

**Операции логического
программирования и их
применение для анализа языка**

Операции логики: изоляция, отождествление и обобщение и их алгоритмическая реализация

Аксиома свертывания

Аксиома свертывания, согласно которой каждому предикату можно поставить в соответствие некоторое множество и наоборот.

Символически аксиома свертывания записывается так: $((x \in M) \equiv P(x))$.

Такая запись означает следующее: на основании того, что некоторые объекты x обладают общим свойством P , их можно выделить в отдельное множество M .

Справедливо и обратное: множеству M , которое состоит из некоторых элементов x , может быть поставлено в соответствии некоторое свойство P , которым обладают элементы x .

Операция изолирования свойств и отношений логического объекта x

Если обозначить эту операцию как A_I , то её применение к некоторому объекту x

можно записать как

$$A_I(x) = \begin{cases} P_i(x), & i = \overline{1, n}; \\ Q_j(x, y_j), & j = \overline{1, m}; \\ \emptyset, & \end{cases} \quad (1)$$

где $A_j(x)$ - i -е свойство из набора свойств, которым обладает объект x , $Q_j(x, y_j)$ - j -е отношение из набора отношений, которым обладает объект x , y_j - объект, с которым x находится в Q_j (отношении Q_j).

Применение A_I к некоторому объекту x , в случае когда какие-либо свойства или отношения для него не определены, будет давать пустой результат, что показано в (1) в виде пустого набора, который обозначен символом ' \emptyset '.

Операция отождествления

Если обозначить операцию отождествления как A_E , то ее применение к некоторой последовательности объектов (x_1, \dots, x_n) можно записать как

$$A_E(x_1, \dots, x_n) = M_i; \forall M_i \equiv P_i(x) \cup Q_i(x, y), x \in M_i, i = \overline{1, m}. \quad (2)$$

Определение логического типа

тип *logic* (логический тип):

(declare) type logic as record of

P: array of string;

Q: array of record

```
self: string,  
where self  $\Leftrightarrow$  Q;  
y: ^logic;  
end;  
end.
```

Или как:

```
(declare) type logic: record:  
P: array: string;  
Q: array: record  
self: string,  
where self  $\Leftrightarrow$  Q;  
y: ^logic;  
end;  
end.
```

В объявленной конструкции задается логический тип как запись, содержащая массив свойств (P) и массив отношений (Q).

Алгоритмическая запись A_I для логических объектов (типа *logic*)

```
function Aisol(x: logic): function;  
begin  
for i:=1 to length(x.P) do  
return x.P[i] as function <P[i]>(_: logic);  
for j:=1 to length(x.Q).do  
return x.Q[j] as function <Q[j]>(_,_: logic);  
end.
```

В результате вызова этой функции возвращается набор отдельных предикатов (массив одноместных функций), извлеченных из массива свойств ($P[]$) объекта x , и набор (массив двухместных функций) отдельных отношений, извлеченных из массива отношений ($Q[]$) объекта x :

$\langle P[1] \rangle(_), \langle P[2] \rangle(_), \dots, \langle P[n] \rangle(_);$

$\langle Q[1] \rangle(_, _), \langle Q[2] \rangle(_, _), \dots, \langle Q[m] \rangle(_, _).$

Алгоритмическая запись операции A_E для логических объектов

Операция A_E определяется алгоритмически так:

//операция по $P(x)$

function AequalP(X: array of logic): array of set;

begin

return Group X[] by X[].P[];

end.

//операция отождествления по $Q(x,y)$

function AequalQ(X: array of logic): array of set;

begin

return Group X[] by X[].Q[] ;

end.

Синтагма *Group ... by ...* обозначает группировку.

В результате ее выполнения возвращаются массив множеств отождествления, который представляет собой структуру вида

$\{\{X[i_1], X[k_1], X[l_1], \dots], [P_{a_1}, P_{b_1}, P_{c_1}, \dots]\},$

$\{X[i_2], X[k_2], X[l_2], \dots], [P_{a_2}, P_{b_2}, P_{c_2}, \dots]\},$

...

