TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM
-------------o0o------------

BÀI GIẢNG MÔN
CÁC PHƯƠNG PHÁP HÌNH THỨC

Số tín chỉ: 2TC
Hệ đào tạo: Đại học
Ngành: Công nghệ phần mềm
Khoa: Công nghệ thông tin

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM
-------------o0o------------

BÀI GIẢNG MÔN
CÁC PHƯƠNG PHÁP HÌNH THỨC

Số tín chỉ: 2TC
Hệ đào tạo: Đại học
Ngành: Công nghệ phần mềm
Khoa: Công nghệ thông tin

Xác nhận của Khoa **Xác nhận của Bộ môn**

MỤC LỤC

CHƯƠNG 1. VAI TRÒ CỦA ĐẶC TẢ

1.1. Mục tiêu môn học

- Cung cấp kiến thức cốt lõi:
  - Về các phương pháp hình thức dựa trên cơ sở toán học chặt chẽ.
  - Một số kỹ thuật đặc tả và ngôn ngữ đặc tả thông dụng.
- Giúp người học:
  - Có khả năng diễn tả được các yêu cầu bài toán từ quá trình phân tích hệ thống.
  - Mô tả cách xử lý yêu cầu một cách chặt chẽ và đúng đắn.
  - Hệ thống hóa lại các kiểu dữ liệu vật lý và các kiểu dữ liệu trừu tượng.
- Giới thiệu ngôn ngữ đặc tả thông dụng: VDM, Z, …

1.2. Một số khái niệm cơ bản

- ***Các phương pháp hình thức (formal methods)***

  Trong tin học, thuật ngữ phương pháp hình thức (ngôn ngữ hình thức, đặc tả hình thức, …) thường được dùng để chỉ các kỹ thuật dựa trên cơ sở toán học dùng trong quá trình mô tả chi tiết (đặc tả), phát triển và kiểm chứng các hệ thống phần mềm cũng như phần cứng.

  Cách tiếp cận này thường được áp dụng cho các hệ thống có kết cấu chặt chẽ, đòi hỏi độ tin cậy và tính an toàn cao, đảm bảo rằng trong quá trình xây dựng và phát triển hệ thống xảy ra ít lỗi nhất.

  Các phương pháp hình thức đặc biệt hiệu quả trong các giai đoạn đầu của quá xây dựng hệ thống (thường là ở giai đoạn xác định yêu cầu và đặc tả hệ thống), tuy nhiên chúng có thể dùng trong toàn bộ quy trình phát triển hệ thống.
- Các phương pháp hình thức có thể được xếp loại theo 3 mức độ khác nhau:

- *Mức 0*: Đặc tả hình thức được sử dụng để đặc tả hệ thống trước khi phát triển nó. Trong nhiều trường hợp thì việc sử dụng phương pháp hình thức ở giai đoạn này tỏ ra đặc biệt hiệu quả, nhất là về mặt chi phí.
- *Mức 1*: Phát triển và kiểm chứng hình thức có thể được áp dụng để tạo ra một chương trình (hay một hệ thống) một cách tự động, dựa trên các đặc tả hình thức đã có trước đó. Quá trình này đặc biệt thích hợp đối với các hệ thống đòi hỏi độ tin cậy và tính an toàn cao.
- *Mức 2*: Chứng minh tự động.

➢ ***Đặc tả (specification)***

Mô tả cấu trúc hoạt động của các sự vật hiện tượng, quá trình nào đó. Việc mô tả này có thể ở mức độ khái quát, nhưng cũng có thể là những mô tả ở mức độ hết sức chi tiết.

Có nhiều ngôn ngữ cho phép đặc tả:

- Ngôn ngữ tự nhiên: tiếng Việt, tiếng Anh, tiếng Pháp, …
- Ngôn ngữ loài vật: chó, chim, mèo, …
- Ngôn ngữ lập trình: Pascal, C, C++, Java, CSharp, Visual Basic, …
- Ngôn ngữ hình thức: Là ngôn ngữ với bộ từ vựng, cú pháp, ngữ nghĩa được định nghĩa chặt chẽ dựa trên cơ sở của toán học.

➢ ***Đặc tả hình thức (formal specification)***

Đặc tả hình thức là đặc tả với các tính chất:

- Chính xác và nhất quán.
- Ngắn gọn và đầy đủ.
- Có thể xử lý được bởi máy vi tính.

Đặc tả hình thức có các ứng dụng như sau:

- Sử dụng trong giai đoạn phân tích, thiết kế, nhằm mục đích tạo ra các phác hoạ chi tiết, cụ thể và chặt chẽ về hệ thống sẽ được xây dựng.

- Trong quá trình xây dựng hệ thống, các đặc tả này sẽ là công cụ định hướng để đảm bảo hệ thống được xây dựng một cách phù hợp và đầy đủ.
- Sau khi hệ thống được xây dựng thì đặc tả sẽ đóng vai trò là thước đo để kiểm chứng, khẳng định hệ thống được tạo ra có đúng đắn và tin cậy hay không.

Ví dụ về mô tả quá trình xây dựng phần mềm theo mô hình thác nước

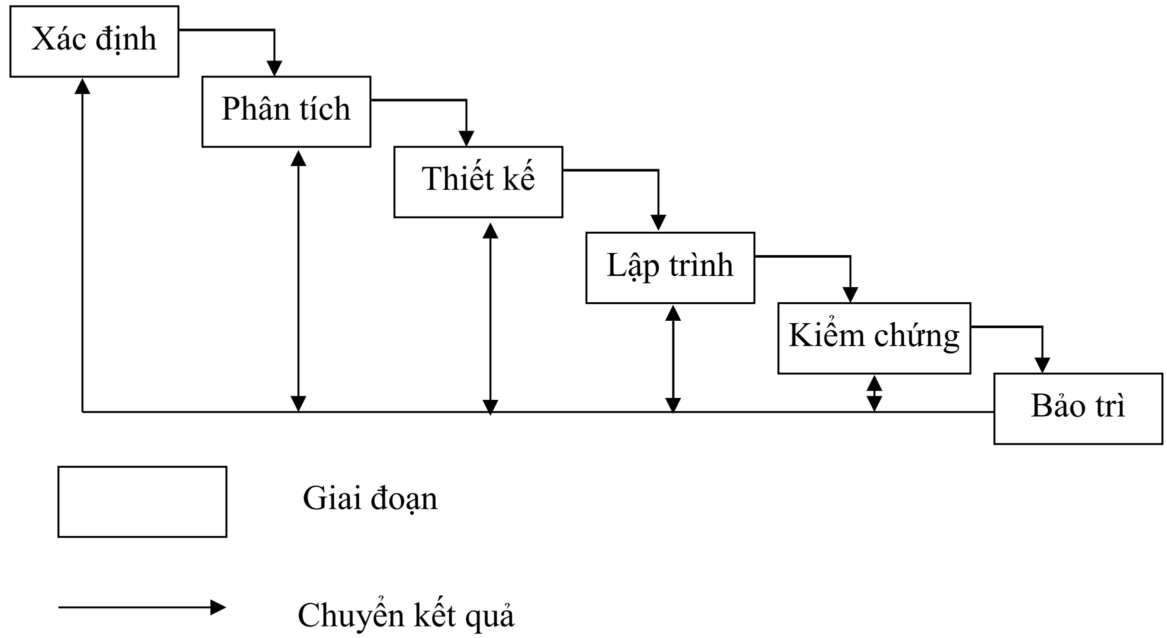
➢ ***Cách 1: Dùng ngôn ngữ tự nhiên***

Quy trình xây dựng phần mềm được tiến hành tuần tự qua các bước:

- Xác định yêu cầu
- Phân tích
- Thiết kế
- Lập trình
- Kiểm chứng
- Bảo trì

Sau khi tiến hành xong 1 bước sẽ chuyển giao kết quả cho bước kế tiếp hoặc có thể chuyển ngược lại cho các bước trước đó nếu còn phát hiện lỗi và sau quá trình lại tiếp tục.

Cách 2: Dùng sơ đồ



Đánh giá: Quan tâm đến kết quả thực hiện và chuyển giao giữa các giai đoạn:

- *Cách 1*: + Độ chính xác không cao, có thể gây ra hiểu lầm.
  + Dài dòng nếu mô tả đầy đủ.
  + Thích hợp cho việc mô tả chi tiết.
- *Cách 2:* + Độ chính xác tăng lên.
  + Trình bày ngắn gọn, trực quan.
  + Phù hợp cho việc mô tả một cách tổng quát.

1.3. Lịch sử ra đời và phát triển của đặc tả hình thức

Các kỹ thuật về đặc tả hình thức đã được sử dụng trong ngành Tin học trong suốt hơn 30 năm qua (từ những năm đầu của thập niên 70). Có rất nhiều mô hình cũng như ngôn ngữ đặc tả được ra đời và đa số chúng đều dựa trên cơ sở của toán học.

Các ngôn ngữ đặc tả được thiết kế và ra đời để sử dụng cho nhiều mục đích khác nhau. Các ngôn ngữ này được phân loại theo 3 tiêu chí chính:

- **Mức độ trừu tượng hoá:** Việc đặc tả hệ thống có thể có nhiều mức độ khác nhau.có thể một ngôn ngữ đặc tả chỉ dùng để mô tả các hệ thống tĩnh và nhỏ, nhưng cũng có thể hệ thống cần đặc tả rất to lớn và phức tạp cả về quy mô cũng như hoạt động. Mức độ trừu tượng hoá quyết định một ngôn ngữ đặc tả có thể dùng để mô tả hệ thống nào. Nếu bao quát quá nhiều thứ thì cồng kềnh, nhưng nếu đơn giản quá thì sẽ không có nhiều khả năng ứng dụng.

- **Phạm vi ứng dụng:** Mỗi ngôn ngữ đặc tả thường thiết kế nhằm mục đích phục vụ cho một hay một số lĩnh vực cụ thể. Ví dụ: VDM được thiết kế để sử dụng trong thiết kế các mạch số; phép toán mệnh đề được sử dụng trong đặc tả và chứng minh các thuật toán tính toán; UNITY được dùng trong đặc tả và kiểm chứng các hệ thống song song;….
- **Mục đích sử dụng:** Một ngôn ngữ đặc tả thường được thiết kế nhằm phục vụ cho một trong hai đối tượng chính là con người và máy tính. Điều khó khăn ở đây là phải làm sao dung hoà được điều này, vì nếu ngôn ngữ đó gần gũi với ngôn ngữ tự nhiên của con người thì máy tính rất khó phân tích, xử lý và diễn giải; ngược lại, nếu nó quá gần với ngôn ngữ máy tính thì con người gặp khó khăn trong quá trình sử dụng.

❖ **Lịch sử phát triển:**

Các ngôn ngữ đặc tả không hình thức:

- **Thế hệ thứ nhất:** Booch, Rumbaugh
- **Thế hệ thứ hai:** UML
- **Thế hệ thứ ba:** OOCL – Object-oriented Change and Learning (dùng trong khoa học nhận dạng và trí tuệ nhân tạo – biểu diễn tri thức).

Các ngôn ngữ đặc tả hình thức:

- OCL, Predicate Calculus, CDM, UNITY, VDM, Z
- Object-Z (Z++), VDM++

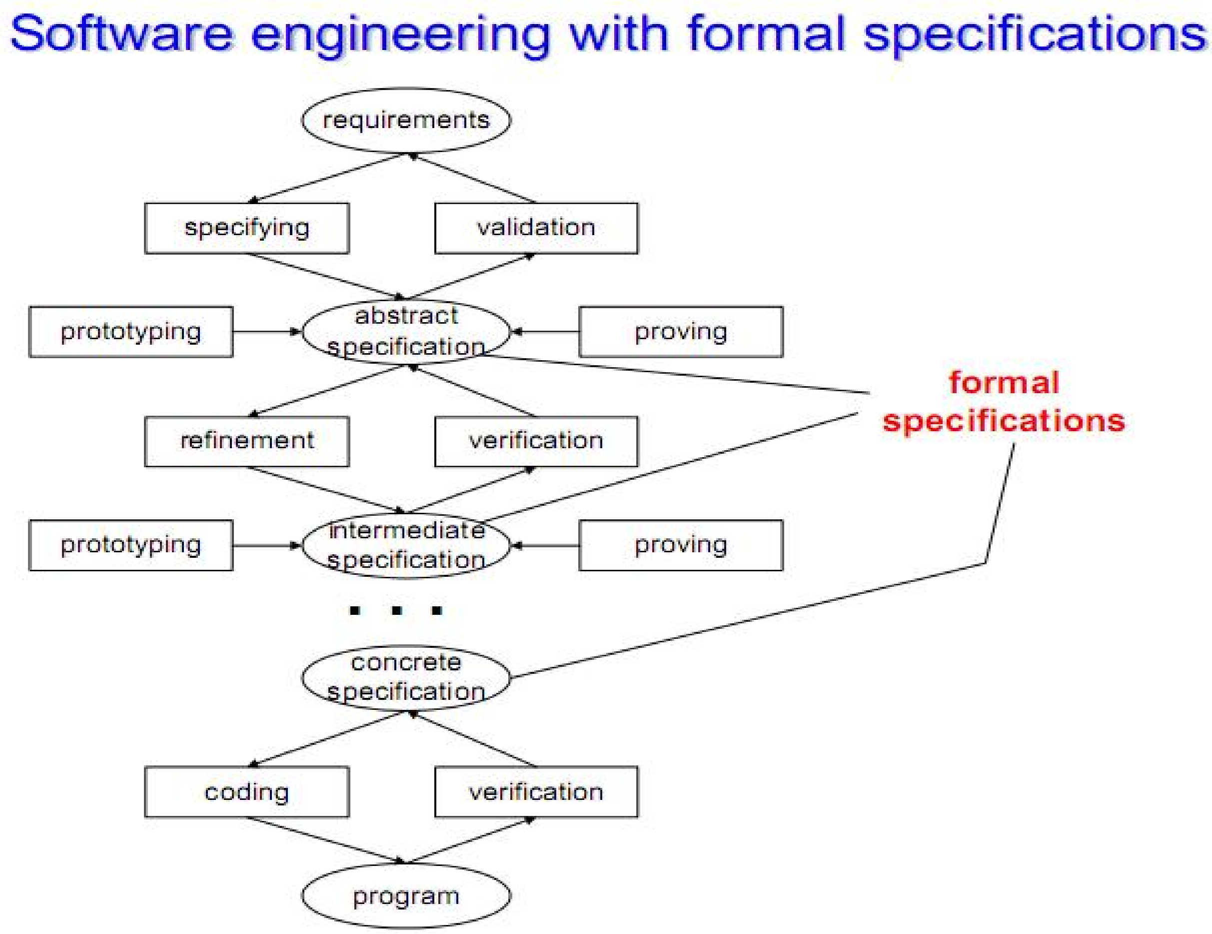
1.4. Đặc tả hình thức và quy trình phát triển phần mềm

