MỘT THUẬT TOÁN TÌM NGỮ NGHĨA CỦA MỘT TERM TRONG HỆ THÔNG TIN

Hà Quang Thụy

1. HỆ THÔNG TIN VÀ NGÔN NGỮ CỦA NÓ

1. Hệ thông tin được đề cập ở đây theo quan điểm của Pawlak [1] là bộ bốn:

$$S = (X, A, V, \rho)$$

trong đó: X là tập các đối tượng, X hữu hạn
A là tập các thuộc tính, A hữu hạn
V là tập các giá trị các thuộc tính, V hữu hạn
$\rho : X \times A \to V$

$\forall x \in X, \forall a \in A : \rho(x, a) \in V$

Kí hiệu $V_a = [\rho(x, a) : x \in X]$ và $\rho_x = A \to V\ \rho_x(a) = \rho(x, a)$

Đối tượng ở đây có thể là đối tượng mà tại thời điểm xuất phát làm một công việc nào đó chúng ta quan tâm đến hoặc tại một thời điểm sau đó, lại là một tổ hợp giữa đối tượng vừa có và giá trị một số thuộc tính của đối tượng ấy.

2. Ta có các quan hệ (dưới đây kí hiệu là $\tilde{a}$ và $\tilde{S}$) được xác định như sau:

$$\forall a \in A \text{ thì } : \forall x, y \in X : x \,\tilde{a}\, y <=> \rho(x, a) = \rho(y, a)$$

và $\tilde{S} : \tilde{S} = \bigcap_{a \in A} \tilde{a}$.

$\tilde{S}$ cũng như các $\tilde{a}$ là các quan hệ tương đương. $\tilde{S}$ chia X (tập các đối tượng) thành các lớp tương đương. Ta gọi tập các lớp tương đương đó là E_s:

$$\forall c \in E_s : c = [x \in X : \rho(x, a_i) = v_i,\ v_i \in Va_i ; V_i = 1, .. \| A \|]$$
$$\text{và } c \neq \phi$$

được gọi là các tập cơ bản của S

Một tập con của X hoặc rỗng hoặc là hợp của các tập cơ bản được gọi là mô tả được trong S. Ta kí hiệu:

$$D_s = [\text{tập mô tả được trong S}]$$

Nhận xét rằng, trong một số trường hợp E_s là phân hoạch rời rạc của S song nói chung lực lượng của E_s nhỏ thua lực lượng của X và để thay cho công việc với X ta chỉ làm việc với tập E_s. Để đạt được E_s ta phải xử lý sơ bộ S.

3. Ngôn ngữ (để đặt câu hỏi) trong hệ thông tin được xây dựng từ bảng chữ (Kí hiệu là Ls)

. 0, 1 là các hằng

. $\forall a \in A, \forall v \in V$.

. Các dấu (và) cùng với dấu, hoặc các dấu phép toán logic $\cap$ và $\cup$ cùng với $\sim$(phép lấy ngược : phủ định)

Lúc đó term (hay câu hỏi) trong L_s được định nghĩa như sau :

1) 0 và 1 là các hằng $\in L_s$

2) $\forall a \in A, \forall v \in V_a$: $(a, v) \in L_s$

3) Nếu t và $t' \in L_s$ thì $(t \cup t') \in L_s$

$(t \cap t') \in L_s$ và $\sim(t) \in L_s$

Ngữ nghĩa các term được cho bởi $\sigma : L_s \to D_s$ như sau :

1) $\sigma(0) = \phi$, $\sigma(1) = X$

2) $\sigma((a, v)) = [x \in X : \rho(x, a) = v]$

3) $\sigma((t \cup t')) = \sigma(t) \cup \sigma(t')$

$\sigma((t \cap t')) = \sigma(t) \cap \sigma(t')$

$\sigma(\sim(t)) = X \setminus \sigma(t)$

Trong một số trường hợp không thể xảy ra nhầm lẫn, chúng ta có thể bỏ đi các dấu (và) thừa.

Term dưới dạng $t = (a_1, v_1) \cap (a_2, v_2) \cap \ldots \cap (a_{\|A\|}, X_{\|A\|})$

với $v_i \in V_i$, $a_i \neq a_j$ nếu $i \neq j$

được gọi là term sơ cấp.

Ta kí hiệu L_E [Term sơ cấp]. Ta có sự tương ứng $1 - 1$ giữa E_s với L_E qua σ. (bỏ qua các term sơ cấp có ngữ nghĩa ϕ)

Một term t dưới dạng $t = t_1 \cup t_i \cup \ldots \cup t_K$ ($t_i \in L_E$) được gọi là dưới dạng chuẩn.

