора компиляторов, ибо вместо $Spec(Spec, Spec)$ можно вычислять $Spec(M\text{-}Int, M\text{-}Int)$.

Литература

1. **Beckman L., Haraldson A., Oskarsson O., Sandewall E. A partial evaluator, and its use as a progming tool. - Artificial Intelligence, Vol. 7, No4, 1976, pp.319-357.**

2. **Dixon J. The specializer, a method of automatically writing computer programs.- Division of Computer Research and Technology, National Institute of Health, Bethenda, Maryland, 1971.**

3. Jones N.D., Sestoft P., Sondergaard H. An experiment in partial evaluation: The generation of a compiler generator. - SIGPLAN Notices, Vol.20, No.8, 1985, pp.82-87.

4. Jones N.D., Sestoft P., Sondergaard H. An experiment in partial evaluation: The generation of a compiler generator.- In Proc. 1st Intl. Conf. on Rewriting Techniques and Application, Dijon, France, 1985. Springer LNCS 202(1985),pp.124-140

5. Lombardi L.A. Incremental computation. - Advances in Computers, 8, Academic Press, New York, 1967.

6. Романенко С.А. Генератор компиляторов, порожденный самоприменением специализатора, может иметь ясную и естественную структуру. Препринт ИПМ им.М.В.Келдыша АН СССР № 26, М.: 1987.

7. Sestoft P. The structure of a self-applicable partial evaluator. - In: Ganzinger H. and Jones N.D.(Eds). Programs as Data Objects,Copenhagen, Denmark, 1985. Springer LNCS 217 (1986) pp. 236-256.

8. Базисный рефал и его реализация на вычислительных машинах. М.: ЦНИПИАСС, 1977, с.92-95.

9. Futtamura Y. Partial evaluation of computation process- an approach to a compiler compiler.- Systems, Computers, Controls, Vol.2, No.5, 1971, pp.45-50.

10. Chang Ch. Lee R. Symbolic logic and mechanical theorem proving. - Academic Press, 1973. (Русский перевод: Ч.Чень, Р.Ли. Математическая логика и автоматическое доказательство теорем: Пер. с англ. /Под ред. С.Ю.Маслова. - М.: 1983. - 360 с.).

СЕМИОТИЧЕСКИЕ АСПЕКТЫ ФОРМАЛИЗАЦИИ ИНТЕЛЛЕКТУАЛЬНОЙ ДЕЯТЕЛЬНОСТИ

Всесоюзная школа-семинар
«Боржоми-88»

ТЕЗИСЫ ДОКЛАДОВ И СООБЩЕНИЙ

МОСКВА 1988

Всесоюзный институт научной и технической информации
ГКНТ СССР и АН СССР
Институт кибернетики АН Грузинской ССР
Вычислительный центр АН СССР
Научный совет по проблеме "Искусственный интеллект" Отделения
информатики, вычислительной техники и автоматизации АН СССР

СЕМИОТИЧЕСКИЕ АСПЕКТЫ ФОРМАЛИЗАЦИИ ИНТЕЛЛЕКТУАЛЬНОЙ ДЕЯТЕЛЬНОСТИ

Всесоюзная школа-семинар
г.Боржоми, 22-30 апреля 1988 г.

ТЕЗИСЫ ДОКЛАДОВ И СООБЩЕНИЙ

Москва 1988

Председатели Оргкомитета чл.-корр. АН ГССР Г.Л. Харатишвили
проф. П.В. Нестеров

Председатель Программного комитета академик Г.С. Поспелов

Руководители секций
Проф. Д.А. Поспелов, докт. филол.наук Вяч.В.Иванов

Составитель
к.т.н. В.К. Финн

Редакторы
к.ф.-м.н. М.М. Забежайло, Е.В. Рахилина,
В.В. Рыжиков, к.ф.-м.н. Д.П. Скворцов,
Е.Ф. Фабрикантова, М.В. Филиппенко

Ответственный редактор
доц. Р.С. Гиляревский

С Е К Ц И Я I

ЛОГИЧЕСКИЕ ПРОБЛЕМЫ И ПРОГРАММНЫЕ СРЕДСТВА ПОСТРОЕНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Руководитель

Д.А. П о с п е л о в

Ученый секретарь

Е.Ф. Ф а б р и к а н т о в а

Сдано в набор 15.03.88
Подписано в печать 26.02.88 Т-02869
Формат 60x90 1/16 Печать офсетная Бум.офс.
Усл.печ.л. 26,5 Усл.кр.-отт. 26,62 Уч.-изд.л. 22,81
Тир. 250 экз. Зак. 2226 Цена 1р.55к.

Производственно-издательский комбинат ВИНИТИ
140010, Люберцы 10, Московской обл.,
Октябрьский проспект, 403