ИНФОРМАЦИОННЫЙ АНАЛИЗ

УДК 81'93:[510.652:510.57]

И. Х. Шмаин (Франция)

ГАРФ — ЯЗЫК РЕКУРСИВНЫХ ФУНКЦИЙ ОБОБЩЕННОЙ АРИФМЕТИКИ: ФОРМАЛЬНОЕ ОПИСАНИЕ СЕМАНТИКИ

Дается формальное описание семантики языка GARF (его синтаксис был описан в предыдущей работе) средствами самого этого языка, с использованием в этом случае некоторых неконструктивных (невычислимых) функций и операторов (также определенных в предыдущей работе, см. НТИ, № 3). Для каждой синтаксической конструкции языка определяется ее «модель» в «аппликативной алгебре», т. е. в универсальной семантической области без типов с единственной операцией «применения», рассматриваемой в самом широком смысле.

ВВЕДЕНИЕ

Семантика языка GARF состоит в задании для каждого его синтаксического класса некоторой функции («семантической функции»), связывающей объекты этого класса с элементами некоторой (заранее фиксированной) алгебраической структуры («аппликативной алгебры»), построенной вокруг понятия «применения» («аппликации»). Эта связь не является просто сопоставлением некоторой синтаксической конструкции ее «значения» в аппликативной алгебре; на самом деле ситуация несколько сложнее. Например, значение некоторого выражения определено только при данных значениях всех входящих в него примитивных («распределение значений на примитивных»), а примитивные мы рассматриваем в языке GARF только в связи с некоторой их спецификацией. Мы приходим, таким образом, к понятию «смысла» данного выражения A как некоторой функции, сопоставляющей каждой тройке {S, PHI, L} (где S — спецификация, L — аппликативная алгебра и PHI — распределение значений из L на примитивных, специфицированных в S) некоторый элемент KSI из L. Иными словами, «смысл» выражения A заключается в том, что при всяком «разумном» («согласованном», «правильном») задании L, S и PHI это выражение определяет некоторое конкретное значение KSI из L.

Аналогичным образом определяется семантика уравнения. Эти два синтаксических класса стоят особняком; во всех остальных случаях (а именно, в определении семантики спецификации, системы уравнений, определения класса, сегмента, GARF-комплекта) центральным является понятие *модели*. Схема задания семантики во всех этих случаях такова: данный синтаксический элемент *W* считается «осмысленным» (при данных L, S и PHI или при данных L и PHI в случае семантики спецификации), если у него есть хотя бы одна модель. Моделью называется такое придание значений всем примитивным, «определяемым» («задаваемым», «описываемым») в данном синтаксическом объекте W, что все выраженные этим объектом условия выполняются. Первоначальное распределение значений рассматривается при этом только на *примитивных, «внешних» по отношению* к W, т. е. в нем не определяемых. Во всех перечисленных только что случаях, кроме случая спецификации, определяется затем понятие «главной» (или «минимальной») модели. Такая модель (при данных, как всегда, L, S, и PHI) обладает свойствами «единственности с точностью до изоморфизма» и поэтому может считаться «значением» объекта W.

КЛАСС АППЛИКАТИВНЫХ АЛГЕБР

```
|||CLASS: AALG;
  Sign: Aalg;
  Bas: Set, Apply, Nonsense, True, False;
  Const:
    {BOOL: true, false, ⊃, &},
    {NAT: 0, sc, > =, 1},
    {VECT(X): ir, ·, rt, rp, lp, lt, length, j1, j2, vun};
Arg: {U : : U},
    {U: ro(j, v), non, t, c, j},
    {VECT(U): v},
    {NAT: i};
  Obj: {AALG: Aalg(U, ro, non, t, c)};
  Restr:
    non< >t, non< >c, t< >c,
    ∀(v)|((ro(non, v) = non)&(ro(t, v)=non)&(ro(c,
v)=non))),
    ∀((v|el(non, v)⊃∀(j|(ro(j, v)=non)));
||Set(Aalg(U, ro, non, t, c))≅U||;
||Apply(Aalg(U, ro, non, t, c))≅ro||;
||Nonsense(Aalg(U, ro, non, t, c))≅non||;
||True(Aalg(U, ro, non t, c))≅t||;
||False(Aalg(U, ro, non, t, c))≅c||;
END|||.
```

Определение класса аппликативных алгебр является аксиоматическим описанием всех аппликативных алгебр. Аппликативная алгебра L=Aalg(U, ro, non, t, c) задается неким множеством U=Set(L), двуместной функцией ro=Apply(L) (называемой *аппликацией* или *оператором применения*), сопоставляющей каждому элементу j из U и каждому вектору v элементов из U некий новый элемент из U (называемый «результатом применения j к v»), и тремя различными выделенными элементами множества U: «бессмыслицей» non=Nonsense(L), «истиной» t=True(L) и «ложью» c=False(L). Любой элемент j из U может быть применен к любому набору v таких элементов. Не всякое применение считается, однако, осмысленным; а именно, иногда результат применения некоторого j к некоторому вектору v оказы-

2*

вается равным поп. В частности, каждый из трех выделенных элементов U, т. е. поп, t и с, будучи примененным к любому вектору v, дает бессмыслицу.

