BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SỸ KHOA HỌC

NGHIÊN CỨU CÁC PHƯƠNG PHÁP BIỂU DIỄN TRI THỨC TRONG LẬP TRÌNH LOGIC

NGÀNH: CÔNG NGHỆ THÔNG TIN

NGUYỄN THANH TÚ

Người hướng dẫn khoa học: PGS.TS.NGUYỄN THANH THỦY

HÀ NỘI 2006

LỜI CẢM ƠN

Trước tiên tôi xin gửi lời cảm ơn đặc biệt nhất tới PGS.TS Nguyễn Thanh Thủy, người đã định hướng đề tài và tận tình hướng dẫn chỉ bảo tôi trong suốt quá trình thực hiện luận văn thạc sỹ khoa học, từ những ý tưởng trong đề cương nghiên cứu, phương pháp giải quyết vấn đề, đến điều kiện lý tưởng để thực hành bản luận văn này.

Tôi xin chân thành bày tỏ lòng biết ơn tới tất cả các giáo sư, đặc biệt là GS José Júlio Alferes, trung tâm Logic tính toán, Universidade Nova de Líboa, Bồ Đào Nha đã cho tôi nhiều kiến thức quý báu về các vấn đề hiện đại của ngành logic tính toán, trí tuệ nhân tạo, công nghệ thông tin, đã cho tôi một môi trường tập thể, một khoảng thời gian khó quên và đã động viên, giúp đỡ và khích lệ tôi trong thời gian thực hiện luận văn này.

Bản luận văn này được hoàn thành với sự động viên giúp đỡ của các bạn bè lớp cao học Công nghệ thông tin 2004 – 2006. Tôi xin bày tỏ lòng cám ơn chân tình tới tất cả các bạn, nhất là các bạn đã dành nhiều thời gian quý báu của mình để trao đổi, giúp đỡ tôi khi gặp những vướng mắc trong suốt thời gian thực hiện bản luận văn này.

Nguyễn Thanh Tú
Công nghệ thông tin 2004 - 2006

MỤC LỤC

MỞ ĐẦU

Logic tính toán được các nhà logic học đưa ra vào những năm 1950, dựa trên các kỹ thuật tự động hóa quá trình suy diễn logic. Logic tính toán được phát triển thành lập trình logic vào những năm 1970. Từ đó hình thành một khái niệm quan trọng là lập trình khai báo (*declarative programming*) đối lập với lập trình cấu trúc (*procedural programming*). Về ý tưởng, các lập trình viên chỉ cần đưa ra khai báo của chương trình còn việc thực hiện cụ thể do máy tính tự xác lập, trong khi đó việc thực hiện các chương trình hướng thủ tục lại được xác lập cụ thể bởi lập trình viên. Ngôn ngữ Prolog là một công cụ thực hiện rõ ý tưởng này. Chương trình dịch Prolog đầu tiên ra đời đã chứng tỏ đó là một ngôn ngữ thực hành và được phổ biến trên toàn thế giới.

Sự phát triển của lập trình logic chính thức bắt đầu vào cuối những năm 1970. Những phát triển xa hơn đạt được vào đầu thập kỷ 80, bắt đầu với sự xuất hiện của quyển sách đầu tiên nói về các cơ sở lập trình logic. Việc lựa chọn lập trình logic làm mô hình cơ sở cho dự án Các hệ thống máy tính đời thứ 5 của Nhật (*Japanese Fifth Generation Computer Systems Project*) đã mở đầu cho sự phát triển của các ngôn ngữ lập trình logic khác.

Nhờ khả năng khai báo tự nhiên của lập trình logic, Prolog nhanh chóng trở thành một ứng cử viên cho việc biểu diễn tri thức. Tính đầy đủ của nó trở nên rõ ràng hơn khi mối liên hệ giữa các chương trình logic với cơ sở dữ liệu suy diễn được đưa ra vào giữa thập kỷ 80.

Việc sử dụng lập trình logic và cơ sở dữ liệu suy diễn để biểu diễn tri thức được gọi là “cách tiếp cận logic cho việc biểu diễn tri thức”. Cách tiếp cận này dựa trên ý tưởng là chương trình máy tính được cung cấp các đặc thù

logic của tri thức trong đó, do đó nó độc lập với bất kỳ cách thực hiện riêng biệt nào, với ngữ cảnh tự do, dễ dàng thao tác và suy diễn.