1.4.1. ***Quy trình chung***

Vấn đề xảy ra nếu qui trình phát triển phần mềm không sử dụng đặc tả hình thức

- Với việc xác định yêu cầu không rõ ràng =>
  - Hiểu các yêu cầu theo những hướng khác nhau giữa khách hàng và người cung cấp, phát triển phần mềm.
  - Khó khăn trong việc đánh giá sự phù hợp của thiết kế và chương trình thực thi với các yêu cầu đã xác định.
- Mô hình phần mềm được thiết kế không theo chuẩn =>
  - Việc hiểu mô hình phần mềm theo hướng khác nhau của mỗi lập trình viên -> Chương trình sai.
  - Các thuộc tính của từng module thực thi trong chương trình sẽ không được mô tả rõ ràng.
- Việc kiểm thử phần mềm chỉ phát hiện ra lỗi nhưng không thể chỉ được ra vị trí lỗi.
- Đánh giá và đưa ra sản phẩm cho khách hàng muộn so với quy định.

1.4.2. *Qui trình phát triển phần mềm sử dụng đặc tả hình thức*



Notions

- **Specifying**
  - » Construction of a formal specification from informal requirements
- **Validation**
  - » Check whether the specification "correctly" represents the informal requirements
- **Prototyping**
  - » Execution of a specification for early validation
- **Proving**
  - » Conducting formal proofs of the properties of a specification
- **Refinement**
  - » Transition from an abstract to a concrete specification
- **Verification**
  - » Formal proof that an implementation satisfies the specification
- **Coding**
  - » Creation of a program satisfying the specification

Ưu điểm :

- Yêu cầu là những định nghĩa rõ ràng về mặt hình thức:
  - Khách hàng và nhà cung cấp có những trao đổi, quan điểm nhất quán.
  - Xác định tốt đầu vào cho pha thiết kế.
- Tài liệu của hệ thống phần mềm rõ ràng, cụ thể.
- Các định nghĩa hình thức của mô hình phần mềm sẽ là đầu vào cụ thể và rõ ràng cho việc lập trình.
- Sự chứng minh hình thức của các thuộc tính hệ thống.
- Xây dựng phát triển nguyên mẫu nhanh.
- Tự động sinh mã.
- Làm mịn theo kiểu bậc thang từ những đặc tả trừu tượng đến những đặc tả cụ thể.
- Đưa ra được nguồn gốc của việc kiểm tra dữ liệu từ những đặc tả hình thức.

Nhược điểm :

- Những hệ hình thức khó hiểu sẽ gây khó khăn trong việc trao đổi với khách hàng.
- Việc xây dựng ngôn ngữ hình thức là hết sức khó khăn và nó chỉ thích hợp cho những phần của hệ thống phần mềm thiếu khả năng thay đổi cấu hình hay qui mô.
- Việc sử dụng phương thức đặc tả nào và ngôn ngữ đặc tả nào là yêu cầu khá cao đối với người phát triển.
- Ngay cả việc đặc tả và chứng minh hình thức vẫn có thể có lỗi xãy ra.

  Giới hạn công cụ hỗ trợ cho việc đặc tả.

1.4.3. Phân tích

a) Lập các mô hình thế giới thực

- Mô hình dữ liệu
- Các ràng buộc
- Mô hình xử lý
- Mô hình trạng thái
- Mô hình thời gian
- Mô hình không gian

b) Dùng đặc tả

- Các sơ đồ
- Các phát biểu về ràng buộc
- Các quy định về công thức tính toán
- Thiết kế dữ liệu
- Các hàm kiểm tra ràng buộc

1.4.4. Thiết kế

a) Lập mô hình phần mềm

- Hệ thống dữ liệu
- Hệ thống giao diện
- Hệ thống xử lý

b) Dùng đặc tả

- Các sơ đồ
- Các thao tác trên màn hình
- Các hàm xử lý

1.4.5. Kiểm chứng

a) Kiểm tra tính đúng đắn

- Dữ liệu
- Hàm
- Giao diện

b) Dùng đặc tả

- Kiểm tra tính đúng đắn của hàm

Ứng dụng của đặc tả

- ❖ Mô tả lại các kết quả đã đạt được trong từng giai đoạn của quy trình công nghệ phần mềm. Ứng dụng dạng này thường được sử dụng trong các báo cáo.
- ❖ Phát sinh kết quả cho giai đoạn kế tiếp dựa vào đặc tả của giai đoạn trước.

CHƯƠNG 2.
CÁC CƠ SỞ CỦA ĐẶC TẢ

Các nội dung chính của chương:

2.1. LOGIC MỆNH ĐỀ

2.1.1.Mệnh đề

Mệnh đề là những phát biểu (câu) đúng hoặc sai, chứ không thể vừa đúng vừa sai.

Ví dụ : các câu sau là các mệnh đề

+ Hà Nội là thủ đô của nước Việt Nam

+ Trong tập số tự nhiên thì 1+1 =2

+ 1+2 = 2

+ 2 là một số nguyên tố

+ Thủ đô của nước Mỹ là London

Ví dụ : các câu sau không là mệnh đề

+ x- 1 =3

+ Bây giờ là mấy giờ ?

+ x –y=z

Các mệnh đề đúng được nói là có giá trị chân lý đúng (hay chân trị đúng), các mệnh đề sai được nói là có giá trị chân lý sai.

Người ta thường dùng các chữ cái p,q,r,t,... để đặt tên cho các mệnh đề. Giá trị chân lý đúng được ký hiệu là số 1 hoặc T (True- Đúng), giá trị chân lý sai được ký hiệu là 0 hoặc F (False - Sai).

2.1.2.Các phép toán trên mệnh đề

a.Phép phủ định

Phủ định của mệnh đề p là một mệnh đề « Không phải là p» và được ký hiệu là ¬p (đọc là KHÔNG p). Chân trị của ¬p là 0 nếu chân trị của p là 1 và ngược lại.

P	**¬p**
0	1
1	0

Ví dụ : Cho p là mệnh đề « 2 là một số nguyên tố» thì ¬p là mệnh đề « 2 không phải là số nguyên tố». Như vậy p có chân trị 1 nên ¬p có chân trị là 0.

Ví dụ: p = « 4 là số nguyên tố» thì ¬p = « 4 không phải là số nguyên tố» . p có chân trị là 0 nên ¬p có chân trị là 1.

b. Phép hội (phép AND - VÀ)

Mệnh đề **hội** của hai mệnh đề p và q là một mệnh đề được ký hiệu bởi p∧q (đọc là p VÀ q). Chân trị của p ∧ q là 1 nếu chân trị của cả p và q đều bằng 1, trong tất cả các trường hợp còn lại, p ∧ q có chân trị 0.

p	**q**	**p ∧ q**
0	0	0

0	1	0
1	0	0
1	1	1

Ví dụ : p = « 2 là số nguyên tố», q = « Hà nội là thủ đô của Việt Nam». Khi đó $p \wedge q$ là mệnh đề có chân trị là 1.

Ví dụ : cho p = « Hôm nay trời mưa» và q = « Hôm nay là thứ Ba». Vậy $p \wedge q$ là mệnh đề « Hôm nay trời mưa và là thứ Ba».

c. Phép tuyển (phép OR- HOẶC)

Mệnh đề **tuyển** của hai mệnh đề p và q được ký hiệu bởi $p \vee q$ (đọc là p HOẶC q). Chân trị của $p \vee q$ là 0 nếu chân trị của cả p và q đều bằng 0, trong tất cả các trường hợp còn lại $p \vee q$ có chân trị 1.

p	**q**	**$p \vee q$**
0	0	0
0	1	1
1	0	1
1	1	1

d.Phép kéo theo

Mệnh đề nếu p thì q được ký hiệu là $p \rightarrow q$ (đọc là p KÉO THEO q). Bảng chân trị của mệnh đề này như sau :

p	**q**	**$p \rightarrow q$**
0	0	1
0	1	1
1	0	0
1	1	1

e.Phép tương đương (Phép kéo theo hai chiều)

Mệnh đề nếu p thì q và ngược lại được ký hiệu $p \leftrightarrow q$ (đọc là p KHI VÀ CHỈ KHI q). Bảng chân trị của mệnh đề này như sau :

p	**q**	**$p \leftrightarrow q$**
0	0	1
0	1	0
1	0	0
1	1	1

2.1.3. Dạng mệnh đề

Là các biểu thức logic được xây dựng bằng cách kết hợp các biến mệnh đề với nhau bởi các phép nối theo một thứ tự nhất định. Mỗi dạng mệnh đề có một chân trị xác định đối với từng bộ chân trị của các biến mệnh đề. Tập tất cả các chân trị của dạng mệnh đề ứng với từng chân trị của các biến mệnh đề lập thành bảng chân trị của dạng mệnh đề đó.

Ví dụ : Dạng mệnh đề: $E(p, q, r) = p \wedge (q \vee r)$ có bảng chân trị như sau :

P	**q**	**r**	**$q \vee r$**	**$E=p\wedge(q\vee r)$**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Bài tập áp dụng: Lập bảng chân trị của các dạng mệnh đề sau:

a) $p \rightarrow (q \wedge r)$

b) $(p \wedge q) \vee (p \wedge r)$

c) $(p \wedge q) \vee (\neg p \wedge r)$

d) $(p \rightarrow q) \wedge (\neg p \rightarrow r)$

2.1.4. Tương đương logic

Hai dạng mệnh đề E và F được gọi là tương đương logic nếu chúng có cùng một bảng chân trị. Khi ấy ta viết $E \Leftrightarrow F$ và mệnh đề dạng $E \leftrightarrow F$ luôn mang chân trị bằng 1 cho dù các biến có lấy giá trị nào đi nữa.

Ví dụ: Xét 2 mệnh đề: $p \rightarrow q$ và $\neg p \vee q$

P	**q**	**$\neg$p**	**p $\rightarrow$ q**	**$\neg$p $\vee$ q**
0	0	1	1	1
0	1	1	1	1
1	0	0	0	0
1	1	0	1	1

Như vậy ta nói 2 mệnh đề trên là tương đương logic, và được viết là:

$$(p \rightarrow q) \Leftrightarrow (\neg p \vee q)$$

2.1.5. Hệ quả logic

Dạng mệnh đề F được gọi là hệ quả logic của dạng mệnh đề E nếu $E \rightarrow F$ luôn có chân trị đúng. Khi đó ta viết là $E \Rightarrow F$. Ta có thể nói cách khác: E có hệ quả logic là F.

2.1.6. Các nguyên tắc thay thế

1. **Quy tắc 1:** trong dạng mệnh đề E, nếu ta thay thế biểu thức con F bởi một dạng mệnh đề tương đương logic thì dạng mệnh đề thu được vẫn còn tương đương logic với E.

2. **Quy tắc 2:** Giả sử dạng mệnh đề E(p, q, r,…) là một hằng đúng. Nếu ta thay thế những nơi p xuất hiện trong E bởi một dạng mệnh đề tuỳ ý F (p’, q’, r’,…) thì dạng mệnh đề nhận được theo các biến p, q, r,…, p’, q’, r’,… vẫn còn là một hằng đúng.