Аппликативная алгебра, таким образом, представляет собой некоторое универсальное множество функций, *не распределенное* ни по «классам» (или «типам»), ни по «уровням» применения, ни даже по арностям. Конкретные аппликативные алгебры будут, как правило, иметь все эти дополнительные структуры; например, можно определить «предметы» (иначе, нульарные функции нулевого уровня) как такие элементы j из U, результат применения которых к любому вектору v дает поп; далее элемент j из U определяется как «n-арная функция 1-го уровня», если его применение к некоторым векторам v, имеющим длину n и состоящим только из «предметов», дает тоже некоторый «предмет», а его применение ко всякому вектору любой другой длины или содержащему хотя бы один «не-предмет» дает всегда поп. Модели классов и систем уравнений в ГАРФе никогда не содержат «неопределенно-арных» функций; мы не включили, однако, соответствующих ограничений в определение аппликативной алгебры, остающееся максимально схематичным, так как вся необходимая точность нашей семантики достигается не уточнением семантической области, а уточнением отображения в нее наших синтаксических объектов.

ОБЩАЯ СПЕЦИФИКАЦИЯ

Семантика языка GARF описывается в одном большом сегменте, состоящем из нескольких «разделов» с общей спецификацией в начале этого сегмента. Каждый «раздел» посвящен семантике одного синтаксического класса ГАРФа. В спецификации этого сегмента мы для краткости опустим список констант; все они взяты из соответствующих сегментов описания синтаксиса (см. предыдущую работу).

```
|||SEGM: SEMANTICS;
  Def:  Σexpr, Σspec, modspec, imod, satisf, Modif,
        Instantiate, Σequat,
        Σsystem, Modsystem, narr, modsystem,
        Σclad, Modclad, hom, modclad,
        Σsegm, Modsegm, msegm1, msegm2, modsegm,
        ΣGARF, ModGARF, modGARF, valconst;
Aux: f1, f2, f3, f4, f5, f6, f7, f8;
  Const: ;
  Arg·  {PI: : PI},
        {PI: x, y},
        {VECT(PI): b},
        {SPEC(PI): S},
        {EXPR(S): A, C, r),
        CALL(S): q},
        {EQUAT(S): E},
        {SYSTEM(S): Y},
        {CLAD(S): D},
        {SEGM(PI): G},
        {GARF(PI): H},
        {AALG: L},
        {VECT(Set(L)): v},
         {Set(L): KSI, HI(KSI), PHI(x), ALFA(x),
              delta(x), delta1(x), TETA(x), ETA(x),
               DELTA(y)(x), DELTA1(y)(x)}.
```

Нуждаются в комментариях только несколько последних специфицированных здесь аргументов.

L — это аппликативная алгебра, выбираемая в качестве семантической области. Она может быть выбрана произвольно, но при «неудачно» сделанном выборе семантика получится очень бедная.

KSI — произвольный элемент из этой алгебры, а точнее из ее несущего множества Set(L); OMEGA — вектор таких элементов; HI — отображение Set(L) в себя (используется в определении гомоморфизма моделей класса).

PHI, ALPHA, delta, delta1, ETA, TETA — различные *распределения,* иначе — придания значений из Set(L) примитивным из PI; они на самом деле рассматриваются только на примитивных, специфицированных в S; при этом ALPHA используется, когда рассматривается придание значений аргументным, delta — когда рассматривается придание значений примитивным, определяемым в данном синтаксическом объекте, delta1 — при сравнении двух моделей, PHI — как совокупность «задаваемых значений» (для констант и «внешних определяемых»).

DELTA, DELTA1 и GAMMA — тоже распределения, но более сложного вида. А именно, DELTA(v)(x) дает значение (в L) примитивной x в сегменте с именем v. Таким образом разрешается проблема, связанная с независимым использованием одной и той же примитивной в разных сегментах.

СМЫСЛ ВЫРАЖЕНИЯ

Для всякого выражения A в спецификации S его смысл Σexpr(A) есть функция от аргументов S, PHI, L со значениями в L. Она определяется рекурсией по построению выражения A с разбором случаев по самой внешней сигнатурной:

```
||Σexpr(A)(S, PHI, L) : : = {

simple(A)→PHI(prim(A));
```

(1) если A — простое выражение, то значение A (точнее, значение его смысла на аргументах S, PHI, L) равно значению, присваемому распределением PHI примитивной, из которой состоит A;

```
appl(A)→Apply(L)(Σexpr(functor(A)) (S, PHI, L),
         Vappl((varg((A), C|Σexpr(C)(S, PHI, L)));
```

(2) если A — выражение применения, то для вычисления его значения надо построить вектор OMEGA из значений всех аргументов A (это будет вектор элементов из L), и «применить» к нему значение KSI функтора выражения A; это «применение» задается, конечно, операцией Apply(L) применения в L;

```
eq(A)→Σexpr(lpt(A)) (S, PHI, L)=Σexpr(rpt(A)) (S,
                                             PHI, L);
```

(3) если A — равенство, то значение A равно истине (в L), если значения левой и правой частей равны друг другу (в L), и лжи (в L) — в противном случае;

```
lambda(A)→f1(kernel(A), vbound(A), S, PHI, L);
```