Ta có khẳng định $\forall t \in L_s \exists t'$: t' dưới dạng chuẩn,

$$\sigma(t') = \sigma(t)$$

I. THUẬT TOÁN TÌM NGỮ NGHĨA MỘT TERM :

Cho một hệ thông tin $S = X, A, V$,

Bài toán đặt ra : với term t cho trước, hãy tìm ngữ nghĩa của nó.

Theo Pawlak trong [1], với term t đó, ta sử dụng thuật toán tìm term t' dưới dạng chuẩn cùng ngữ nghĩa với nó và lúc đó

$$\sigma(t) = \sigma(t') = \bigcup_{i=1}^{K} \sigma(t_i) \quad \text{với} \quad t' = t_1 \cup t_2 \cup \ldots \cup t_K$$

và $\sigma(t_i) = e_i \in E_s$

hoặc rỗng

Ở đây, chúng ta đưa ra một thuật toán từ term t đưa ra ngữ nghĩa của nó mà không tìm term dưới dạng chuẩn. Trước hết, cũng như [1] ứng với $S = <X, A, V, \rho>$ chúng ta xây dựng E_s. Ngoài ra thuật toán sử dụng một số thao tác phụ nữa đối với hệ S. Để sử dụng sau này, chúng ta sẽ thay thế phép toán 1 ngôi $\sim$ thành phép toán hai ngôi ;

$$\sim(t) = (1 - t) \quad \text{và lúc đó} \quad \sigma((t - t')) = \sigma(t) \setminus \sigma(t')$$

Thuật toán được thiết lập như sau:

1. Từ hệ S đánh số thứ tự các phần tử của E_s từ 1 cho đến $\|E_s\|$

Mặt khác, ∀ cặp $(a,v) \in L_s$ ta tương ứng với một véc tơ bit $T_{a,v}$ với số bit $= \|E_s\|$, trong đó:

$$T_{a,v}(i) = \begin{cases} 1 & \text{nếu } e_i \in \delta((a,v)) \\ 0 & \text{nếu } e_i \notin \delta((a,v)) \end{cases}$$

Khi hệ S đã được cho, E_s và mọi vectơ $T_{a,v}$ là được cho.

2. Đồng với việc phân tích cú pháp cho một câu hỏi t chúng ta phân biệt:

– Mỗi cặp dạng (a, v) là một từ vị và con trỏ tới miền giá trị của từ vị này trỏ tới chính véctơ $T_{a,v}$

– Các dấu phép toán ∪, ∩, — được gọi là phép toán. Thực chất việc tìm ngữ nghĩa cho t tương tự việc tính giá trị của một biểu thức và không có gì khó khăn, chúng ta thu được cách viết Ba lan sau cho t, sau khi phân tích cú pháp nói

3. Đây chính là nội dung cơ bản mà chúng ta trình bày và vì vậy chúng ta sẽ miêu tả chi tiết.

– Ngữ nghĩa của t sẽ tương ứng với vectơ T_t là vectơ bit giống các vectơ $T_{a,v}$. Ngoài các từ vị đã có, chúng ta đưa thêm từ vị $\widetilde{\theta}$ trỏ đến T_t. Có vectơ phụ T_t^F.

– Thuật toán được chia làm 2 thủ tục: NẠP và TÍNH và một bộ đếm BD nhận giá trị 0 và 1. Ngoài ra sử dụng 2 stack: stack chính và stack phụ. Đầu tiên BD = 0 và ta xét dần từng kí tự trong cách viết Ba lan sau. Nạp kí tự đầu tiên vào stack chính rồi đi tới thủ tục NẠP

A) THỦ TỤC NẠP

(1). Xét kí tự tiếp theo.

(2) Trường hợp BD = 0, nếu kí tự đang xét là từ vị thì nạp nó vào stack chính và quay về (1).

Ngược lại (nó là phép toán) thì BD = 1, nạp nó vào stack chính và quay về (1)

Trường hợp BD = 1, nếu kí tự đó là phép toán thì nạp nó vào stack chính và quay về (1).

nếu là từ vị hoặc xâu vào rỗng thì BD = ϕ, coi kí tự đó chưa xét và về thủ tục TÍNH

B) THỦ TỤC TÍNH

1. Lấy liên tiếp ngọn stack chính nạp vào stack phụ cho đến khi ở stack chính chỉ còn từ vị.

2. T_t^F = T ngọn stack chính. Khử ngọn stack chính.

3. Giả sử ngọn stack chính là từ vị θ, ngọn stack phụ là phép toán p, ta thực hiện:

$$T_t^F = T_\theta \, \bar{P} \, T_t^F \quad \text{(chú ý: } \theta \text{ ở đây có thể là } \widehat{\theta})$$

4. Khử ngọn 2 stack. Nếu stack phụ chưa rỗng thì quay lại bước

Ngược lại, nạp $\hat{\theta}$ vào stack chính. $T_t = T_t^F$. (T_t chính là $T_{\hat{\theta}}$).