2.1.7. Các quy luật logic

Với p, q, r là các biến mệnh đề, 1 là hằng đúng và 0 là hằng sai, ta có các tương đương logic như sau:

1. **Phủ định của phủ định**

$$\neg\neg p \Leftrightarrow p$$

2. **Quy tắc De Morgan**

$$\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$$

và $$\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$$

3. **Luật giao hoán**

$$p \wedge q \Leftrightarrow q \wedge p$$

và $$p \vee q \Leftrightarrow q \vee p$$

4. **Luật kết hợp**

$$p \wedge (q \wedge r) \Leftrightarrow (p \wedge q) \wedge r$$

và $$p \vee (q \vee r) \Leftrightarrow (p \vee q) \vee r$$

5. **Luật phân bố**

$$p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$$

và $$p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$$

6. **Luât luỹ đẳng**

$$p \wedge p \Leftrightarrow p$$

và $$p \vee p \Leftrightarrow p$$

7. Luật trung hoà

$$p \wedge 1 \Leftrightarrow p$$

và $$p \vee 0 \Leftrightarrow p$$

8. Luật về phần tử bù

$$p \wedge \neg p \Leftrightarrow 0$$

và $$p \vee \neg p \Leftrightarrow 1$$

9. Luật thống trị

$$p \wedge 0 \Leftrightarrow 0$$

và $$p \vee 1 \Leftrightarrow 1$$

10. Luật hấp thụ

$$p \wedge (p \vee q) \Leftrightarrow p$$

và $$p \vee (p \wedge q) \Leftrightarrow p$$

Ví dụ 1: Hãy chứng minh dạng mệnh đề sau là một hằng đúng:

$$[(r \rightarrow s) \wedge [(r \rightarrow s) \rightarrow (\neg t \vee u)]] \rightarrow (\neg t \vee u)$$

Chứng minh:

Thay $r \rightarrow s$ bởi p và $\neg t \vee u$ bởi q, ta đưa về bài toán chứng minh dạng mệnh đề sau là một hằng đúng:

$$[p \wedge (p \rightarrow q)] \rightarrow q$$

Ta có: $[p \wedge (p \rightarrow q)] \rightarrow q$

$\Leftrightarrow [p \wedge (\neg p \vee q)] \rightarrow q$

$\Leftrightarrow [(p \wedge \neg p) \vee (p \wedge q)] \rightarrow q$

$\Leftrightarrow [0 \vee (p \wedge q)] \rightarrow q$

$\Leftrightarrow (p \wedge q) \rightarrow q$

$\Leftrightarrow \neg p \vee \neg q \vee q$

$\Leftrightarrow \neg p \vee 1 \;\; \Leftrightarrow 1$

Ví dụ 2: Từ $(p \wedge q) \rightarrow r$, ta có:

$(p \wedge q) \rightarrow r$

$\Leftrightarrow \neg p \vee \neg q \vee r$

$\Leftrightarrow \neg p \vee (\neg q \vee r)$

$\Leftrightarrow p \rightarrow (\neg q \vee r)$

$\Leftrightarrow p \rightarrow (q \rightarrow r)$

Như vậy : $(p \wedge q) \rightarrow r \Leftrightarrow p \rightarrow (q \rightarrow r)$

Áp dụng vào lập trình, xét 3 đoạn chương trình sau :

Đoạn chương trình 1:

```
int x, y, z = 3
    for (i=1 ; i<=10 ; i++)  {
        x = z – i ;
```

```
        y = z + 2*i ;
        if ((x > 0) && (y > 0)
                printf (“x + y = %d\n”, x + y) ;
}
```

Đoạn chương trình 2:

```
int x, y, z = 3
for (i=1 ; i<=10 ; i++)  {
        x = z – i ;
        y = z + 2*i ;
        if (x > 0)
        if  (y > 0)
                printf (“x + y = %d\n”, x + y) ;
}
```

Đoạn chương trình 3:

```
int x, y, z = 3
for (i=1 ; i<=10 ; i++)  {
        x = z – i ;
        y = z + 2*i ;
        if (y > 0)
        if (x > 0)
```

printf(“x + y = %d\n”, x + y) ;

}

Số lần thực hiện phép so sánh của đoạn chương trình trên có sự khác nhau : ở (1) là 20 lần (10 phép so sánh x và 10 phép so sánh y) ; ở (2) chỉ là 12 lần (10 phép so sánh x và 2 lần so sánh y ứng với i = 1 và i = 2) ; ở (3) thì vẫn lại là 20 lần.

2.1.8. Các quy tắc suy diễn

1. **Modus Ponens (khẳng định)**

$$[(p \rightarrow q) \wedge p] \Rightarrow q$$

2. **Syllogism (Tam đoạn luận)**

$$[(p \rightarrow q) \wedge (q \rightarrow r)] \Rightarrow (p \rightarrow r)$$

3. **Modus Tollens (phủ định)**

$$[(p \rightarrow q) \wedge \neg q] \Rightarrow \neg p$$

4. **Tam đoạn luận rời**

$$[(p \vee q) \wedge \neg p] \Rightarrow q$$

5. **Quy tắc mâu thuẫn**

$$[(p1 \wedge p2 \wedge \ldots \wedge pn) \rightarrow q] \Leftrightarrow [(p1 \wedge p2 \wedge \ldots \wedge pn \wedge \neg q) \rightarrow 0]$$

6. **Quy tắc chứng minh theo trường hợp**

$$[(p \rightarrow r) \wedge (q \rightarrow r)] \Rightarrow (p \vee q) \rightarrow r)$$

Ví dụ: Chứng minh $f(n) = n^3 + 2n$ luôn chia hết cho 3.

Ta có: f(n) = n(n2 + 2). Lấy n là một số nguyên tuỳ ý, khi đó có 2 trường hợp xảy ra :

TH1 : n chia hết cho 3, như vậy dễ thấy f(n) chia hết cho 3. **(1)**

TH2: n không chia hết cho 3, khi đó đặt n = 3k ± 1 (k là số nguyên nào đó), ta có:

$n^2 + 2 = (3k \pm 1)^2 + 2 = 9k^2 \pm 6k + 3 = 3(k^2 \pm 2k + 1)$

suy ra f(n) cũng chia hết cho 3. **(2)**

Từ **(1)** và **(2),** ta kết luận f(n) chia hết cho 3 trong mọi trường hợp.

2.2. LOGIC VỊ TỪ

2.2.1. Vị từ

Là một khẳng định P(x, y,…) trong đó có chứa một số biến x, y, ... lấy giá trị trong những tập hợp cho trước A,B,... sao cho:

a) Bản thân P(x, y,...) không phải là một mệnh đề.

b) Nếu thay x, y,… bằng những phần tử cố định nhưng tuỳ ý $a \in A$, $b \in B$,…thì ta được một mệnh đề P(a, b,…), tức là chân trị của nó hoàn toàn xác định. Khi đó x, y,… gọi là các biến tự do của vị từ.

Nói một cách khác, một vị từ là một hàm số dạng:

$$f: X \rightarrow \mathrm{B}$$

trong đó: X = A × B × … và **B = {0, 1}**

Ví dụ: P(n) = ‘‘n là một số nguyên tố’’ là một vị từ theo một biến tự do $n \in$N.

$$f: \mathrm{N} \rightarrow \mathrm{B}$$

Với n = 2, 5, 7, 11 ta có các mệnh đề đúng P(2), P(5), P(7), P(11) : còn với n = 4, 8 thì ta có các mệnh đề sai P(4), P(8).

2.2.2. Lượng từ

Giả sử p(x) là vị từ theo một biến tự do $x \in$A, khi đó có 3 trường hợp có thể xảy ra :

TH1 : khi thay x bởi một phần tử bất kỳ a∈ A, ta đều được mệnh đề đúng p(a).

TH2 : với một (hoặc một số) giá trị $a \in A$ thì p(a) là mệnh đề đúng, và với một số giá trị $b \in A$ thì p(b) là mệnh đề sai.

TH3 : khi thay x bởi một phần tử bất kỳ $a \in A$, ta đều được mệnh đề sai p(a).

Đối với TH1 thì mệnh đề ‘‘với mọi $x \in A$, p(x)’’ là một mệnh đề đúng, ký hiệu bởi ‘‘$\forall x \in A$, p(x)’’. Bản chất của mệnh đề này là phép VÀ ($\wedge$)

Nếu TH1 hoặc TH2 xảy ra thì mệnh đề ‘‘tồn tại $x \in A$, p(x)’’ là một mệnh đề đúng, ký hiệu bởi ‘‘$\exists\, x \in A$, p(x)’’. Bản chất của mệnh đề này là phép HOẶC ($\vee$)

Các mệnh đề $\forall x \in A$, p(x) và $\exists\, x \in A$, p(x) được gọi là lượng từ hoá của vị từ p(x) bởi lượng từ phổ dụng ($\forall$) và lượng từ tồn tại ($\exists$).

Chú ý:

a. Trong mệnh đề lượng từ hoá, x không còn là biến tự do nữa mà nó bị ràng buộc bởi các lượng từ.
b. TH3 ở trên có thể được viết lại : $\forall x \in A, \neg p(x)$. Như vậy, phủ định của mệnh đề $\forall x \in A$, p(x) xảy ra khi TH2 hoặc TH3 xảy ra,tức là mệnh đề $\exists x \in A, \neg p(x)$ là mệnh đề đúng. Rút ra :

Phủ định của $\forall x \exists A$, p(x) là mệnh đề $\exists x \in A, \neg p(x)$

Phủ định của $\exists x \in A$, p(x) là mệnh đề $\forall x \in A, \neg p(x)$

2.3. LÝ THUYẾT TẬP HỢP

Ở đây ta chỉ nhắc lại một số khái niệm cơ bản nhất, cũng như các ký hiệu dùng trong lý thuyết tập hợp mà thôi :

a) Nếu a là một phần tử thuộc tập hợp A, ta viết $a \in A$, ngược lại ta viết $a \notin A$.

b) Tập hợp A thoả một tính chất nào đó, tính chất ở đây được biểu diễn dưới dạng một vị từ p(x), ta viết : A = $\{x \in U / p(x)\}$, trong đó U được gọi là tập vũ trụ.

Ví dụ : A = $\{x \in N$ / x là số nguyên tố$\}$

A = $\{x \in Z / x^2 < 5\}$

c) Ngoài ra có thể biểu diễn tập hợp bằng cách liệt kê tất cả các phần tử của nó, ví dụ 2 ở trên có thể được viết lại A = $\{-2, -1, 0, 1, 2\}$

d) Tập hợp không có phần tử nào cả gọi là tập hợp rỗng, được ký hiệu là $\varnothing$

e) Giả sử A và B là 2 tập hợp con của tập vũ trụ U, ta nói A là tập con của B (hay A được bao hàm trong B, hay B bao hàm A), ký hiệu $A \subset B$ nếu:

$$\forall x \in U, (x \in A) \Rightarrow (x \in B)$$

f) Nếu $A \subset B$ và $B \subset A$, ta nói A bằng B, được ký hiệu A = B. Như vậy rõ ràng A = B khi và chỉ khi:

$$\forall x \in U, (x \in A) \Leftrightarrow (x \in B)$$

g) Hợp ($\cup$) , giao ($\cap$) và phần bù của tập hợp:

$$A \cup B = \{x \in U / (x \in A) \vee (x \in B)\}$$

$$A \cap B = \{x \in U / (x \in A) \wedge (x \in B)\}$$

$\bar{A} = \{x \in U / (x \notin A)\}$, $\bar{A}$ được gọi là phần bù của A trong U.

h) Một số tính chất trên tập hợp: Cho A, B, C là các tập con tuỳ ý của U, ta có:

i. Tính giao hoán:

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

ii. Tính kết hợp:

$$A \cup (B \cup C) = (A \cup B) \cup C$$

$$A \cap (B \cap C) = (A \cap B) \cap C$$

iii. Luật De Morgan:

$$\overline{A \cup B} = \bar{A} \cap \bar{B}$$

$$\overline{A \cap B} = \bar{A} \cup \bar{B}$$

iv. Tính phân bố:

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

v. Phần tử trung hoà:

$$A \cup \varnothing = A$$

$$A \cap U = A$$

vi. Phần bù:

$$A \cup \bar{A} = U$$

$$A \cap \bar{A} = \varnothing$$

vii. Tính thống trị:

$$A \cup U = U$$

$$A \cap \varnothing = \varnothing$$

CHƯƠNG 3
ĐẶC TẢ CÁC THÀNH PHẦN CƠ BẢN

3.1. ĐẶC TẢ KIỂU DỮ LIỆU

Một kiểu dữ liệu bao gồm:

- Tập hợp các giá trị
- Hệ thống các phép toán cơ sở

Dựa trên các phép toán này, ta có thể đặc tả các phép toán còn lại.

Một phép toán (có thể được gọi là 1 hàm) là một ánh xạ riêng phần trên tập D $\subset$ X :

$$\begin{aligned} f: \quad & X \to Y \\ & x \mapsto y \end{aligned}$$

Domain : điều kiện x $\in$ D

Where: y là ảnh của x.

D được gọi là miền xác định của f.

Ta có các kiểu dữ liệu cơ bản như sau:

Ký hiệu	Tên kiểu	Các phép toán cơ sở
N	Các số tự nhiên (số nguyên không âm)	+, –, *, / >, >=, <, <=, =, !=
Z	Các số nguyên	
Q	Các số hữu tỉ	
R	Các số thực	
B	Kiểu logic, gồm 2 giá trị True và False	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \Rightarrow, \Leftrightarrow$

C	Kiểu các ký tự	>, >=, <, <=, =, !=
S	Các chuỗi ký tự	

❖ Xét kiểu dữ liệu chuỗi (S):

- Là tập hợp tất cả các chuỗi ký tự.
- Các hàm (phép toán) cơ sở bao gồm:

1. $$\begin{aligned} Rong: \quad & S \to B \\ & s \mapsto b \end{aligned}$$

   Domain:

   Where: b=True ⇔ s là chuỗi rỗng

2. $$\begin{aligned} Khoitao: \quad & C \to S \\ & c \mapsto s \end{aligned}$$

   Domain:

   Where: s là chuỗi gồm đúng 1 ký tự là c

3. $$\begin{aligned} Chendau: \quad & C \times S \to S \\ & (c,s) \mapsto t \end{aligned}$$

   Domain:

   Where: t được tạo lập bằng cách chèn ký tự c vào đầu chuỗi s

4. $$\begin{aligned} Xoadau: \quad & S \to S \\ & s \mapsto t \end{aligned}$$

   Domain: ¬Rong(s)

   Where: t được tạo lập bằng cách xóa đi ký tự đầu của s.

5. $$\begin{aligned} PTdau: \quad & S \to C \\ & s \mapsto c \end{aligned}$$

Domain: ¬Rong(s)

Where: c là ký tự đầu tiên của chuỗi s

3.2. ĐẶC TẢ HÀM

3.2.1. Cú pháp chung khi đặc tả hàm (dùng ngôn ngữ toán học) :

$$\begin{aligned} f: \quad & X \to Y \\ & x \mapsto y \end{aligned}$$

Domain : điều kiện x ∈ D

Where: y là ảnh của x.