(4) если A — λ-выражение, то его значение определяется вспомогательной функцией f1, приводимой ниже;

```
width(valt(A))=1→Σexpr(lp(valt(A)))(S, PHI, L);
Σexpr(lp(valt(A))) (S, PHI, L)=True(L)→Σexpr(El(1,
                               valt(A)))(S, PHI, L);
Σexpr(lp(valt(A)))(S, PHI, L)=False(L)→
                  Σexpr(lt(ll(valt(A))))(S, PHI, L);
#→Nonsense(L)};
```

(5) во всех остальных случаях (т.е. когда A есть альтернатива) значение A есть значение первого такого исхода (см. комментарии к синтаксису альтернативы), для которого значение соответствующего условия равно истине (в L); если ни одного такого условия не оказалось (т. е. значения всех условий равны лжи), то значение есть значение безусловного исхода альтернативы.

```
f1(A, b, S, PHI, L)(w): : = {
width(w)<>width(b)→Nonsense(L);
#→Σexpr(A)(S, pref(PHI, x|El(ind(x, b), w), x|el(x,
                                            b)), L)};
||;
```

[f1] определяет значение λ-выражения A (как всегда, при данных S, PHI и L) как такой элемент KSI из L, что применение KSI (в L) к некоторому вектору OMEGA элементов из L «неопределено» (т. е. равно Nonsene(L)), если длина вектора OMEGA не равна длине вектора связанных переменных b в A; в противном случае значение A равно значению его ядра kernel(A) при распределении, получаемом из распределения PHI приданием каждой примитивной из вектора b соответствующего значения из вектора OMEGA.

СМЫСЛ СПЕЦИФИКАЦИИ

Для всякой спецификации S ее смысл Σspec(S) есть предикат от двух аргументов: распределения PHI (рассматриваемого только на константах в S) и аппликативной аглебры L. При данной алгебре L он обращается в истину на всяком таком распределении PHI на константах, которое может быть «достроено» до некоторой модели TETA спецификации S.

Σspec(S)(PHI, L) : : $= \exists$(TETA|modspec(L, S, pref(TETA, PHI, x|const(x, S))))||;

С самого начала модель модифицируется распределением на константах; иными словами, рассматриваются только модели «при данных значениях констант из S».

||modspec(L, S, TETA): : $= \forall$(ALPHA|f1(L, S, Modif(TETA, ALPHA, S, L));

[modspec]: распределение TETA есть модель спецификации S, если для любого распределения ALPHA на аргументных из S результат *конкретизации* (см. ниже функцию «Modif») распределения TETA распределением ALPHA...

f1(L, S, delta): : =f2(L, S, delta)$\supset$imod(L, S, delta);

[f1]: ... есть распределение delta, удовлетворяющее следующему условию:

f2(L, S, delta): : $= \vee\forall$(sp(S), x|(arg(x, S)$\supset$satisf(S, x, delta, L)))
||;

[f2]: если delta согласовано со спецификацией S (см. ниже функцию satisf) на всех аргументных спецификации S,

||imod(L, S, delta): : $= \vee\forall$(sp(S), x|($\neg$const(x, S)$\supset$ $\supset$satisf(S, x, delta, L)))
||;

[imod]: то оно согласовано со спецификацией S на всех определяемых и вспомогательных спецификации S.

||Modif(TETA ALPHA, S, L)(x): : =Instantiate(x, ALPHA, TETA, S, L)||;

[Modif]: Конкретизация распределения TETA распределением ALPHA в S — это такое распределение, которое каждому x из PI придает значение следующим образом:

||Instantiate(x, ALPHA, TETA, S, L): : ={
$\neg$spec(x, S)$\rightarrow$TETA(x);
const(x, S)$\rightarrow$TETA(x);

если x не специфицировано в S или специфицировано как константа, то значение задается распределением TETA;

arg(x, S)$\rightarrow$ALPHA(x);

если x специфицировано как аргументная, то распределением ALPHA;

vgpar(x, S)= $\rightarrow$TETA(x);

если x определяемая или вспомогательная, причем ее спецификация не зависит от параметров (даже «обобщенных параметров»; это выражается условием «вектор (обобщенных) параметров пуст» — см. функцию vgpar в описании синтаксиса ГАРФа), то значение задается распределением TETA;

#$\rightarrow$(Apply(L)(TETA(x), Vappl(vgpar(x, S), ALPHA))}
||;

наконец, если x — определяемая или вспомогательная, причем его спецификация зависит от набора обобщенных параметров $a_1, \ldots, a_n$, перечисленных именно в том порядке, в котором они появляются в векторе спецификации S, то для получения значения на таком x надо сначала применить ALPHA ко всем параметрам $a_1, \ldots, a_n$, затем применить TETA к самому x, и, наконец, применить (в L, т. е. операцией Apply(L)) полученное значение TETA(x) к вектору значений ALPHA(a_1), ..., ALPHA(a_n).

В этом определении содержится один из ключевых моментов семантики языка GARE. Поясним его на примере. Пусть определяемая f специфицирована в S следующим образом:

Def: {A(Z) : f(u, v)};

причем

Arg: {U(M) : u}, {V(K) : v}.