Chính vì vậy, cú pháp của ngôn ngữ lập trình phải kết hợp được bất kỳ chương trình nào với đặc thù khai báo của nó. Khi đó, việc thực hiện các phương pháp tính toán sẽ thông qua so sánh các thuộc tính cụ thể với cú pháp khai báo. Việc đưa ra một cú pháp thích hợp cho các chương trình logic được coi như một trong những lĩnh vực nghiên cứu quan trọng nhất và khó nhất trong lập trình logic.

Luận văn này sẽ trình bày các kết quả nghiên cứu về cú pháp và ngữ nghĩa của chương trình logic, bao gồm các lập trình logic thông thường và lập trình logic mở rộng, tiếp đó sẽ đề cập môi trường lập trình logic DLV (Datalog with Vel) và cách thức kết hợp môi trường logic này trong mã nguồn hướng đối tượng Java, cuối cùng trình bày hai bài toán minh họa (bài toán N quân hậu và bài toán Cây khung nhỏ nhất) được cài đặt trên DLV và được chạy trong mã nguồn hướng đối tượng Java.

Chương 1

CHƯƠNG TRÌNH LOGIC TỔNG QUÁT

1.1 Mở đầu

Ngôn ngữ Λ của một chương trình logic tổng quát Π được xây dựng trên bảng chữ cái A được định nghĩa như sau:

Định nghĩa 1.1 Bảng chữ cái A bao gồm các loại ký hiệu sau:

- Các biến
- Các hằng số đối tượng (có thể gọi là hằng số)
- Các ký hiệu hàm (function symbol)
- Các ký hiệu vị từ (predicate symbol)
- Các liên kết logic: "*not*", "$\leftarrow$" và ","
- Các ký hiệu phân cách "(" và ")"

□

Trong đó, *not* là liên kết logic được gọi là *phủ định ngầm* (*negation as failure*); biến là xâu bất kỳ bao gồm các ký tự của bảng chữ cái và các chữ số, được bắt đầu bằng chữ cái viết hoa; hằng số, ký hiệu hàm và ký hiệu vị từ là các xâu bắt đầu bởi chữ cái viết thường. Thông thường, sử dụng các chữ cái *p*, *q*,... cho các ký hiệu vị từ, *X*, *Y*, *Z*,... cho các biến, *f*, *g*, *h*,... cho các ký hiệu hàm và *a*, *b*, *c*,... cho các hằng số.

Định nghĩa 1.2 Một toán hạng được định nghĩa như sau:

(i) biến là toán hạng,

(ii) hằng số là toán hạng,

(iii) Nếu f là một ký hiệu hàm bậc n và $t_1,...,t_n$ là các toán hạng thì $f(t_1,...,t_n)$ cũng là một toán hạng.

□

Định nghĩa 1.3 Một toán hạng được gọi là có tính chất *nền* (*ground*) nếu không có biến nào xuất hiện trong nó.

□

Định nghĩa 1.4 Một nguyên tố biểu diễn trên bảng chữ cái A là một biểu thức có dạng $p(t_1,...,t_n)$, trong đó p là một ký hiệu vị từ trong A và t_i là các toán hạng. Nếu mọi t_i là toán hạng nền thì nguyên tố này cũng được gọi là có tính chất nền.

□

Một luật của chương trình được biểu diễn dưới dạng:

$$A_0 \leftarrow A_1, \dots, A_m, not\ A_{m+1}, \dots, not\ A_n. \qquad (1.1)$$

trong đó, A_i là các nguyên tố. Vế trái của luật được gọi phần đầu hay là kết luận, vế phải của luật là phần thân hay là giả thiết. Một tập các luật tạo thành một chương trình logic tổng quát (còn được gọi là chương trình logic thông thường). Chương trình logic tổng quát không chứa *not* thì được gọi là chương trình xác định. Các biểu thức và luật không chứa biến thì được gọi là có tính chất nền.