X: tập các giá trị nguồn

Y: tập các giá trị đích

D: Miền xác định của hàm

3.2.2. Ví dụ 1:

Đặc tả các hàm sau:

1. Tìm căn bậc 2 của 1 số thực
2. Tìm phần nguyên của 1 số thực
3. Hàm DIV (2 số không âm)
4. Hàm MOD (2 số không âm)

Giải:

1. $$\begin{aligned} f: \quad & R \to R \\ & x \mapsto y \\ & Domain: x \geq 0 \\ & Where: (y \geq 0) \wedge (y * y = x) \end{aligned}$$

$$
\begin{aligned}
g:\quad & R \to Z \\
& x \mapsto k
\end{aligned}
$$

2. $Domain:$

$$
\begin{aligned}
Where: \quad & (x \ge 0) \wedge (k \le x < k+1) \\
\vee \quad & (x < 0) \wedge (k-1 < x \le k)
\end{aligned}
$$

3.
$$
\begin{aligned}
& DIV: \quad N \times N \to N \\
& \qquad (a,b) \mapsto k \\
& Domain: b != 0 \\
& Where: k = g(a/b)
\end{aligned}
$$

4.
$$
\begin{aligned}
& MOD: \quad N \times N \to N \\
& \qquad (a,b) \mapsto k \\
& Domain: b != 0 \\
& Where: k = a - DIV(a,b) * b
\end{aligned}
$$

3.2.3. Ví dụ 2

Dựa vào các hàm cơ sở trên kiểu dữ liệu chuỗi, hãy đặc tả các hàm sau:

1. f: xóa 2 ký tự đầu của 1 chuỗi
2. g:Tìm phần tử thứ 2 của 1 chuỗi
3. h:Chèn 1 ký tự vào ví trí thứ 2 của chuỗi

Giải:

1.
$$
\begin{aligned}
& f: \quad S \to S \\
& \qquad s \mapsto t \\
& Domain: \neg Rong(Xoadau(s)) \\
& Where: t = Xoadau(Xoadau(s))
\end{aligned}
$$

2.
$$
\begin{aligned}
& g: \quad S \to C \\
& \qquad s \mapsto c \\
& Domain: \neg Rong(Xoadau(s)) \\
& Where: c = PTdau(Xoadau(s))
\end{aligned}
$$

$$\begin{aligned} h: \quad & S \times C \to S \\ & (s,c) \mapsto t \end{aligned}$$

3. $Domain : \neg Rong(s)$

$$\begin{aligned} Where: \quad & ch \in C \wedge ch = PTDau(s) \\ \wedge \quad & t = Chendau(ch, Chendau(c, (Xoadau(s)))) \end{aligned}$$

Câu này còn có thể được đặc tả bằng 1 cách khác như sau (dựa vào 2 hàm f và g của câu 1 và câu 2):

$$\begin{aligned} h: \quad & S \times C \to S \\ & (s,c) \mapsto t \end{aligned}$$

$Domain : \neg Rong(s)$

$$\begin{aligned} Where: \quad & (PTDau(t) = PTDau(s)) \\ \wedge \quad & (g(t) = c) \\ \wedge \quad & (f(t) = Xoadau(s)) \end{aligned}$$

hay có thể triển khai đầy đủ như sau:

$$\begin{aligned} h: \quad & S \times C \to S \\ & (s,c) \mapsto t \end{aligned}$$

$Domain : \neg Rong(s)$

$$\begin{aligned} Where: \quad & (PTDau(t) = PTDau(s)) \\ \wedge \quad & (PTdau(Xoadau(t) = c) \\ \wedge \quad & (Xoadau(Xoadau(t)) = Xoadau(s)) \end{aligned}$$

3.3. ĐẶC TẢ ĐỆ QUY

Đặc tả đệ quy thực chất là cách thức đặc tả hàm theo dạng truy hồi. Một đặc tả được gọi là đệ quy nếu trong phần biểu thức Where của hàm có sử dụng đến chính bản thân hàm đang được đặc tả.

Ví dụ 1 : Ta xét hàm tìm kích thước của một chuỗi ký tự bất kỳ:

$$
\begin{aligned}
&Len: \quad S \to N \\
&\qquad\quad s \mapsto n \\
&Domain: \\
&Where: \qquad (Rong(s) \wedge n = 0) \\
&\qquad\quad \vee \qquad (\neg Rong(s) \wedge n = 1 + Len(Xoadau(s))
\end{aligned}
$$

Ví dụ 2 : Xét hàm tính tổng các phần tử trong mảng a, kích thước n

MANG = Z^n

$$
\begin{aligned}
&Tong: \quad MANG \times N \times Z \to N \\
&\qquad\quad (a, n) \qquad \mapsto t \\
&Domain: \\
&Where: \qquad (n = 0 \wedge t = 0) \\
&\qquad\quad \vee \qquad (n > 0 \wedge a[n-1] != x \wedge t = a[n-1] + Tong(a, n-1)
\end{aligned}
$$

Ví dụ 3 : Xét hàm đếm số lần xuất hiện của 1 số nguyên x trong mảng a, kích thước n

$$
\begin{aligned}
&Dem: \quad MANG \times N \times Z \to N \\
&\qquad\quad (a, n, x) \qquad \mapsto d \\
&Domain: \\
&Where: \qquad (n = 0 \wedge d = 0) \\
&\qquad\quad \vee \qquad (n > 0 \wedge a[n-1] != x \wedge d = Dem(a, n-1, x) \\
&\qquad\quad \vee \qquad (n > 0 \wedge a[n-1] = x \wedge d = 1 + Dem(a, n-1, x)
\end{aligned}
$$

Ví dụ 4 : Hàm kiểm tra mảng tăng dần

$$
\begin{aligned}
&KTTang: \qquad MANG \times N \to B \\
&\qquad\qquad\quad (a, n) \quad \mapsto b \\
&Domain: \\
&Where: \qquad (n = 0 \wedge b = False) \\
&\qquad\quad \vee \qquad (n = 1 \wedge b = True) \\
&\qquad\quad \vee \qquad (n > 1 \wedge a[n-1] < a[n-2] \wedge b = False) \\
&\qquad\quad \vee \qquad (n > 1 \wedge a[n-1] \geq a[n-2] \wedge b = KTTang(a, n-1))
\end{aligned}
$$

3.4. CÁC QUY TẮC TÍNH TOÁN

Trong quá trình xây dựng các hệ thống phần mềm, đặc biệt là các hệ thống quản lý luôn có 1 yêu cầu không thể thiếu là thực hiện việc tính toán theo 1 hoặc 1 số quy tắc nhất định nào đó. Do vậy, việc đặc tả các quy tắc này là cần thiết, để nhằm đảm bảo cho việc tính toán được chính xác và đầy đủ.

Nguyên tắc chung khi đặc tả các quy tắc tính toán:
1. Đặt tên cho các đại lượng có liên quan đến quá trình tính toán
2. Xác định kiểu dữ liệu của các đại lượng này
3. Đặc tả hàm tính toán cho từng quy tắc

Ví dụ : Ta xét phần mềm quản lý thư viện với yêu cầu tính toán tiền phạt cho mỗi quyển sách khi trả sách trễ hạn với các yêu cầu khác nhau có thể có như sau :

a) Mỗi ngày trả trễ phạt 1000đ.

b) Mỗi ngày trả trễ phạt 1000đ, nhưng từ ngày trễ thứ 5 trở đi thì phạt 2000đ/ngày.

c) Có 2 loại sách A và B. Hình thức phạt như câu b), nhưng mức phạt khác nhau cho mỗi loại sách :

Loại A : 1000đ và 2000đ

Loại B : 1500đ và 2500đ

Giải :

Gọi t : số tiền phạt ($t \in N$)

n: số ngày trả trễ hạn ($n \in N$)

l: loại sách ($l \in \{A, B\}$)

a)
$$\begin{aligned}&Phat: \ N \to N\\&\qquad n \mapsto t\\&Domain:\\&Where: t = 1000 * n\end{aligned}$$

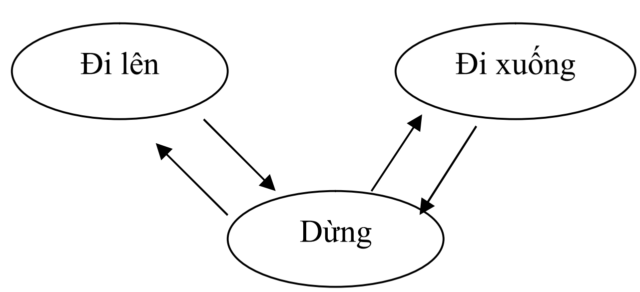
b)
$$\begin{aligned}&Phat: \ N \to N\\&\qquad n \mapsto t\\&Domain:\\&Where: ((n < 5) \wedge t = 1000 * n) \vee\\&\qquad ((n \geq 5) \wedge (t = 4000 + 2000 * (n - 4))\end{aligned}$$

c)
$$\begin{aligned}&Phat: \ N \to N\\&\qquad n \mapsto t\\&Domain:\\&Where: ((l = A) \wedge (n < 5) \wedge t = 1000 * n) \vee\\&\qquad ((l = A) \wedge (n \geq 5) \wedge (t = 4000 + 2000 * (n - 4)) \vee\\&\qquad ((l = B) \wedge (n < 5) \wedge t = 1500 * n) \vee\\&\qquad ((l = B) \wedge (n \geq 5) \wedge (t = 6000 + 2500 * (n - 4))\end{aligned}$$

3.5. CÁC SƠ ĐỒ TRẠNG THÁI

Một sơ đồ trạng thái thường dùng các hình vẽ để mô tả trạng thái và hoạt động của một đối tượng, 1 quá trình nào đó. Tuy nhiên, việc sử dụng hình vẽ khi xử lý trên máy tính lại tỏ ra hết sức khó khăn, độ chính xác lại không cao.

Ví dụ 1 : Sơ đồ hoạt động của 1 thang máy



Nguyên tắc chung khi đặc tả 1 sơ đồ trạng thái :
1. Gán các giá trị 1, 2, 3, … cho các trạng thái có trong sơ đồ

2. Gọi t_1 : trạng thái hiện hành ($t_1 \in N$)

   t_2 : trạng thái có thể chuyển đến trực tiếp từ t_1 ($t_2 \in N$)

3. Xây dựng các ràng buộc trên t_1 và t_2 dưới dạng một mệnh đề để mô tả các tình trạng có thể có của sơ đồ.

Để đặc tả sơ đồ trong ví dụ trên, ta đặt :

1 : Đi lên

2 : Dừng

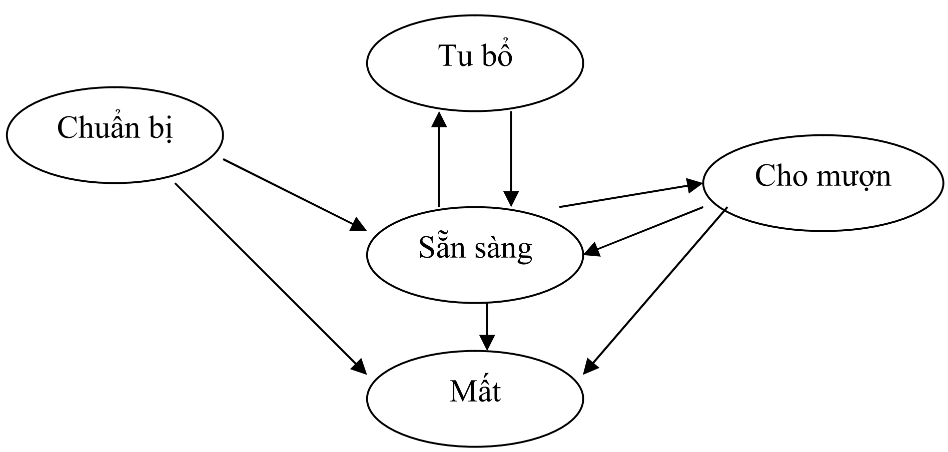
3 : Đi xuống

t_1 : trạng thái hiện hành t_2 : trạng thái có thể chuyển đến từ t_1

Ta có đặc tả :

$$
\begin{aligned}
S = \quad & (t1 = 1 \wedge t2 = 2) \vee \\
& (t1 = 2 \wedge (t2 = 1 \vee t2 = 3)) \vee \\
& (t1 = 3 \wedge (t2 = 2))
\end{aligned}
$$

Ví dụ 2: Sơ đồ mô trả trạng thái của 1 quyển sách trong thư viện



Để đặc tả sơ đồ trong ví dụ trên, ta đặt :

1 : Trạng thái chuẩn bị

2 : Sẵn sàng

3 : Tu bổ

4 : Cho mượn

5 : Mất

t_1 : trạng thái hiện hành t_2 : trạng thái có thể chuyển đến từ t_1

Ta có đặc tả :

$$
\begin{aligned}
S = \quad & ((t1 = 1 \wedge (t2 = 2 \vee t2 = 5)) \vee \\
& (t1 = 2 \wedge (t2 = 3 \vee t2 = 4 \vee t2 = 5)) \vee \\
& (t1 = 3 \wedge (t2 = 2)) \vee \\
& (t1 = 4 \wedge (t2 = 2 \vee t2 = 5)) \vee \\
& (t1 = 5 \wedge t2 = 5))
\end{aligned}
$$

3.6. CÁC ĐỐI TƯỢNG HÌNH HỌC

Một điểm trên mặt phẳng được biểu diễn dưới dạng (x, y) trong đó x và y là 2 số thực biểu diễn hoành độ và tung độ của điểm đó. Ta định nghĩa kiểu dữ liệu :

DIEM = R × R

M ∈ DIEM

Ký hiệu : M.x hoành độ của M

M.y tung độ của M

Đặc tả 1 đối tượng hình học trên mặt phẳng là đặc tả hàm kiểm tra các điểm có liên quan có thỏa các điều kiện ràng buộc để tạo nên đối tượng hay không.