Таким образом, хотя формально здесь специфицирована функция f от двух аргументов u и v, на самом деле мы имеем спецификацию целого семейства F таких функций — семейства, зависящего от трех параметров Z, M, K (по меньшей мере от этих трех параметров, если предположить, что мы и сделаем, что спецификация сама не содержит новых параметров). Конкретная функция f из этого семейства получается при конкретных значениях этих параметров. Приведенные выше определения говорят, что модель TETA спецификации S сопоставляет примитивной f значение TETA(f), которое придается, в некотором смысле, не ей самой, а переменной F; этим значением является функция от параметров Z, M, K, которая для конкретных значений этих параметров (задаваемых распределением ALFA) дает уже функцию от u и v, т. е. то, что может быть значением для *f* в спецификации S. Полученное таким образом новое распределение является инстанцированной (или конкретизированной) моделью S, а сам описанный процесс называется *конкретизацией* (или *инстанцией*) TETA с помощью ALPHA в S. Это процесс согласованного выбора в каждом семействе функций по представителю.

||satisf(S, x, delta))::=

[satisf] выражает «локальную согласованность *PHI* с S на x», и к нему так или иначе сводятся все остальные определения. Здесь рассматривается пара ⟨специфицирующее выражение для x в S — называющий терм для x в S⟩, т. е. элементарная спецификация для x, выделенная из S. Предполагается, что распределение PHI согласовано с S на всех примитивных этих двух выражений, кроме самой x, и проверяется эта согласованность на x; а именно, рассматриваются два случая:

{spelem(x, S)$\rightarrow$Apply(L)(Σexpr(range(x, S))(S, delta, L), j1(Σexpr(spcall(x, S))(S delta, L)))=True(L);

(a) спецификация x как элемента, т. е. спецификация вида A: x(y, ..., z). В этом случае (поскольку мы систематически отождествляем всякое множество с его характеристическим предикатом) требуется просто, чтобы применение (в L) значения (в L) «левой части элементарной спецификации для x», т. е. выражения

3—1936

range(x, S), к значению (в L) ее «правой части», т. е. выражения spcall(x, S), давало бы истину (в L). Иначе говоря, надо чтобы значение правой части лежало в множестве, определенном левой частью.

ǂ→∀(KSI | (Apply(L)(Σexpr(spcall(x, S))(S, delta, L)j1(KSI))=True(L))⊃(Apply(L)(Σexpr(range(x, S))(S, delta, L),j1(KSI))=True(L))))}
||;

(б) спецификация как части, т. е. спецификация вида A::x(y,..., z). В этом случае значение правой части должно быть множеством (в L), т. е. одноместным предикатом в L (относительно левой части это верно всегда), и это множество должно быть подмножеством множества, определяемого левой частью, т. е. для всякого элемента KSI из L, если применение к нему значения правой части элементарной спецификации дает истину (в L), то то же должно быть и для левой части элементарной спецификации.

СМЫСЛ УРАВНЕНИЯ

Для всякого уравнения E в спецификации S его смысл Σequat(E) есть предикат от аргументов S, PHI, L (рассматриваемый на всех примитивных, кроме аргументных), определяемый следующим образом:

||Σequat(E)(S, PHI, L)::=∀(ALPHA|f2(E, S, pref(PHI, ALPHA, x|(arg(x, S)&f1(x, E))), L));

[Σequat]: для всякого доопределения PHI некоторым распределением ALPHA на аргументных,

f1(x, E)::=occur(x, lpart(E))∧occur(x, rpart(E));

[f1]: входящих в левую или правую часть этого уравнения,

f2(E, S, PHI, L)::=imod(L, S, PHI)⊃(f3(E, S, PHI, L)&f4(lpart(E), S, PHI, L));

[f2]: если результат этого доопределения является конкретизированной моделью спецификации S, то

f3(E, S, PHI, L)::=Σexpr(lpart(E))(S, PHI, L)==Σexpr(rpart(E))(S, PHI, L);

[f3]: значения левой и правой частей этого уравнения E равны друг другу (в смысле семантики выражений),

f4(A, S, PHI, L)::=def(dfunc(A), S)⊃Σexpr(A)(S, PHI, L)〈〉Nonsense(L)
||;

[f4]: причем, если глубинный функтор левой части есть определяемая, то значение левой части «осмысленно в L».

СМЫСЛ СИСТЕМЫ УРАВНЕНИЙ

Для всякой системы уравнений Y ее смысл Σsystem(Y) определяется как предикат от трех переменных S, PHI, L, принимающий значение истины, когда в L существует такое TETA, которое есть модель системы Y (в L).

||
Σsystem(Y)(S, PHI, L)::=∃(TETA|modsystem(L, Y, S, PHI, TETA))
||;

[Σsystem]: данное как аргумент распределение PHI принимается во внимание только на константах и на «внешних определяемых», т. е. на таких примитивных, которые специфицированы в S как константы или как определяемые, но не определяются системой Y (т. е. не являются глубинными функторами никакого уравнения системы Y). Они суть «извне (для Y) определенные» (это выражено ниже в уравнении modsystem). Напротив, распределение TETA принимается во внимание только на «определяемых системы», т. е. на таких примитивных, которые специфицированы в S как определяемые и притом определяются именно системой Y (т. е. являются глубинными функторами уравнений системы Y). Таким образом, распределение TETA есть «модель в L системы Y при данных (распределением PHI) значениях констант и внешних определяемых».