5. Nếu xâu vào rỗng thì dừng.

Ngược lại về thủ tục NẠP.

Các thao tác $\bar{p}$ như sau

+ Nếu $p = U$ (hoặc $\cup$) thì $\bar{p}$ là thao tác hợp (hoặc nhân) lôgic theo từng bit của 2 vectơ bit.

+ Nếu $p = -$ thì $\bar{p}$ là thao tác đảo mã của vectơ bit ứng với toán hạng sau

Sau khi dừng thuật toán, ý nghĩa (câu trả lời) đối với term t sẽ được tìm ra qua phép hợp:

$$\sigma(t) = \bigcup_{T_t(i)=1} e_i \quad (*)$$

Định lý: $\forall_t \in L_s$, thuật toán đúng đắn tìm ra vectơ bit T_t thỏa mãn điều kiện (*).

Do việc thay thế phép toán $\sim$ bởi phép toán $-$, việc đi tìm ý nghĩa term t tương đương với việc đi tìm giá trị một biểu thức mà được viết theo cách viết Ba lan sau. Quá trình thực hiện thuật toán qua hai thủ tục NẠP và TÍNH thể hiện đúng việc tính giá trị đó song thay cho các thao tác trên các tập hợp, chúng ta làm các thao tác các vectơ bit.

Ví dụ: Để minh họa, chúng ta xét một ví dụ đơn giản.

Từ hồ sơ cá nhân có chứa các thuộc tính: chính trị (có là đảng viên hay không), lương, học vị (có phó tiến sĩ hay không) có giảng dạy hay không:

Họ và tên	Đảng viên	Lương	Học vị	Giảng dạy
Nguyễn văn M.	có	359	có	có
Phạm văn D.	có	359	0	0
Đỗ văn S.	0	310	0	có
Đặng thị H.	0	310	0	có
Phạm văn H.	0	310	0	có
Phạm đăng L.	0	310	0	có

Khi tạo lập hệ thông tin, chúng ta có thể tạo ngay E_s như sau:

Với mỗi hồ sơ, ta xem tập giá trị của nó có trùng với tập giá trị của một bản ghi e nào đó đã có trong E_s hay chưa. Nếu chưa có, ta tạo ra một bản ghi C mới mà các thành phần là giá trị của hồ sơ và thêm một thành phần cuối là con trỏ tới miền danh sách đối tượng (trong trường hợp này danh sách đó chính là đối tượng vừa xét).

$e_i = (v_1, \ldots, v_{\|A\|}$, trỏ i) trỏ i lá địa chỉ một danh sách các đối tượng.

Nếu đã có, chúng ta chỉ cần đưa thêm đối tượng vào cuối danh sách đối tượng ứng với e đó mà thôi. Trong ví dụ của ta, lần lượt các vectơ sau tạo ra E_s

e_1 = (có, 359, có, có, trỏ 1)

e_2 = (có, 359, có, 0, trỏ 2)

e_3 = (0, 310, 0, có, trỏ 3)

Trong đó:

Trỏ 1 trỏ tới danh sách gồm {Nguyễn văn M.}

Trỏ 2 trỏ tới danh sách gồm {Phạm văn D.}

Trỏ 3 trỏ tới danh sách gồm {Đỗ văn S., Đặng thị H., Phạm văn H., Phạm đăng L.} (các miền này không chưa giá trị các thuộc tính). Và lúc đó (khi gọi i thay cho thuộc tính thứ i) ta tạo các vectơ:

$$T_{1,\text{có}} = (1\ 1\ 0)$$
$$T_{1,0} = (0\ 0\ 1)$$
$$T_{2,359} = (1\ 1\ 0)$$
$$T_{2,310} = (0\ 0\ 1)$$
$$T_{3,\text{có}} = (1\ 0\ 0)$$
$$T_{3,0} = (0\ 1\ 1)$$
$$T_{4,\text{có}} = (1\ 0\ 1)$$
$$T_{4,0} = (0\ 1\ 0)$$

Với các câu nói: «Tìm những người đang giảng dạy và hoặc là phó tiến sĩ, hoặc là lương 310».