Dạng thức chung :

$$\begin{array}{ll} Doituong: & DIEM^N \to B \\ & (M_1, M_2, ...) \mapsto b \\ Domain: & \\ Where: b \Leftrightarrow các ràng buôc trên M_1, M_2, ... & \end{array}$$

Ví dụ: Cho A, B, C, D là các điểm. Đặc tả các phát biểu sau :

a) A, B là 2 điểm trùng nhau

b) A, B, C là 3 điểm thẳng hàng

c) A, B, C là 3 đỉnh của 1 tam giác

d) AB // CD

e) AB $\perp$ CD

f) C là trung điểm của AB

g) Hàm tính độ dài của đoạn thẳng AB

h) Hàm tính khoảng cách từ điểm A đến đường thẳng qua BC

Giải :

a)
$$\begin{array}{ll} Trung: & DIEM \times DIEM \to B \\ & (A, B) \mapsto b \\ Domain: & \\ Where: b \Leftrightarrow (A.x = B.x \wedge A.y = B.y) & \end{array}$$

b)
$$\begin{array}{ll} ThangHang: & DIEM^3 \to B \\ & (A, B, C) \mapsto b \\ Domain: & \\ Where: b \Leftrightarrow ((B.x - A.x) * (C.y - A.y) - (B.y - A.y) * (C.x - A.x) != 0) & \end{array}$$

c)
$$\begin{array}{ll} TamGiac: & DIEM^3 \to B \\ & (A, B, C) \mapsto b \\ Domain: & \\ Where: b \Leftrightarrow \neg ThangHang(A, B, C) & \end{array}$$

d)

$$
\begin{array}{lll}
CungPhuong: & DIEM^4 & \to B \\
 & (A,B,C,D) & \mapsto b
\end{array}
$$

$Domain: \neg Trung(A,B) \wedge \neg Trung(C,D)$

$Where: b \Leftrightarrow ((B.x - A.x)*(D.y - C.y) - (B.y - A.y)*(D.x - D.x) = 0)$

e)

$$
\begin{array}{lll}
VuongGoc: & DIEM^4 & \to B \\
 & (A,B,C,D) & \mapsto b
\end{array}
$$

$Domain: \neg Trung(A,B) \wedge \neg Trung(C,D)$

$Where: b \Leftrightarrow ((B.x - A.x)*(D.x - C.x) + (B.y - A.y)*(D.y - D.y) = 0)$

f)

$$
\begin{array}{lll}
TrungDiem: & DIEM^3 & \to B \\
 & (A,B,C) & \mapsto b
\end{array}
$$

$Domain: \neg Trung(A,B)$

$Where: b \Leftrightarrow (ThangHang(A,B,C) \wedge Dodai(A,C) = Dodai(B,C))$

g)

$$
\begin{array}{lll}
Dodai: & DIEM \times DIEM & \to R \\
 & (A,B) & \mapsto r
\end{array}
$$

$Domain:$

$Where: r = \sqrt{(B.x - A.x)*(B.x - A.x) + (B.y - A.y)*(B.y - A.y)}$

h)

$$
\begin{array}{ll}
KhoangCach: & DIEM^3 \to R \\
 & (A,B,C) \mapsto d
\end{array}
$$

$Domain: \neg Trung(B,C)$

$Where: (ThangHang(A,B,C) \wedge d = 0) \vee$

$(\neg ThangHang(A,B,C) \wedge$

$H \in DIEM \wedge ThangHang(B,C,H) \wedge VuongGoc(A,H,B,C) \wedge$

$d = Dodai(A,H))$

3.7. CÁC RÀNG BUỘC

3.7.1. Ràng buộc trên kiểu dữ liệu

Gọi T là tập hợp nền của kiểu dữ liệu T

Ràng buộc trên kiểu dữ liệu T là hàm kiểm tra tính hợp lệ :

$f: \quad T \rightarrow B$ $\quad\quad t \mapsto b$ Domain: Where: b⇔ t thỏa ràng buộc

Ví dụ : Kiểu dữ liệu NGAY, biểu diễn một (ngày, tháng, năm). Kiểu có thể có các tập hợp nền như sau :

a) NGAY ⊂ N×N×Z   d∈NGAY → d = (ng, t, n)

b) NGAY ⊂ {1..31}×{1..12}×Z   d∈NGAY → d = (ng, t, n)

c) NGAY ⊂ N×Z   d∈NGAY → d = (stt, n)

d) NGAY ⊂ {1..366}×Z   d∈NGAY → d = (stt, n)

Đặc tả hàm kiểm tra ngày hợp lệ dựa trên 4 tập hợp nền ở trên.

Giải :

Trước tiên, ta đặc tả hàm kiểm tra năm nhuận :

$$
\begin{aligned}
&Nhuan: \quad Z \rightarrow B \\
&\quad\quad\quad\quad n \mapsto b \\
&Domain: \\
&Where: b \Leftrightarrow ((Mod(n,400) = 0) \vee \\
&\quad\quad\quad\quad (Mod(n,4) = 0) \wedge (Mod(n,100) != 0))
\end{aligned}
$$

Sau đó đặc tả các hàm kiểm tra tính hợp lệ đối với từng tập hợp nền:

a) NGAY ⊂ N×N×Z

$$
\begin{aligned}
&HopLe: && NGAY \to B\\
& && d \quad\quad \mapsto b\\
&Domain: &&\\
&Where: b \Leftrightarrow (&& (d.ng \ge 1 \wedge d.ng \le 31) \wedge\\
& && (d.t \ge 1 \wedge d.t \le 12) \wedge\\
& && (d.t \in \{4,6,9,11\} \wedge d.ng \le 30) \wedge\\
& && (d.t = 2 \wedge \neg Nhuan(d.n) \wedge d \le 28) \wedge\\
& && (d.t = 2 \wedge Nhuan(d.n) \wedge d \le 29))
\end{aligned}
$$

b) NGAY $\subset$ {1..31}×{1..12}×Z

$$
\begin{aligned}
&HopLe: && NGAY \to B\\
& && d \quad\quad \mapsto b\\
&Domain: &&\\
&Where: b \Leftrightarrow (&& (d.t \in \{4,6,9,11\} \Rightarrow d.ng \le 30) \wedge\\
& && (d.t = 2 \wedge \neg Nhuan(d.n) \Rightarrow d.ng \le 28) \wedge\\
& && (d.t = 2 \wedge Nhuan(d.n) \Rightarrow d.ng \le 29))
\end{aligned}
$$

c) NGAY $\subset$ N×Z

$$
\begin{aligned}
&HopLe: && NGAY \to B\\
& && d \quad\quad \mapsto b\\
&Domain: &&\\
&Where: b \Leftrightarrow ((d.stt \ge 1 \wedge d.stt \le 366) \wedge &&\\
& && (\neg Nhuan(d.n) \Rightarrow d.stt \le 365))
\end{aligned}
$$

d) NGAY $\subset$ {1..366}×Z

$$
\begin{aligned}
&HopLe: && NGAY \to B\\
& && d \quad\quad \mapsto b\\
&Domain: &&\\
&Where: b \Leftrightarrow (\neg Nhuan(d.n) \Rightarrow d.stt \le 365) &&
\end{aligned}
$$

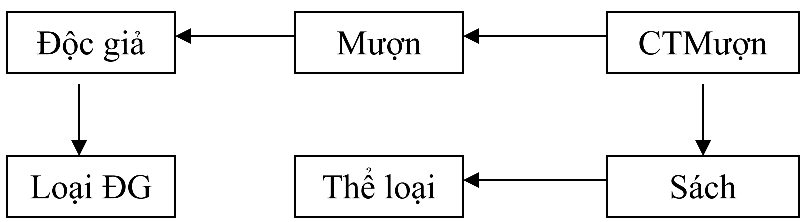
3.7.2. Ràng buộc trên sơ đồ logic

Các ràng buộc trên sơ đồ logic chính là các ràng buộc toàn vẹn trong cơ sở dữ liệu. Việc đặc tả các ràng buộc trên sơ đồ logic sử dụng ngôn ngữ đại số quan hệ (các vị từ và lượng từ).

Các ràng buộc toàn vẹn bao gồm :

- Các ràng buộc khóa nội, khóa ngoại
- Ràng buộc thuộc tính
- Ràng buộc thời điểm
- Ràng buộc liên thuộc tính
- v.v...

Ví dụ 1 : Xét phần mềm quản lý thư viện với sơ đồ logic như sau :



Độc giả (MĐG, MLĐG, Họtên, Địachỉ, Điệnthoại)

Loại ĐG (MLĐG, Tên, Ghichú)

Thể loại (MTL, Tên, Ghichú)

Sách (MS, MTL, Tên, NămXB, Tácgiả)

Mượn (MMượn, MĐG, Ngàymượn)

CTMượn (MMượn, MS, Ngàytrả)

Hãy đặc tả các ràng buộc sau :

- Ràng buộc về khóa nội, khóa ngoại
    a) MĐG là duy nhất trong Độc giả
    b) MTL của Sách phải có trong Thể loại
- Ràng buộc thuộc tính :
    c) Ngày mượn phải khác rỗng
- Ràng buộc thời điểm :
    d) Tại 1 thời điểm, sách chỉ có thể cho 1 độc giả mượn
- Ràng buộc liên thuộc tính :
    e) Ngày trả sau ngày mượn

Giải :

a) $\forall d1, d2 \in Docgia \bullet d1 \neq d2 \Rightarrow d1.MDG \neq d2.MDG$

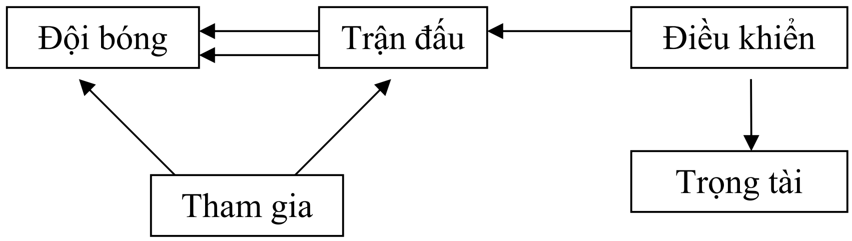
b) $\forall d \in Docgia \bullet \exists l \in Theloai \mid d.MTL = l.MTL$

c) $\forall m \in Muon \bullet m.Ngaymuon \neq NULL$

d)
$$\begin{aligned}
&\forall m \in Muon.\forall ctm \in CTMuon \bullet m.MMuon = ctm.MMuon \Rightarrow \\
&\neg(\exists m1 \in Muon.\exists ctm1 \in CTMuon \bullet \\
&(m1.MMuon \neq m.MMuon \wedge ctm1.MMuon \neq ctm.MMuon)) \wedge \\
&(ctm1.MS = ctm.MS) \wedge \\
&(m1.Ngaymuon \geq m.Ngaymuon) \wedge \\
&(m1.Ngaymuon < ctm.Ngaytra))
\end{aligned}$$

e)
$$\begin{aligned}
&\forall m \in Muon.\forall ctm \in CTMuon \bullet m.MMuon = ctm.MMuon \Rightarrow \\
&m.Ngaymuon < ctm.Ngaytra
\end{aligned}$$

Ví dụ 2: Xét phần mềm quản lý giải bóng đá với sơ đồ sau :



Đặc tả các ràng buộc :

a) Các trọng tài trong cùng 1 trận đấu phải cùng thuộc 1 quốc gia

b) Trong 1 trận đấu chỉ có 1 trọng tài chính

c) Trọng tài và các đội thi đấu không cùng thuộc 1 quốc gia

d) Mội trận đấu chỉ có 2 đội tham gia

CHƯƠNG 4
ĐẶC TẢ VÀ TÍNH ĐÚNG ĐẮN CỦA HÀM

4.1. CÁC PHƯƠNG PHÁP KIỂM TRA TÍNH ĐÚNG ĐẮN

4.1.1. Đặt vấn đề

Ta xét hàm f với đặc tả như sau:

$$\begin{aligned} f: \quad & X \to Y \\ & x \mapsto y \end{aligned}$$

Domain : P(x)

Where: Q(x, y)

Với cài đặt tương ứng:

```
Y f(X)
{
    //…
}
```

Bài toán đặt ra ở đây là hãy cho biết cài đặt trên có phù hợp với đặc tả đã có hay không?

Đây chính là bài toán về kiểm tra tính đúng đắn của hàm. Nếu cài đặt cụ thể của hàm f là phù hợp với đặc tả, ta nói cài đặt đó là đúng, ngược lại, cài đặt đó là chưa chính xác.

Để kiểm tra tính đúng đắn của 1 hàm sau khi cài đặt, người ta sử dụng một số phương pháp khác nhau, hoặc cũng có thể kết hợp đồng thời các phương pháp này. Trong các phần tiếp theo của chương, chúng tôi sẽ giới thiệu các tiếp cận của 1 số phương pháp phổ biến hiện nay.

4.1.2. Kiểm tra động

Phương pháp kiểm tra động (dynamic testing), hay còn được gọi là phương pháp thử nghiệm được thực hiện như sau :

1. Lập các bộ số làm thử nghiệm, hay còn được gọi là các bộ dữ liệu thử nghiệm (test case) : $(x, y) \in X \times Y$.
2. Lần lượt cho thực hiện hàm f với các giá trị x đã chọn, ghi nhận lại kết quả y_0.
3. So sánh y và y_0.

Phương pháp này chính là phương pháp kiểm thử chủ đạo được áp dụng trong giai đoạn lập trình và giai đoạn kiểm thử của quy trình phát triển phần mềm trên thế giới hiện nay.