||modsystem(L, Y, S, PHI, TETA)::=f1(L, Y, S, pref(PHI, TETA, x|(sdef(x, S, Y)));

[modsystem]: модель в L системы Y при данных (распределением PHI) значениях констант и внешних определяемых есть такое распределение TETA (на определяемых системы Y), что:

f1(L, Y, S, PHI)::=∃(ETA|f2(L, Y, S, pref(PHI, ETA, x|(saux(x, S, Y))));

[f1]: в L существует такое распределение ETA (на вспомогательных системы Y), что:

f2(L, Y, S, PHI)::=modspec(L, S, PHI)&∀(ALPHA|f3(L, Y, S, Modif(PHI, ALPHA, S, L)));

[f2]: распределение PHI, модифицированное распределением delta на определяемых системы Y и распределением ETA на вспомогательных системы Y,

(1) есть модель в L спецификации S,

(2) для всякого распределения ALPHA на аргументных спецификации S, если результат конкретизации PHI посредством ALPHA...

f3(L, Y, S, delta)::=imod(L, S, delta)⊃f4(L, Y, S, delta));

[f3]:... есть конкретизированная модель спецификации S,...

f4(L, Y, S)::=∀∀(vequat(Y), E|Σequat(E)(S, PHI, L))
||;

[f4]:... то PHI обращает все уравнения системы Y в истину (в смысле семантики уравнений).

МИНИМАЛЬНАЯ МОДЕЛЬ СИСТЕМЫ УРАВНЕНИЙ

Начиная с этого пункта семантики, мы будем определять для каждой синтаксической конструкции языка, кроме ее смысла и понятия ее модели, еще и понятие минимальной модели. Собственно, именно минимальные модели нас интересуют прежде всего; именно они являются значениями наших конструкций; но конструкция может считаться осмысленной уже в том случае, если она имеет «хотя бы какую-нибудь» модель. Итак:

||Modsystem(L, Y, S, PHI, TETA)::=modsystem(L, Y, S, PHI, TETA)&∀(TETA1|modsystem(L, Y, S, PHI, TETA1)⊃f1(L, Y, S, PHI, TETA, TETA1)));

[Modsystem]: минимальная модель TETA системы Y (при данных L, S и распределении PHI на константах) — это такая модель этой системы, что для любой другой модели TETA1 системы Y...

f1(L, Y, S, PHI, TETA, TETA1)::=
f2(L, Y, S, pref(PHI, TETA, x|sdef(x, S, Y)),
pref(PHI, TETA1, x|sdef(x, S, Y)));

[f1]: (сначала выполняется модификация обеих моделей как в уравнении для modsystem; заметим, что мы здесь для простоты включили в каждую модель и значения для вспомогательных системы Y)

f2(L, Y, S, PHI, TETA, TETA1) : : =∀(ALPHA|f3(L, Y, S, Modif(TETA, ALPHA, S, L), Modif(TETA1, ALPHA, S, L));

[f2]: ... что при всяком придании значений ALPHA аргументным из S (которыми мы интересуемся здесь как параметрами спецификации), если оба результата конкретизации (соответственно, delta и delta1)...

f3(L, Y, S, PHI, delta, delta1): : =(imod(L, S, delta)&imod(L, S, delta1))⊃f4(L, Y, S, delta, delta1));

[f3]:... являются конкретизированными моделями спецификации S, ...

f4(L, Y, S, PHI, delta, delta1) : : =∀∀(sp(S),x|(sdef(x, S Y)⊃narr(L, delta, delta1, x)));

[f4]: ... то значение конкретизированной первой модели (delta) на каждой определяемой системы Y является сужением (см. ниже функцию narr) соответствующего значения конкретизированной второй модели (delta1).

narr(L, delta, delta1, x) : : =∀(OMEGA|((Apply(L)(delta(x), OMEGA)◇Nonsense(L))⊃(Apply(L)(delta(x), OMEGA)=(Apply(L)delta1(x), OMEGA)))
||;

[narr]: Мы говорим, что распределение delta является на данном x *сужением* другого распределения delta1, если delta1(x) как элемент аппликативной алгебры применимо по крайней мере ко всем таким векторам OMEGA из L, к которым применимо delta (x), и результаты применения совпадают всюду на области применимости delta(x).

СМЫСЛ ОПРЕДЕЛЕНИЯ КЛАССА

Для всякого определения класса D в спецификации S его смысл Σclad(D) есть предикат от трех аргументов, S, PHI и L, принимающий значение истины, если у определения класса D имеется в L модель TETA.

||Σclad(D)(S, PHI, L)::=∃(TETA|modclad(L, D, S, PHI, TETA))||;

[Σclad]: распределение PHI в данном контексте принимается в расчет только на константах и на определяемых спецификации S, не являющихся ни сигнатурными, ни «глубинным именем» в определении класса D, т. е. на «внешних» (для D) определяемых». Распределение TETA, наоборот, рассматривается только на сигнатурных и «глубинном имени». Полностью оно называется: «модель (в L) определения класса D в спецификации S при данных (распределением PHI) значениях констант и внешних определяемых».