$\{X[i_n], X[k_n], X[l_n], \dots], [P_{a_n}, P_{b_n}, P_{c_n}, \dots]\},$

и остаток из элементов массива X , которые не имеют с другими элементами массива и между собой общих свойств, т.е. не поддаются отождествлению

$[X[j], X[m], X[n], \dots]$.

Множество отождествления

Set_of_Equalization → '{' '[' X ',' '[' Y '{';

X → *xX* □ *x* ',' *xX* | ']';

Y → *yY* | *y* ',' *yY* | ']'.

Множество отождествления состоит из двух частей, двух массивов. В левой части сгруппированы элементы множества, а в правой части – предикаты множества (свойства или отношения).

Обращение к элементам левой части:

//обращение к 1-му элементу множества

S{[1] |};

//обращение к 1-му предикату множества

$S\{[1]\};$

//выбор произвольного элемента множества

select S;

Операция включения элемента в множество. Возвращает значение *true* если элемент имеет

общий предикат с данными множества и успешно включен, возвращает *false* в противном случае.

include(x, S);

Операция исключения элемента из множества. Возвращает значение *true* если
исключаемый
элемент принадлежит множеству. Возможно исключение элемента как по имени, так
и по

//индексу

exclude(S, x);

Понятия

Понятие – это логический объект, который является отображением других логических объектов и содержит их отличительные признаки.

К таковым относятся свойства и отношения.

В каждом понятии различают содержание и объем.

Содержание понятия – это совокупность (набор) отраженных в нем признаков предметов.

Объем понятия – это множество логических объектов, каждому из которых принадлежат признаки, относящиеся к содержанию понятия, т.е. предикаты.

Таким образом, понятие – это логический объект, состоящий из двух частей.

Понятия являются результатом применения отождествления и алгоритмически их можно записать так:

declare type notion □ as record of

//массив логических объектов, образующих множество

X: array of logic;

//массив предикатов (свойств/отношений), которыми обладают все объекты

P: array of predicate;

end.

Отображаемые в понятии объекты выделяются из состава более обширного множества, чем объем понятия.

Таким образом, можно говорить об иерархии понятий.

По характеру элементов объема понятия делят на несобирательные и собирательные. К несобирательным относятся такие понятия, признаки содержания которых присущи каждому элементу объема.

К собирательным относятся понятия, у которых признаки, составляющие содержание, присущи объему в целом, а не отдельным его элементам.

Например, созвездие “Большая Медведица”, “коллектив нашего учреждения”.

Предельно широкие по своему объему понятия называются категориями.

Категории могут получаться и при применении классификационной составляющей некоторого отношения (или системы свойств) в качестве обобщающей к другим отношениям (или системам свойств). При этом слова, выражающие конкретные понятия принимают дополнительные обобщающие значения или вместо них создаются новые слова, нигде ранее не использованные.

Если по иерархии понятий можно продвигаться вверх (от категорий и понятий более низкого уровня к понятиям более высокого уровня) с помощью отождествления путем добавления новых признаков, то в обратную сторону можно продвигаться с помощью отождествления путем отбрасывания имеющихся.

Именование полученных в результате применения отождествления понятий является нерешенной задачей.

Выводы

Таким образом, алгоритмически был определен следующий инструментарий: тип *logic* (“логический тип данных”), реализующий логический объект с набором свойств и отношений, реализованных как массивы; функция *Aisol*; функция *AequalP*, реализующие отождествление по свойствам, функция *AequalQ*, реализующие отождествления по отношениям; синтагма *Group...by...*, позволяющая группировать с помощью символьного решателя, встроенного в систему, логические объекты по их свойствам или отношениям и получать в результате массив множеств отождествления; тип *set* (“множественный тип”/тип “множество отождествления”), представляющий из себя набор, в левой части которого находится массив логических объектов, являющихся элементами данного множества, а в правой части которого находится массив предикатов (свойств или отношений), являющихся общими для всех элементов данного множества; операция обращения к массиву элементов множества отождествления: $M\{[]\}$; операция обращения к массиву предикатов множества отождествления: $M\{[]\}$; операция выбора произвольного элемента из множества – *select*; операция включения элемента в множество – *include*; операция исключения элемента x из множества – *exclude*; операция исключения элемента из множества по индексу – *exclude*; тип *notion* (“понятийный тип данных”).