Các công cụ hỗ trợ phương pháp kiểm tra động (thử nghiệm) :

- ❖ Công cụ phát sinh các dữ liệu thử nghiệm.
- ❖ Công cụ cho phép thực thi hàm và ghi nhận kết quả.
- ❖ Công cụ kiểm tra tính phù hợp của cài đặt hàm so với đặc tả.
- ❖ Các công cụ tự động phát sinh hàm từ đặc tả, cho thực hiện hàm phát sinh và tự động kiểm tra kết quả thực hiện. Những công cụ tự động dạng này hiện vẫn chưa phổ biến, số lượng lẫn chất lượng của các công cụ này vẫn chưa đáp ứng được sự mong đợi của giới Công nghệ phần mềm.
- ❖ Phương pháp kiểm tra động này còn được biết đến với 1 tên gọi khác là "kiểm thử hộp đen" (black box testing), vì phương pháp không chú trọng đến cài đặt bên trong của các hàm, nó chỉ quan tâm đến kết quả thực hiện của hàm để khẳng định hàm đó có phù hợp với đặc tả không mà thôi, hoàn toàn chưa đề cập đến việc sửa chữa cài đặt của hàm lại sao cho phù hợp.

4.1.3. Kiểm tra tĩnh

Phương pháp kiểm tra tĩnh, còn được gọi là "kiểm thử hộp trắng" (white box testing). Phương pháp này thực hiện theo cơ chế lần lượt đọc các lệnh trong chương trình nguồn (phần cài đặt của hàm), sau đó chứng minh bằng tay, hoặc chứng minh tự động tính đúng đắn của hàm sau từng lệnh.

Cho hàm f với đặc tả và cài đặt tương ứng :

$$\begin{aligned} f:\quad & X \to Y \\ & x \mapsto y \end{aligned}$$

Domain : P(x)

Where: Q(x, y)

Y f(X)

{

lệnh 1;

lệnh 2;

…

lệnh k;

From P(x)

Q_1(x, y) lệnh 1 được thực hiện.

Q_2(x, y) lệnh 2 được thực hiện.

...

Q_k(x, y) lệnh k được thực hiện.

Infer Q(x, y)

4.2. CHỨNG MINH VỚI CÁC LUẬT SUY DIỄN

4.2.1. Mở đầu

Cho E1, E2 là 2 mệnh đề. Trong quá trình chứng minh, nếu suy diễn được E1 thì cũng suy diễn được E2, ta ký hiệu :

$$E1 \mapsto E2 \text{ hay } \frac{E1}{E2}$$

Nếu $E1 \mapsto E2$ và $E2 \mapsto E1$ thì ta ký hiệu :

$$\frac{\underline{E1}}{E2} \Leftrightarrow \frac{E1}{E2} \, và \, \frac{E2}{E1}$$

4.2.2. Các luật suy diễn

4.2.2.1. *Các luật suy diễn trên phép phủ định (¬)*

$$\frac{E}{\neg\neg E} \, và \, \frac{\neg\neg E}{E} \qquad \textbf{(¬-1)}$$

4.2.2.2. *Các luật suy diễn trên phép tuyển (∨)*

1. **Suy diễn mở rộng:** Nếu đã có E rồi thì có thể suy diễn là có E hoặc là có bất cứ thứ gì khác nữa.

   $$\frac{E}{E \vee X} \; ; \; \frac{E}{X \vee E} \qquad \textbf{(∨-MR)}$$

   Ví dụ : Tôi đã có nhà. Vậy có thể nói "Tôi đã có nhà hoặc xe", hay "Tôi đã có xe hoặc nhà", bất kể là tôi đã thực sự có xe hay chưa.

2. **Suy diễn theo trường hợp :** Ta đã có E1 hoặc E2 (ít nhất 2 trong 2 thứ). Nếu từ E1 suy diễn ra được X, từ E2 cũng suy diễn ra được X. Như vậy ta suy diễn ra được X.

   $$\frac{E1 \vee E2; E1 \mapsto X; E2 \mapsto X}{X} \qquad \textbf{(∨-TrH)}$$

Ví dụ : Nếu đi làm thêm thì tôi sẽ có tiền.

Nếu được nhận học bổng thì tôi cũng sẽ có tiền

Mà hiện giờ tôi đã đi làm thêm hoặc đã nhận học bổng.

Vậy tôi có tiền.

3. **De Morgan:**

$$\frac{\neg(E1 \vee E2)}{\neg E1} \; ; \; \frac{\neg(E1 \vee E2)}{\neg E2} \qquad \textbf{(}\vee\textbf{-DM)}$$

hay :

$$\frac{\neg(E1 \vee E2)}{\neg E1 \wedge \neg E2} \qquad \text{(chiều thuận)}$$

Ví dụ : Tôi không có xe hay nhà gì cả. Vậy tôi không có xe, mà cũng chẳng có nhà.

4. **Tổng hợp trên phép tuyển:**

$$\frac{\neg E1; \neg E2}{\neg(E1 \vee E2)} \qquad \textbf{(}\vee\textbf{-TH)}$$

hay :

$$\frac{\neg E1 \wedge \neg E2}{\neg(E1 \vee E2)} \qquad \text{(chiều đảo)}$$

Ví dụ : Ngược lại, tôi không có xe, tôi cũng chẳng có nhà. Vậy tôi chẳng có xe hay nhà gì cả.

5. **Thay thế trên phép tuyển:**

$$\frac{E1 \vee E2; E2 \mapsto X}{E1 \vee X} \; ; \; \frac{E1 \vee E2; E1 \mapsto X}{X \vee E2} \qquad \textbf{(}\vee\textbf{-TT)}$$

6. **Giao hoán trên phép tuyển:**

$$\frac{E1 \vee E2}{E2 \vee E1} \qquad \textbf{(}\vee\textbf{-GH)}$$

7. **Kết hợp trên phép tuyển:**

$$\frac{E1 \vee (E2 \vee E3)}{(E1 \vee E2) \vee E3} \qquad \textbf{(}\vee\textbf{-KH)}$$

8. **Phân phối trên phép tuyển:**

$$\frac{E1 \vee (E2 \wedge E3)}{(E1 \vee E2) \wedge (E1 \vee E3)} \; ; \; \frac{(E1 \vee E2) \wedge (E1 \vee E3)}{E1 \vee (E2 \wedge E3)} \quad \textbf{(}\vee\textbf{-PP)}$$

4.2.2.3. *Các luật suy diễn trên phép hội (∧)*

a)

$$\frac{\neg(\neg E1 \vee \neg E2)}{E1 \wedge E2} \text{ và } \frac{E1 \wedge E2}{\neg(\neg E1 \vee \neg E2)} \qquad \textbf{(}\wedge\textbf{-1)}$$

Ví dụ : Tôi đã có cả xe lẫn nhà. Vậy không thể có chuyện tôi không có xe hay không có nhà, và ngược lại.

b) **Suy diễn thu hẹp (rút gọn) :**

$$\frac{E1 \wedge E2}{E1} \text{ và } \frac{E1 \wedge E2}{E2} \qquad \textbf{(}\wedge\textbf{-RG)}$$

Ví dụ : Tôi đã có cả xe lẫn nhà. Vậy tôi có xe.

Tôi đã có cả xe lẫn nhà. Vậy tôi có nhà.

c) **Tổng hợp trên phép hội :**

$$\frac{E1; E2}{E1 \wedge E2} \qquad \textbf{(}\wedge\textbf{-TH)}$$

Ví dụ : Tôi đã có xe. Tôi cũng có nhà. Vậy tôi có cả xe lẫn nhà.

d) **Thay thế trên phép hội:**

$$\frac{E1 \wedge E2; E2 \mapsto X}{E1 \wedge X} \; ; \; \frac{E1 \wedge E2; E1 \mapsto X}{X \wedge E2} \qquad \textbf{(}\wedge\textbf{-TT)}$$

Ví dụ : Tôi đã có cả xe lẫn nhà. Nếu có nhà thì tôi có đất. Vậy tôi có cả xe lẫn đất.

e) **Phân phối trên phép hội:**

$$\frac{E1 \wedge (E2 \vee E3)}{(E1 \wedge E2) \vee (E1 \wedge E3)} \; ; \; \frac{(E1 \wedge E2) \vee (E1 \wedge E3)}{E1 \wedge (E2 \vee E3)} \qquad \textbf{(}\wedge\textbf{-PP)}$$

4.2.2.4. *Các luật suy diễn trên phép kéo theo (⇒)*

a) Suy diễn theo luật bài tam :

$$\frac{}{E \vee \neg E1} \qquad \textbf{(BT)}$$

b) Suy diễn dựa trên mẫu thuẫn :

$$\frac{E \wedge \neg E1}{X} \qquad \textbf{(MT)}$$

c) Suy diễn kéo theo :

$$\frac{\neg E1 \vee E2}{E1 \Rightarrow E2} \; ; \; \frac{E1 \Rightarrow E2}{\neg E1 \vee E2} \qquad \textbf{(}\Rightarrow\textbf{-1)}$$

d) Suy diễn tương đương:

$$\frac{(E1 \Rightarrow E2) \wedge (E2 \Rightarrow E1)}{E1 \Leftrightarrow E2} \qquad \textbf{(}\Rightarrow\textbf{-2)}$$

4.2.3. Chứng minh luật suy diễn

4.2.3.1. *Dạng thức :*

Để chứng minh luật suy diễn : $\frac{E}{F}$, ta áp dụng dạng thức như sau :

From E

Kết quả 1 Luật suy diễn áp dụng(các kết quả sử dụng)

Kết quả 2 Luật suy diễn áp dụng(các kết quả sử dụng)

...

Infer F

4.2.3.2. *Ví dụ*

Chứng minh lại các luật suy diễn sau :

1. $\dfrac{E1 \wedge E2}{E1}$

Giải :

1. **From** $E1 \wedge E2$ (h)

    1. $\neg(\neg E1 \vee \neg E2)$ $\wedge$-1(h)

    2. $\neg\neg E1$ $\vee$-DM(1)

    Infer *E1* $\neg$-1(2)

2. $\dfrac{E1; E2}{E1 \wedge E2}$

    From $E1; E2$ (h)

    1. $\neg\neg E1; \neg\neg E2$ $\neg$-1(h)

    2. $\neg(\neg E1 \vee \neg E2)$ $\vee$-TH(1)

    Infer $E1 \wedge E2$ $\wedge$-1(2)

3. $\dfrac{E1 \wedge E2; E2 \mapsto X}{E1 \wedge X}$

From $E1 \wedge E2; E2 \mapsto X$ (h)

1. $E1$ $\wedge$-RG(h)

2. $E2$ $\wedge$-RG(h)

3. X h(1)

Infer $E1 \wedge X$ $\wedge$-TH(1,3)

4. $$\frac{E1 \vee E2; E2 \mapsto X}{E1 \vee X}$$

From $E1 \vee E2; E2 \mapsto X$ (h)

1. From $E1$ (h1)

Infer $E1 \vee X$ $\wedge$-MR(h1)

2. From $E2$ (h2)

2.1 X h(h2)

Infer $E1 \vee X$ $\wedge$-MR(2.1)

Infer $E1 \vee X$ $\wedge$-TrH(h,1,2)

CHƯƠNG 5
NGÔN NGỮ ĐẶC TẢ Z

Giới thiệu

Ký pháp Z, hay còn được gọi là ngôn ngữ Z được xây dựng dựa trên lý thuyết tập hợp và logic toán học. Đây là một ngôn ngữ toán học chặt chẽ, được sử dụng chủ yếu trong đặc tả hình thức để đặc tả các yêu cầu chức năng của 1 hệ thống, đặc biệt là hệ thống phần mềm.

Ngôn ngữ Z không được thiết kế để mô tả các yêu cầu phi chức năng của hệ thống, ví dụ như công dụng, hiệu năng, kích thước hay độ tin cậy của hệ thống. Ngôn ngữ cũng không được thiết kế cho các đặc tả theo thời gian hay xử lý song song. Muốn làm được điều này, ta phải kết hợp Z cùng với các công cụ khác nữa.

5.1. Các thành phần của ngôn ngữ

5.1.1. Logic toán học

5.1.1.1. Logic mệnh đề

Các khái niệm về logic mệnh đề cũng như hệ thống ký hiệu của logic mệnh đề trong ký pháp Z hoàn toàn giống với ngôn ngữ toán học thông thường.

Có 5 phép toán mệnh đề trong ngôn ngữ Z được liệt kê trong bảng dưới đây, với độ ưu tiên giảm dần từ trên xuống dưới :

Phép toán	Ý nghĩa
$\neg$	Phủ định
$\wedge$	Tuyển
$\vee$	Hội
$\Rightarrow$	Kéo theo
$\Leftrightarrow$	Tương đương

Ý nghĩa của mỗi phép toán đã được trình bày trong Chương 2 (Các cơ sở của đặc tả). Do vậy, phần này xin được phép không nhắc lại chi tiết nữa.

5.1.1.2. Logic vị từ

Các khái niệm và định nghĩa của logic vị từ cũng hoàn toàn giống như trong toán học, ngoại trừ ký hiệu có khác biệt.

a) Cú pháp chung của 1 vị từ sử dụng lượng từ:

$\boldsymbol{Q\ x : A \mid p \bullet q}$

Trong đó:

Q : lượng từ (∃ hoặc ∀).
x : biến ràng buộc.
A : tập các giá trị của x.
p : ràng buộc trên biến.
q : vị từ.

$\boldsymbol{\exists x : A \mid p \bullet q}$ được phát biểu như sau: "Tồn tại 1 giá trị x thuộc về tập A, sao cho x thỏa p thì ta có q".

$\boldsymbol{\forall x : A \mid p \bullet q}$ được phát biểu như sau: "Với mọi giá trị x thuộc về tập A, sao cho x thỏa p thì ta có q".

Ta có các phát biểu tương đương như sau:

$$\exists x : A \mid p \bullet q \Leftrightarrow \exists x : A \bullet p \wedge q$$

$$\forall x : A \mid p \bullet q \Leftrightarrow \forall x : A \bullet p \Rightarrow q$$

Ví dụ 1: Vị từ "Tồn tại 1 số tự nhiên x lớn hơn 5" được viết như sau trong ký pháp Z:

$\exists x : N \bullet x > 5$

Như vậy, ký hiệu thuộc về 1 tập hợp trong Z là dấu hai chấm (:), thay vì là dấu $\in$ như trong toán học.