||modclad(L, D, S, PHI, TETA) : : =f1(L, D, S, pref(PHI, TETA, x|(cdef(x, S)));

[modclad]: модель определения класса — это такое распределение TETA (на сигнатурных и имени класса), ...

f1(L, D, S, PHI) : : =modspec(L, S, PHI)&∀(ALPHA|f2(L, D, S, Modif(PHI, ALPHA, S, L)));

[f1]: ... что оно (после модификации при помощи PHI очевидным образом)

(1) является моделью спецификации S,

(2) для любой своей конкретизации при помощи некоторого ALPHA на аргументных (а также на параметрах), ...

f2(L, D, S, delta) : : =imod(L, S, delta)⊃f3(L, D, S, delta);

[f2]: ... такой, что (а) результат delta этой конкретизации является конкретизированной моделью спецификации S, ...

f3(L, D, S, delta) : : =∀(ALPHA|f4(L, D, S, pref(delta, ALPHA, x|(carg(x, D))));

[f3]: ... и (б) для любого такого придания значений ALPHA аргументам определения класса D,

f4(L, D, S, delta)::=f5(L, D, S, delta)⊃(f6(L, D, S, delta)&f7(L, D, S, delta));

[f4]: ... что модифицированное им распределение delta...

f5(L, D, S, delta)::=imod(L, S, delta)&∀∀(restr(D), r|Σexpr(r(D)(S, delta, L)=True(L)));

[f5]: ... по-прежнему является конкретизированной моделью S и при этом «удовлетворяются все ограничения из D» (т. е. обращаются в истину как выражения), — итак для любого такого придания значений сначала конкретизирующим параметрам, а затем аргументам при выполнении ограничений...

f6(L, D, S, PHI) : : =∀∀(obj(D), q|(Apply(L)(Σexpr(cname(D))(S, delta, L), j1(Σexpr(q)(S, delta, L)))= =True(L)));

[f6]: ... (свойство 1) получаемые значения «принадлежат определяемому классу» (т. е. значение полного имени класса, примененное к каждому значению из графы obj (как к выражению), дает истину в L) и...

f7(L, D, S, delta) : : =∀∀(correl(D), E|Σequat(E)(S, delta L));
||

[f7]: ... (свойство 2) все соотношения класса «выполняются» (т. е. обращаются в истину как равенства).

МИНИМАЛЬНАЯ МОДЕЛЬ ОПРЕДЕЛЕНИЯ КЛАССА

||Modclad(L, D, S, PHI, TETA)::=modclad(L, D, S, PHI, TETA)&∀(TETA1|modclad(L, D, S, PHI, TETA1)⊃f1(L, D, S, PHI, TETA, TETA1)));

[Modclad]: минимальная модель определения класса D — это такая его модель TETA, что для любой другой его модели TETA1...

f1(L, D, S, PHI, TETA, TETA1)::=f2(L, D, S, pref(PHI, TETA, x|def(x, S)), pref(PHI, TETA1, x|def(x, S)));

[f1]: (после обычной модификации обеих моделей заданным распределением PHI)...

f2(L, D, S, PHI, TETA, TETA1)::=∀(ALPHA|f3(L, D, S, Modif(TETA, ALPHA, S, L), Modif(TETA1, ALPHA, S, L)));

[f2]:... и для любой конкретизации обеих моделей при помощи некоторого ALPHA на параметрах, ...

f3(L, D, S, PHI, delta, delta1) : : =(imod(L, S, delta) & imod(L, S, delta1))⊃∃(HI|hom(L, S, HI, delta, delta1)) ||;

[f3]: ... совместимой с S, т. е. дающей конкретизированные модели delta и delta1, существует гомоморфизм HI из конкретизированной версии delta модели TETA в соответствующую версию delta1 модели TETA1.

ГОМОМОРФИЗМ МОДЕЛЕЙ

||hom(L, S, HI, delta, delta1) : : =∀∀(sp(S), x|(def(x, S)⊃(HI(delta(x)=delta1(x))&f1(L, HI, delta(x)))));

[hom]: элемент HI аппликативной алгебры L называется ***гомоморфизмом*** из модели delta в модель delta1, если **он**, а точнее, его присоединенная функция {KSI| Appl(L)(HI, j1(KSI)}, для которой мы пишем просто HI(KSI), переводит значение каждой определяемой (и константы) модели delta в значение HI(KSI) этой определяемой в модели delta1, причем:

f1(L, HI, KSI) : : =∀(OMEGA|(Apply(L)(KSI, OMEGA)<>Nonsense(L))⊃HI(Apply(L)(KSI, OMEGA))=(Apply(L)(HI(KSI),Vappl(OMEGAA, HI)))))
||;

[f1]: на всей области определения элемента KSI действие элемента HI «перестановочно» с действием оператора Apply(L) применения в L, т. е. для любого OMEGA из этой области (допуская формальную неточность, которой нет в самом уравнении),

HI(KSI(OMEGA))=HI(KSI)(HI(OMEGA)).

Замечание. Коротко, наши определения выражают то требование, что модель определения класса должна, при данных «допустимых» значениях его параметров, сопоставить полному имени класса множество всех значений, «порождаемых» всеми термами q при условии выполнения всех ограничений r.