Định nghĩa 1.5 Không gian xác định Herbrand biểu diễn trên ngôn ngữ Λ của chương trình Π, ký hiệu là $HU(\Pi)$, là tập tất cả các toán hạng nền được biểu diễn với các hàm và hằng số trong Λ. Tập tất cả các nguyên tố nền trong ngôn ngữ của một chương trình Π được định nghĩa là $HB(\Pi)$ (cơ sở Herbrand của Π). Với một vị từ p, *atoms(p)* được định nghĩa là tập con của

$HB(\Pi)$ được biểu diễn dưới dạng vị từ p và với một tập các vị từ A, *atoms(A)* là một tập con các phần tử của $HB(\Pi)$ được biểu diễn dưới dạng các vị từ thuộc A.

□

Ví dụ 1.1 Xét chương trình logic thông thường Π sau:

$p(a).$

$p(b).$

$p(c).$

$p(f(X)) \leftarrow p(X).$

Ngôn ngữ của chương trình Π dựa trên bảng chữ cái bao gồm vị từ p, hàm f và các hằng số a, b và c.

$$HU(\Pi) = \{a, b, c, f(a), f(b), f(c), f(f(a)), f(f(b)), ...\}$$

$$HB(\Pi) = \{p(a), p(b), p(c), p(f(a)), p(f(b)), p(f(c)), p(f(f(a))), ...\}$$

□

Một chương trình logic được coi là một đặc tả cho phép xây dựng các lý thuyết có thể cho một thế giới quan còn các luật trong chương trình là những ràng buộc mà các lý thuyết này cần phải thỏa mãn. Ngữ nghĩa của chương trình logic được phân biệt tùy theo cách định nghĩa tính thỏa mãn các luật. Trong luận văn này sẽ sử dụng ngữ nghĩa về mô hình ổn định và các dạng mở rộng của nó. Với ngữ nghĩa này, các lý thuyết được xác định nhờ các tập nguyên tố nền, gọi là các mô hình ổn định của một chương trình. Ngữ nghĩa được định nghĩa như sau:

Định nghĩa 1.6 Mô hình ổn định của một chương trình xác định Π là một tập con nhỏ nhất S của HB sao cho với mọi luật $A_0 \leftarrow A_1, ..., A_m$ của Π, nếu $A_1, ..., A_m \in S$ thì $A_0 \in S$.

Mô hình ổn định của chương trình xác định Π được ký hiệu là $a(\Pi)$.

□

Gọi Π là một chương trình logic tổng quát bất kỳ. Với mọi tập phần tử *S*, đặt Π^S là một chương trình thu được từ Π bằng cách xóa:

(i) các luật có chứa *not A* với $A \in S$

(ii) tất cả các *not A* trong các luật còn lại.

Rõ ràng, Π^S không chứa *not* và tồn tại một mô hình ổn định đã định nghĩa ở trên. Nếu mô hình ổn định này trùng với *S*, thì ta nói rằng *S* là một mô hình ổn định của Π. Hay nói cách khác, mô hình ổn định của Π được biểu diễn bởi phương trình:

$$S = a\left(\Pi^S\right) \tag{1.2}$$

Một phần tử nền *P* là *đúng* trong *S* nếu $P \in S$, ngược lại *P* là *sai* (tức là $\neg P$ là *đúng*) trong *S*. Π suy diễn ra một biểu thức *f* (ký hiệu bởi $\Pi \models f$) nếu *f* là *đúng* trong mọi mô hình ổn định của Π. Ta cũng nói rằng câu trả lời cho một truy vấn nền *q* là *có* nếu *q* là *đúng* trong mọi mô hình ổn định của Π (tức là $\Pi \models q$), là *không* nếu $\neg q$ là *đúng* trong mọi mô hình ổn định của Π (tức là $\Pi \models \neg q$) và *không xác định* trong trường hợp còn lại.

Ví dụ 1.2 Xét ngôn ngữ chứa hai đối tượng *a* và *b* và một chương trình Π:

$$p(X) \leftarrow not\ q(X).$$
$$q(a).$$

Ta sẽ chỉ ra rằng tập $S = \{q(a), p(b)\}$ là một mô hình ổn định của Π. Xây dựng chương trình Π^S theo cách trên, ta có $\Pi^S = \{p(b) \leftarrow, q(a) \leftarrow\}$ có một mô hình ổn định trùng với *S*. Do đó *S* chính là mô hình ổn định của Π.

□