Ký hiệu "sao cho" sử dụng dấu gạch đứng (|), và ký hiệu "thì" sử dụng dấu chấm tròn (•).

Ví dụ 2: Tương tự như thế đối với lượng từ "với mọi" ∀. "Với mọi y thuộc về tập số tự nhiên, y lớn hơn hay bằng 0".

$$\forall y : N \bullet y \geq 0$$

b) **"Tồn tại ít nhất 2 giá trị thỏa điều kiện"** được biểu diễn như sau:

$$\exists x, y : A \mid (x \neq y) \wedge p \bullet q$$

"Tồn tại duy nhất 1 giá trị thỏa điều kiện" được biểu diễn như sau:

$$\exists_1 x : A \mid p \bullet q$$

Ví dụ 3: Phát biểu "có duy nhất 1 quyển sách trên bàn" được biểu diễn dưới dạng vị từ như sau:

$$(\exists b : Books \bullet b \in Desk) \wedge (\forall c : Books \mid c \in Desk \bullet c = b)$$

$x \in Desk$ có nghĩa là quyển sách x nằm trên bàn.

c) **Ký pháp-μ:**

Để xác định đối tượng duy nhất trong tập hợp thỏa điều kiện, ngôn ngữ Z dùng 1 ký pháp được gọi là **ký pháp-μ**, cụ thể như sau:

$$\mu x : A \mid p$$

và được đọc là: **"có duy nhất 1 giá trị x thuộc về tập A sao cho p"**.

Khi ta ký hiệu: $y = (\mu x : A \mid p)$ tức là y là giá trị duy nhất thuộc tập A sao cho p đúng.

Ví dụ 4: Ta nói "2 là số duy nhất thuộc về tập số tự nhiên thỏa 4+n=6"

$2 = (\mu n : N \mid 4 + n = 6)$

5.1.2. Lý thuyết tập hợp

Cũng tương tự như trong phần Logic toán học đã trình bày ở trên, các khái niệm trong lý thuyết tập hợp của ngôn ngữ Z hoàn toàn giống với ngôn ngữ toán học thông thường.

1. Tập hợp được biểu diễn dưới dạng liệt kê :

S == {a, b, c}

2. Tập hợp được biểu diễn dưới dạng vị từ :

S == {x :X | p(x)}

3. Tập hợp rỗng :

S == ∅

4. Để biểu diễn phát biểu : 'giá trị x thuộc về tập S', ta dùng ký hiệu :

$x \in S$

5. Để biểu diễn phát biểu : 'giá trị x không thuộc thuộc về tập S', ta dùng ký hiệu :

$x \notin S \Leftrightarrow \neg(x \in S)$

6. Các phép toán trên tập hợp

a.	Giao : $A \cap B = \{x :U \mid x \in A \wedge x \in B\}$
b.	Hợp : $A \cup B = \{x :U \mid x \in A \vee x \in B\}$
c.	Hiệu : $A \setminus B = \{x :U \mid x \in A \wedge x \notin B\}$
d.	Tập con : $A \subset B \Leftrightarrow \forall x \in A \Rightarrow x \in B$

e.	Tập chứa trong : $A \supset B \Leftrightarrow \forall x \in B \Rightarrow x \in A$
f.	Tập bằng nhau : $A = B \Leftrightarrow A \subset B \wedge B \subset A$
g.	$A \subseteq B \Leftrightarrow A \subset B \vee A = B$
h.	$A \supseteq B \Leftrightarrow A \supseteq B \vee A = B$

7. Tập tất cả các tập con của 1 tập hợp (Power set)

Ký hiệu : P X

Ví dụ : X == {1, 2}

P X = {∅, {1}, {2}, {1, 2}}

8. Tích Decartes của 2 hay nhiều tập hợp.

5.1.3. Hàm và quan hệ

5.1.3.1. Quan hệ 2 ngôi

Nếu X, Y là các tập hợp, ta ký hiệu : X↔Y là tập tất cả các quan hệ 2 ngôi giữa X và Y.

Ta có: X↔Y == P(X×Y)

Mỗi phần tử của X↔Y là một bộ có thứ tự (x, y) trong đó x được lấy từ tập X và y được lấy từ tập Y

5.1.3.2. Miền xác định và miền giá trị

Miền xác định của 1 quan hệ được ký hiệu là **dom,** là tập hợp tất cả các giá trị x thuộc về tập X sao cho x có ảnh y thuộc Y qua R.

dom R $= \{x \in X; y \in Y \mid x \mapsto y \in R \bullet x\}$

hay ta có thể biểu diễn theo 1 cách khác như sau:

dom R $= \{x : X \mid \exists y : Y \bullet x \mapsto y \in R\}$

Miền giá trị của quan hệ được ký hiệu là **ran,** là tập hợp tất cả các giá trị của y thuộc về tập Y sao cho y là ảnh của 1 giá trị x thuộc tập X qua R.

ran R $= \{x \in X; y \in Y \mid x \mapsto y \in R \bullet y\}$

hay ta có thể biểu diễn theo 1 cách khác như sau:

ran R $= \{y : Y \mid \exists x : X \bullet x \mapsto y \in R\}$

5.1.3.3. Hàm

Xét quan hệ R trên 2 tập hợp X và Y, nếu mỗi phần tử thuộc tập X có nhiều nhất 1 ảnh y thuộc tập Y qua R thì ta nói quan hệ R là 1 hàm đi từ tập X đến tập Y. Hàm được phân chia thành 2 loại chính: hàm riêng phần và hàm toàn phần.

5.1.3.4. Hàm riêng phần

Một hàm riêng phần từ tập X đến tập Y là 1 quan hệ trên tập X và tập Y, biến mỗi giá trị x∈X thành **nhiều nhất một** giá trị y∈Y. Ký hiệu:

$$X \to Y == \{f : X \leftrightarrow Y \mid \forall x : X; y_1, y_2 : Y \bullet x \mapsto y_1 \in f \land x \mapsto y_2 \in f \Rightarrow y_1 = y_2\}$$

5.1.3.5. Hàm toàn phần

$$X \to Y == \{f : X \to Y \bullet dom \;\; o = X\}$$

⤔	partial, injective functions
↣	total, injective functions
⤀	partial, surjective functions
↠	total, surjective functions
⤗	partial, bijective functions
⤖	total, bijective functions

5.2. Giản đồ (schemas)

5.2.1. Giới thiệu chung

Trong ngôn ngữ Z có 2 thành phần ngôn ngữ chính : ngôn ngữ toán và ngôn ngữ giản đồ. Ngôn ngữ toán học được sử dụng trong nhiều phần khác nhau của giai đoạn thiết kế : đặc tả các đối tượng, đặc tả các ràng buộc và mối quan hệ giữa chúng, v.v... Trong khi đó, ngôn ngữ giản đồ được dùng để tạo nên các bảng mô tả, kết hợp, đóng gói các phần thông tin khác nhau, đồng thời đặt tên cho chúng để sử dụng lại cho các mục đích khác.

Việc định nghĩa và đặt tên, cũng như khả năng tái sử dụng 1 thành phần, 1 đối tượng nào đó là hết sức cần thiết cho quá trình đặc tả. Nó giúp cho các đặc tả ngắn gọn, chính xác, tránh trùng lắp và rõ ràng, dễ hiểu.

5.2.2. Định nghĩa

Giản đồ (schema) là cú pháp của Z cho phép người đặc tả định nghĩa 1 khái niệm, 1 yếu tố mới gồm nhiều thành phần thông tin khác nhau, có ràng buộc với nhau, giống như khái niệm cấu trúc trong các ngôn ngữ lập trình; đồng thời đặt tên cho cấu trúc này.

Một giản đồ bao gồm 2 phần: phần khai báo các biến và phần vị từ diễn tả các ràng buộc trên những biến này. Ta có thể biểu diễn một giản đồ theo 1 trong 2 dạng sau :

- Theo chiều ngang :

  [khai báo | ràng buộc]

- Theo chiều dọc :

khai báo

ràng buộc

Ví dụ :

$$[seating : \mathbb{P}\ Seat;\ sold : Seat \twoheadrightarrow Customer$$
$$\mid \operatorname{dom} sold \subseteq seating]$$

hay :

$$seating : \mathbb{P}\ Seat;\ sold : Seat \twoheadrightarrow Customer$$

$$\operatorname{dom} sold \subseteq seating$$

5.2.3. Toán tử đặt tên

Nhằm mục đích đặt tên cho 1 giản đồ, tiện cho việc sử dụng lại sau này, ngôn ngữ Z cung cấp 1 toán tử riêng, được ký hiệu là : $\widehat{=}$, để thuận lợi cho việc soạn thảo, ký hiệu này được viết lại thành ^=

Tên ^= [khai báo | ràng buộc]

Tên

khai báo

ràng buộc

Ví dụ :

$$BoxOffice \mathrel{\widehat{=}} [seating : \mathbb{P}\ Seat;\ sold : Seat \twoheadrightarrow Customer \mid$$
$$\operatorname{dom} sold \subseteq seating]$$

hay :

BoxOffice

$$seating : \mathbb{P}\, Seat;\; sold : Seat \nrightarrow Customer$$

$$\mathrm{dom}\, sold \subseteq seating$$

5.2.4. Giản đồ tương đương

Hai giản đồ được gọi là tương đương nhau nếu chúng có cùng các biến và có cùng ràng buộc giống nhau trên các biến này.

Ví dụ : 2 giản đồ sau đây là tương đương :

$$seating : \mathbb{P}\, Seat;\; sold : Seat \nrightarrow Customer$$

$$\mathrm{dom}\, sold \subseteq seating$$

và

$$seating : \mathbb{P}\, Seat;\; sold : Seat \leftrightarrow Customer$$

$$\mathrm{dom}\, sold \subseteq seating \wedge sold \in Seat \nrightarrow Customer$$

5.2.5. Một số ghi chú

- ❖ Để giản đồ đơn giản và dễ đọc hơn, ta có thể thực hiện ngắt các khai báo ra thành nhiều dòng khác nhau, mỗi khai báo trên 1 dòng và giản lược các dấu chấm phẩy (;) ngăn cách. Khi đó phép nối giữa các dòng này mặc nhiên là phép nối liền - $\wedge$ (conjunction)

Ví dụ : giản đồ trong ví dụ ở phần trên

$$seating : \mathbb{P}\, Seat;\; sold : Seat \leftrightarrow Customer$$

$$\mathrm{dom}\, sold \subseteq seating \wedge sold \in Seat \nrightarrow Customer$$

có thể được biểu diễn lại như sau :

$$\begin{array}{|l}
seating : \mathbb{P}\, Seat \\
sold : Seat \leftrightarrow Customer \\
\hline
\mathrm{dom}\, sold \subseteq seating \\
sold \in Seat \rightarrowtail\!\!\!\!\!\rightarrow Customer
\end{array}$$

❖ Nếu 1 giản đồ đơn giản chỉ khai báo các biến mà không nêu ra ràng buộc cụ thể trên các biến này thì ta có thể bỏ phần vị từ bên dưới đi. Ví dụ :

$$\begin{array}{|l}
stalls : \mathbb{P}\, Seat
\end{array}$$

Khi đó, giản đồ sẽ tương đương với giản đồ sau :

$$\begin{array}{|l}
stalls : \mathbb{P}\, Seat \\
\hline
true
\end{array}$$

5.2.6. Giản đồ được sử dụng như 1 kiểu dữ liệu

Trong ngôn ngữ Z, một kiểu dữ liệu có thể là 1 tập cho trước (N, Z, Q, R, ...), 1 tập Power set, 1 kiểu tự do hoặc là 1 tích Decartes của nhiều tập hợp. Ngoài ra, ta có thể sử dụng giản đồ như 1 kiểu dữ liệu, sau khi đã đặt tên cho giản đồ đó bằng phép toán đặt tên ^=.

Ví dụ ta có giản đồ :

$$\begin{array}{|l}
\multicolumn{1}{l}{SchemaOne} \\
a : \mathbb{Z} \\
c : \mathbb{P}\, \mathbb{Z}
\end{array}$$

Như vậy khi khai báo :

s : SchemaOne

tức là s là một biến, 1 phần tử thuộc kiểu SchemaOne, s sẽ bao gồm 2 thành phần: a là 1 số nguyên và c là 1 tập con gồm các số nguyên.

Ví dụ 2 : Ta định nghĩa kiểu dữ liệu Date, bao gồm 2 thông tin là (tháng, ngày) như sau :

Trước hết, ta có định nghĩa 1 kiểu dữ liệu tự do để mô tả các tháng, bao gồm 12 giá trị hằng riêng lẻ như sau :

$$\begin{aligned} Month ::= {} & jan \mid feb \mid mar \mid apr \mid may \mid jun \mid jul \mid aug \mid sep \mid \\ & oct \mid nov \mid dec \end{aligned}$$

Sau đó ta định nghĩa kiểu Date :

$$\begin{array}{|l} \textit{Date} \\ \hline month : Month \\ day : 1 \mathrel{..} 31 \\ \hline month \in \{sep, apr, jun, nov\} \Rightarrow day \leq 30 \\ month = feb \Rightarrow day \leq 29 \\ \hline \end{array}$$

5.2.7. Giản đồ được sử dụng trong các khai báo

Một giản đồ, sau khi được định nghĩa và đặt tên, có thể được sử dụng trong bất kỳ 1 khai báo nào. Kết quả là các biến có trong giản đồ sẽ có mặt trong khai báo đó, cùng với những ràng buộc tương ứng đã quy định trên giản đồ.