Но нам надо добиться еще, чтобы в нашей модели выполнялись (а) все те и (б) только те соотношения, которые задаются списком соотношений из D. Условие (а) выполняется в силу уравнения [f7] системы уравнений [modclad]; но условие (f) не обязательно выполняется для произвольной модели. Например, у класса NAT имеются такие нестандартные модели: множество Z всех целых чисел; множество Nmod(k) вычетов по модулю k (для произвольного фиксированного k). Для каждой из этих моделей существует гомоморфизм в нее из стандартной модели: вложение N→Z в первом случае, факторизация N→Nmod(k) во втором. Обратного же гомоморфизма Z→N или Nmod(k)→N, так легко видеть, не существует.

Таким образом, условие Modclad выделяет именно модель с минимальным множеством соотношений («максимально свободную модель»). Таких моделей, конечно, может быть много, но нетрудно видеть, что все они изоморфны друг другу в следующем смысле: две модели delta и delta1 назовем *изоморфными*, если существует такая пара гомоморфизмов моделей HI: delta→delta1 и HI1: delta1→delta, что для всякого x, определяемого в S, выполняются равенства

HI1(HI(delta(x)))=delta(x)

и

HI(HI1(delta(x)))=delta1(x).

СМЫСЛ СЕГМЕНТА

Для всякого сегмента G его смысл Σsegm(G) есть предикат от двух аргументов PHI, L (где распределение PHI рассматривается только на константах сегмента G), принимающий значение истины, если у сегмента G имеется (при данном PHI на константах) модель в L,...

||Σsegm(G)(FHI, L): :=∃(TETA|modsegm(L, G, PHI, TETA)||;

||modsegm(L, G, PHI, TETA)::=f1(L, G, sspec(G), pref(PHI, TETA, x | (def(x, sspec(G))), TETA);

[modsegm]... т. е. такое распределение TETA на определяемых сегмента, что, будучи модифицировано с помощью PHI на константах...

f1(L, G, S, PHI, TET) : : =V∀(svclad(G), D| modclad(L, D, S, PHI, TET) & V∀(svsyst(G), Y| modsyst(L, Y, S, PHI, TETA))
||;

[f1]:... оно становится моделью каждого определения класса и каждой системы уравнений, составляющих сегмент (разумеется, в каждом случае рассматривается в качестве модели лишь некоторая часть определяемых, соответствующая данному определению классов или системе уравнений).

Итак, говоря коротко, модель сегмента — это совокупность моделей составляющих его определений классов и систем уравнений, «связанных» общей спецификацией.

МИНИМАЛЬНАЯ МОДЕЛЬ СЕГМЕНТА

||Modsegm(L, G, PHI, TETA)::=modsegm(L, G, PHI, TETA) & f1(L, G, sspec(G), pref(PHI, TETA, x|(def(x, sspec(G))), TETA) & ∀(TETA1|modsegm(L, G, PHI, TETA1)⊃f2(L, G, PHI, TETA, TETA1)));

[Modsegm]: минимальная модель сегмента — это (условие 1) такая его модель modsegm(L, G, PHI, TETA),...

f1(L, G, S, PHI, TETA) : : =V∀(svclad(G), D| Modclad(L, D, S, PHI, TETA)) & V∀(svsyst(G), Y| Modsyst(L, S, PHI, TETA));

[f1]:... что она (условие 2) «составлена» из минимальных моделей для систем уравнений и определений классов, составляющих сегмент,...

[Modsegm]:... и что (условие 3) эта минимальность «равномерна», в том смысле, что для любой другой модели, после преобразований [f2]—[f5], аналогичных описанным в комментариях к семантике определения класса,...

f2(L, G, PHI, TETA, TETA1)::=f3(L, G, pref(PHI, TETA, x|def(x, sspec(G))), pref(PHI, TETA1, x|def(x, sspec(G))));
f3(L, G, TETA, TETA1): :=∀(ALPHA|f4(L, G, sspec(G), Modif(TETA, ALPHA, sspec(G), L), Modif(TETA1, ALPHA, sspec(G), L)));
f4(L, G, S, delta, delta1): :=(imod(L, S, delta)& imod(L, S, delta1))⊃f5(L, G, S, delta, delta1);
f5(L, G, S, delta, delta1)::=∃(h|hom(L, S, h, delta, delta1)) & (mseg1(G, delta, delta1)⊃mseg2(L, G, delta, delta1));
||;

[f5]:...мы получаем существование гомоморфизма HI конкретизированных моделей delta→delta1 и, кроме того, еще одно свойство:

||mseg1(G, delta, delta1) : : =V∀(svclad(G), D|V∀(sp(sspec(G)), x|(cdef(x, D)⊃(delta(x)=delta1(x)))))
||;

[mseg1]: если обе конкретизированные модели совпадают на всех определениях классов,...

||mseg2(L, G, delta, delta1) : : =V∀(svsyst(G, Y|V∀(sp(sspec(G)), x|(sdef(x, sspec(G), Y)⊃narr(L, delta, delta1, x))))
||;

[mseg2]:... то на каждой системе уравнений сегмента первая из них (минимальная) является сужением второй.