Ta minh họa câu hỏi đó theo L_s là:

((4, có) $\wedge$ ((3, có) $\vee$ (2, 310))). Sau khi phân tích cú pháp, ta có biểu thức dạng viết Ba lan sau $\theta_1\theta_2\theta_3 \vee \wedge$ cho câu hỏi đó. Ở đây, miền giá trị cho từ vị θ_1 là $T_{4,\text{có}}$, cho θ_2 là $T_{3,\text{có}}$, còn θ_3 là $T_{2,310}$. Quá trình thực hiện thuật toán như sau (kí hiệu [chỉ đáy stack):

+ Đầu tiên stack chính thức θ_1: [θ_1

+ Sau khi vào thủ tục NẠP và ngay trước khi vào thủ tục TÍNH, ở stack chính sẽ là [$\theta_1\theta_2\theta_3 \vee \wedge$

+ Sau bước 1. của thủ tục TÍNH, các stack có nội dung:

[$\theta_1\theta_2\theta_3$ ở stack chính. còn [$\wedge \vee$ ở stack phụ.

+ Còn sau bước 2. TÍNH có [$\theta_1\theta_2$ với [$\wedge \vee$ và

$$T_t^F = T_{2,310}$$

+ Sau 2 lần thực hiện bước 3. TÍNH, chúng ta xác định được T_t^F như giá trị của một biểu thức trên các vectơ bit;

$$T_t^F = ((T_{2,310} \vee T_{3,\text{có}}) \wedge T_{4,\text{có}}) = (((0\ 0\ 1) \vee (1\ 0\ 0)) \wedge (1\ 0\ 1)) = ((1\ 0\ 1) \wedge (1\ 0\ 1) = (1\ 0\ 1)$$

+ Sau đó vì hết kí tự nên $T_t = T_t^F = (101)$

Nhận được vectơ T_t, chúng ta chỉ cần làm thao tác hợp để tìm $\delta(t)$:

$\delta(t) = e_1 \vee e_3$ = {Nguyễn văn M. Đỗ văn S., Đặng thị H., Phạm văn H., Phạm đăng L.}

Khi cài đặt trên các máy với các thao tác bit, thuật toán của chúng ta thực hiện khá thuận tiện về thời gian và bộ nhớ.

IV. Vài điều nhận xét

Mô hình hệ thông tin do Palak đề xướng hoàn toàn có ý nghĩa trong thực tiễn. Một mặt, nó cho phép chúng ta làm giảm thời gian làm việc cũng như dung tích bộ nhớ khi thay thế X bởi E_s. Điều đó rất tiện lợi trong khi làm việc với nhiều hệ thông tin nhất là với các hệ thông tin phân tán ngôn ngữ [1]. Mặt khác mô hình này không phải là tĩnh. Đối với các bài toán bổ xung một record, ta chỉ cần duyệt các vectơ $e \in E_s$ để hoặc là thêm một vectơ mới hoặc là thay đổ một danh sách đối tượng. Với bài toán loại bé, chỉ là giảm bớt một thành phần trong một danh sách hoặc loại bỏ một vectơ của E_s. Còn bài toán thay thế là sự kết hợp hai bài toán trên.

Bởi các nhận xét trên, thuật toán của chúng ta là hoàn toàn có ý nghĩa.

TÀI LIỆU DẪN

[1] Z. Pawlak. Distbuted inferastion Systems. ICS PAS reports N°370, 1979

Ха Куанг Чуй

АЛГОРИТМ ОПРЕДЕЛЕНИЯ СЕМАНТИК ТЕРМА В ИНФОРМАЦИОННОЙ СИСТЕМЕ

Суть метода заключается в том, что при помощи замены операции одно операнта (~) операцией двухоперантов (1—), определение семантики заданного терма в самом деле эквивалентно вычислению значения арифметического выражения. Таким путём в отличие от метода З.ПАВЛАКА /1/, все необходимые операции теперь выполняется на битах а не на множествах и семантика заданного терма определяется быстро и легко.

В статье показаны ходы строения и работы описанного алгоритма с помощью одного примера.

Ha Quang Thuy

THE ALGORITHM TO DETERMINE THE MEANING OF A TERM IN AN INFORMATION SYSTEM

One of main problems in an information system [1] is to determine the meaning of a term. In essence this is the same as the determination of the value of an expression. This algorithm uses bit — actions on bit — vectors which enables the meaning to be found easily and fast.

An example is given in the paper.

Bộ môn Tin học
Trường Đại học Tổng hợp Hà nội

Nhận bài ngày 25-10-1986