Это наше определение, конечно, избыточно; но мы не боимся повторять некоторые условия в составе других, если это увеличивает ясность.

СМЫСЛ GARF-КОМПЛЕКТА

На этом последнем этапе нас встречает дополнительная трудность· константы. Дело в том, что каждая константа какого-нибудь сегмента в другом сегменте того же комплекта является определяемой, причем, возможно, под другим именем. Таким образом, одно и то же имя (примитивная) может быть специфицировано в GARF-комплекте несколько раз — в разных сегментах, и поэтому нельзя манипулировать, как мы это делали до сих пор, одним-единственным, понемногу «уточняемым», распределением. Выход состоит в том, чтобы иметь сразу целый «набор» таких распределений — по одному для каждого сегмента. Точнее, будем рассматривать «метараспределение» — функцию на примитивных (именах сегментов), сопоставляющую каждому имени сегмента некоторое распределение значений на специфицированных в нем примитивных. Смысл ΣGARF(H) данного GARF-комплекта H есть предикат от одного аргумента L; он даст значение истины, если H имеет модель.

||ΣGARF(H)(L) : : =∃(DELTA | modGARF(L, H, DELTA)||;
||modGARF(L, H, DELTA) : : =V∀(vsegm(H), G|modsegm(L, G, valconst(G, DELTA), DELTA,(sname(G))))||;

[modGARF]. Модель GARF-комплекта — это такое метараспределение DELTA, что для каждого составляющего комплект сегмента G с именем, скажем, J, соответствующее распределение DELTA(J), дополненное значениями констант из других сегментов комплекта, дает некоторую модель этого сегмента G.

||valconst(G, DELTA)::=pref(DELTA(sname(G)), x| DELTA(owner(x, sspec(G)))(pname(x)), x|const(x, sspec(G)))||;

[valconst]: значения констант берутся из других сегментов следующим образом: для каждой константы x в G находится (по ее спецификации в sspec(G)) ее собственник (owner), в котором эта константа специфицирована как определяемая под ее «первичным именем» pname(x); таким образом метараспределение DELTA, будучи примененным сначала к имени собственника константы, а затем к ее первичному имени, дает нам ее значение. Так определяется согласование констант между разными сегментами.

МИНИМАЛЬНАЯ МОДЕЛЬ GARF-КОМПЛЕКТА

||ModGARF(L, H, DELTA)::=modGARF(L, H, DELTA)&V∀(vsegm(H), G|Modsegm(L, G, valconst(G, DELTA), DELTA(sname(G))))&∀(DELTA1| modGARF(L, H, DELTA1)⊃f1(L, H, DELTA, DELTA1)));

[ModGARF]. Минимальная модель — это (условие 1) такая модель DELTA, что (условие 2) каждая ее «составляющая» является минимальной моделью соответствующего сегмента, причем (условие 3) эта минимальность «равномерна» в том смысле, что для любой другой модели DELTA1...

f1(L, H, DELTA, DELTA1)::=∀(GAMMA|f2(L, H, GAMMA, DELTA, DELTA1));

[f1]·... и для любого другого метараспределения GAMMA (каждая составляющая которого рассматривается только на аргументных соответствующего сегмента и конкретизирует соответствующие составляющие моделей DELTA, DELTA1 согласно уравнению [f5]) существует гомоморфизм моделей HI, как описано в уравнении [f7]

Уравнения с [f2] по [f6] определяют конкретизацию обеих моделей, которая в данном случае выглядит более громоздко, чем обычно, из-за того, что функции метараспределения двухэтажны.

f2(L, H, HI, DELTA, DELTA1)::=f6(L, H, f3(L, H, HI, DELTA), f3(L, H, HI, DELTA1));
f3(L, H, DELTA, DELTA1)::=f4(0, L, H, DELTA, DELTA1);

[f3]: это уравнение задает начало цикла,...

f4(i, L, H, GAMMA, DELTA) .: ={i>=length(vsegm (H))→DELTA;
#→f4(sc(i), L, H, GAMMA, pref(DELTA, y| f5(L, El(i, H), GAMMA, DELTA), y|(y=sname(El(i, H)))));

[f4]:... который затем проходит по всему GARF-комплекту, всякий раз модифицируя метараспределение локально.

f5(L, G, GAMMA, DELTA)::=Modif(DELTA (sname(G)), GAMMA(sname(G)), sspec (G), L);
f6(L, H, DELTA, DELTA1)::=f7(L, H, DELTA, DELTA1) & f8(L, H, DELTA, DELTA1);
f7(L, H, DELTA, DELTA1) : : =∃(HI|V∀(vsegm(H), G|hom(L, sspec(G), valconst(G, DELTA), valconst(G, DELTA1))));

[f7]: Существует гомоморфизм из каждого такого результата конкретизации для DELTA в соответствующий результат конкретизации для DELTA1, причем выполняется приводимое ниже...

f8(L, H, DELTA, DELTA1) : : =V∀(vsegm(H), G| msegl(L, G, valconst(G, DELTA), valconst(G, DELTA1)))⊃V∀(vsegm(H), G|mseg2(L, G, valconst (G, DELTA), valconst(G, DELTA1)))||;
END |||;

[f8]:... условие минимальности для модели сегмента.

Материал поступил в редакцию 29.04.94.