Xét ví dụ sau, với giản đồ SchemaTwo, gần giống với SchemaOne nhưng có thêm 1 số ràng buộc, tập c phải khác rỗng, đồng thời a phải là 1 phần tử của c.

$$\begin{array}{|l} \textit{SchemaTwo} \\ \hline a : \mathbb{Z} \\ c : \mathbb{P}\,\mathbb{Z} \\ \hline a \in c \land c \neq \emptyset \\ \hline \end{array}$$

Khi đó tập hợp :

$$\{\, SchemaTwo \mid a = 0 \bullet c \,\}$$

sẽ bao gồm tất cả các tập con $c \subset Z$ và c có chứa phần tử số 0. Ta có thể biểu diễn tập hợp này theo cách thông thường, không sử dụng giản đồ như sau :

$$\{ a : \mathbb{Z};\ c : \mathbb{P}\mathbb{Z} \mid a \in c \land c \neq \varnothing \land a = 0 \bullet c \}$$

hoặc cũng có thể biểu diễn theo 1 cách khác, sử dụng 1 đối tượng thuộc kiểu SchemaTwo :

$$\{ s : SchemaTwo \mid s.a = 0 \bullet s.c \}$$

Ví dụ 2 : Trở lại giản đồ mô tả kiểu Date ở trên :

$$\begin{array}{|l}
\textit{Date} \\
\hline
month : Month \\
day : 1 \mathrel{..} 31 \\
\hline
month \in \{sep, apr, jun, nov\} \Rightarrow day \leq 30 \\
month = feb \Rightarrow day \leq 29 \\
\hline
\end{array}$$

Nếu ta có thể định nghĩa tập các tháng có 31 ngày như sau :

$$\{ Date \mid day = 31 \bullet month \}$$

hay nói 1 cách khác, đó là tập :

$$\{jan, mar, may, jul, aug, oct, dec\}$$

5.2.8. Giản đồ được sử dụng trong các biểu thức lượng từ

Như đã khẳng định, một giản đồ có thể được sử dụng như 1 kiểu dữ liệu nên nó có thể xuất hiện trong các biểu thức lượng từ. Ví dụ :

$$\exists\, SchemaTwo \bullet a = 0 \iff \exists\, a : \mathbb{Z};\ c : \mathbb{P}\mathbb{Z} \mid c \neq \varnothing \land a \in c \bullet a = 0$$

hay

$$\forall\, SchemaTwo \bullet a \in c$$

$$\exists\, Date \bullet month = feb \land day = 29$$

Vị từ

$$\forall\, SchemaTwo \bullet a \in c$$

còn có thể được phát biểu theo 1 cách khác như sau :

$$\forall\, s : SchemaTwo \bullet s.a \in s.c$$

5.2.9. Giản đồ được sử dụng như 1 vị từ

Xét giản đồ SchemaThree như sau :

$$
\begin{array}{|l}
\textit{SchemaThree} \\
\hline
a : \mathbb{Z} \\
c : \mathbb{P}\,\mathbb{Z} \\
\hline
c \neq \varnothing \land a \in c \\
c \subseteq \{0, 1\} \\
\hline
\end{array}
$$

Ta có thể phát biểu :

$$\forall\, a : \mathbb{Z};\ c : \mathbb{P}\,\mathbb{Z} \mid SchemaThree \bullet SchemaTwo$$

để diễn tả rằng bất kỳ số nguyên a và tập số c nào thỏa ràng buộc trong SchemaThree thì cũng thỏa ràng buộc trong SchemaTwo.

Phát biểu trên tương đương với phát biểu sau :

$$\forall\, a : \mathbb{Z};\ c : \mathbb{P}\,\mathbb{Z} \mid c \notin \varnothing \land a \in c \land c \subseteq \{0, 1\} \bullet c \notin \varnothing \land a \in c$$

Như vậy, ta nhận thấy toàn bộ phần khai báo của 2 giản đồ đã được lược bỏ, chỉ có phần ràng buộc là được giữ lại.

5.2.10.Dạng chuẩn của 1 giản đồ

Khi định nghĩa 1 giản đồ, phần khai báo của giản đồ bản thân nó đã chứa sẵn 1 số ràng buộc nhất định. Chính vì vậy, khi sử dụng giản đồ như 1 vị từ thì các ràng buộc này cũng dễ bị "bỏ quên", do phần khai báo của giản đồ đã được giản lược (xem phần trên).

Vì vậy, để tránh gây nhầm lẫn, ta nên chuyển giản đồ về một dạng mới sao cho tất cả các ràng buộc đều nằm ở phần ràng buộc bên dưới. Một giản đồ theo dạng này được gọi là giản đồ chuẩn, và thao tác chuyển toàn bộ ràng buộc xuống phần dưới của giản đồ được gọi là chuẩn hóa giản đồ.

Ví dụ : Xét giản đồ sau :

SchemaFour

$a : \mathbb{N}$

$c : \mathbb{P}\,\mathbb{N}$

$a \in c \land c \neq \varnothing$

Thoạt nhìn, ta có thể nghĩ giản đồ này tương đương với giản đồ SchemaTwo, tuy nhiên, ngay trên bản thân kiểu số tự nhiên N đã có ràng buộc rồi. Do vậy, ta chuẩn hóa nó lại như sau :

SchemaFourNormalised

$a : \mathbb{Z}$

$c : \mathbb{P}\,\mathbb{Z}$

$a \in \mathbb{N}$

$c \in \mathbb{P}\,\mathbb{N}$

$a \in c \land c \neq \varnothing$

Ví dụ 2 : Xét giản đồ Date ở trên :

$$\begin{array}{|l}
\textit{Date} \\
\hline
month : Month \\
day : 1 \,..\, 31 \\
\hline
month \in \{sep, apr, jun, nov\} \Rightarrow day \leq 30 \\
month = feb \Rightarrow day \leq 29 \\
\hline
\end{array}$$

Ta chuẩn hóa giản đồ này lại như sau :

$$\begin{array}{|l}
\textit{DateNormalised} \\
\hline
month : Month \\
day : \mathbb{Z} \\
\hline
day \in 1 \,..\, 31 \\
month \in \{sep, apr, jun, nov\} \Rightarrow day \leq 30 \\
month = feb \Rightarrow day \leq 29 \\
\hline
\end{array}$$

5.2.11.Đặt lại tên các thành phần trong giản đồ

Trong quá trình đặc tả, đôi khi cần phải đặt lại tên các thành phần trong 1 giản đồ. Bằng cách này ta có thể tạo nên 1 giản đồ mới có cùng kết cấu và ràng buộc với giản đồ cũ nhưng có các thành phần với tên và ý nghĩa khác đi.

Cú pháp để đổi tên 1 giản đồ :

Nếu Schema là 1 giản đồ, ta ký hiệu :

$$\boxed{Schema[new/old]}$$

là giản đồ được tạo nên bằng cách đổi tên thành phần tên là old trong Schema thành new.

Ví dụ : trong giản đồ SchemaTwo, nếu thay a bằng q và c bằng s, ta có :

$$SchemaTwo[q/a, s/c]$$

khi đó, giản đồ kết quả sẽ tương đương với giản đồ :

$$\begin{array}{|l}
q : \mathbb{Z} \\
s : \mathbb{P}\,\mathbb{Z} \\
\hline
s \neq \varnothing \wedge q \in s
\end{array}$$

Tương tự như vậy, nếu muốn định nghĩa kiểu dữ liệu StartDate từ kiểu dữ liệu Date, ta có thể làm như sau :

$$StartDate \mathrel{\widehat{=}} Date[start\_month / month, start\_day / day]$$

Khi đó, giản đồ sẽ tương đương :

$$\begin{array}{|l}
\multicolumn{1}{l}{StartDate} \\
\hline
start_month : Month \\
start_day : 1\,..\,31 \\
\hline
start_month \in \{sep, apr, jun, nov\} \Rightarrow start_day \leq 30 \\
start_month = feb \Rightarrow start_day \leq 29 \\
\hline
\end{array}$$

Nếu ta định nghĩa thêm giản đồ FinishDate :

$$FinishDate \mathrel{\widehat{=}} Date[finish\_month / month, finish\_day / day]$$

Khi đó, StartDate và FinishDate được xem như 2 kiểu dữ liệu khác nhau, mặc dù chúng đều có 2 thành phần và có các ràng buộc tương tự như nhau.

5.3. Các phép toán trên giản đồ

5.3.1. Phép nối liền (Conjunction)

Cho 2 giản đồ S và T được thể hiện như sau, với P và Q là 2 vị từ diễn tả các ràng buộc lên các biến tương ứng:

$$\begin{array}{|l}
\multicolumn{1}{l}{S} \\
\hline
a : A \\
b : B \\
\hline
P \\
\hline
\end{array}
\qquad\qquad
\begin{array}{|l}
\multicolumn{1}{l}{T} \\
\hline
b : B \\
c : C \\
\hline
Q \\
\hline
\end{array}$$

Ta ký hiệu :

$S \wedge T$

để biểu diễn phép nối liền của 2 giản đồ S và T. Kết quả sẽ là 1 giản đồ mới được tạo nên bằng cách gộp các biến trong S và T lại, đồng thời thực hiện phép nối liền trên 2 vị từ ràng buộc P và Q.

$$
\begin{array}{|l}
\hline
a : A \\
b : B \\
c : C \\
\hline
P \wedge Q \\
\hline
\end{array}
$$

Lưu ý : Nếu trong 2 giản đồ S và T có 1 thành phần giống nhau thì thành phần này phải có kiểu giống nhau ở cả 2 giản đồ. Nếu không giản đồ $S \wedge T$ sẽ không xác định.

5.3.2. Phép đổi tên (Decoration)

Giả sử ta có giản đồ State mô tả 1 trạng thái của hệ thống, có 2 thành phần a, b với ràng buộc là P

$$
\begin{array}{|l}
\hline
\textit{State} \\
a : A \\
b : B \\
\hline
P \\
\hline
\end{array}
$$

Mỗi đối tượng s : State biểu diễn 1 trạng thái hợp lệ của hệ thống.

Để biểu diễn 1 thao tác lên 1 trạng thái nào đó, ta cần sử dụng 2 trạng thái : trạng thái trước khi thực hiện thao tác và trạng thái sau khi thực hiện thao tác đó. Để phân biệt 2 trạng thái này, ta dùng phép đặt tên để tạo nên 1 giản đồ mới, bằng cách thêm dấu nháy (’) vào sau tên giản đồ cũng như tên tất cả các biến được khai báo trong giản đồ :

$State'$
$a' : A$
$b' : B$
$P[a'/a, b'/b]$

Để đặc tả thao tác, ta có thể đưa cả 2 trạng thái State và State' vào phần khai báo của giản đồ. Một giản đồ thao tác sẽ có dạng :

$Operation$
$State$
$State'$
$\ldots$

Ngôn ngữ Z quy định 1 cách thức biểu diễn cả 2 giản đồ State và State', cách thức biểu diễn đó như sau :

$\Delta Schema$
$Schema$
$Schema'$

5.3.3. Phép nối rời (Disjunction)

Cho 2 giản đồ S và T được thể hiện như sau, với P và Q là 2 vị từ diễn tả các ràng buộc lên các biến tương ứng:

S
$a : A$
$b : B$
P

T
$b : B$
$c : C$
Q

Ta ký hiệu :

S ∨ T

để biểu diễn phép nối rời của 2 giản đồ S và T. Kết quả sẽ là 1 giản đồ mới được tạo nên bằng cách gộp các biến trong S và T lại, đồng thời thực hiện phép nối rời trên 2 vị từ ràng buộc P và Q.

$a : A$
$b : B$
$c : C$

$P \vee Q$

Tương tự như trên phép nối liền, tất cả các thành phần (các biến) xuất hiện ở cả 2 giản đồ thành phần phải có cùng kiểu dữ liệu. Tất cả các ràng buộc nếu có xuất hiện ở phần khai báo đều phải được chuyển sang phần vị từ ràng buộc của mỗi giản đồ thành phần trước khi thực hiện phép nối rời.

5.3.4. Phép phủ định (Negation)

Cho giản đồ S T được thể hiện như sau, với P là vị từ diễn tả các ràng buộc lên các biến tương ứng:

S
$a : A$
$b : B$

P

khi đó giản đồ phủ định ¬S sẽ có dạng :

$a : A$
$b : B$

$\neg P$

Lưu ý : phép phủ định chỉ áp dụng cho các giản đồ đã được chuẩn hóa (tức là tất cả các ràng buộc đều phải được chuyển sang phần vị từ ràng buộc bên dưới).

Ví dụ : Giản đồ Date biểu diễn những ngày tháng hợp lệ, do đó $\neg$Date sẽ biểu diễn tất cả những cặp (month, day) nào không phải là một ngày tháng hợp lệ.

$$
\begin{array}{|l}
month : Month \\
day : \mathbb{Z} \\
\hline
day \notin 1\,..\,31 \vee \\
(month \in \{sep, apr, jun, nov\} \wedge day > 30) \vee \\
(month = feb \wedge day > 29)
\end{array}
$$

5.3.5. Phép lượng từ hóa

Ta có thể lượng từ hóa 1 số thành phần của 1 giản đồ để tạo ra 1 giản đồ mới. Nếu Q là 1 lượng từ và dec là phần khai báo, khi đó giản đồ được lượng từ hóa sẽ có dạng :

$$Q\ dec \bullet Schema$$

Giản đồ này được tạo thành bằng cách bỏ đi các thành phần có trong phần khai báo dec và lượng từ hóa chúng bằng lượng từ Q trong phần vị từ ràng buộc bên dưới.

Ví dụ : Cho giản đồ như sau :

$$
\begin{array}{|l}
\multicolumn{1}{l}{S} \\
a : A \\
b : B \\
\hline
P
\end{array}
$$

Khi đó, $\forall b : B \bullet S$ là giản đồ :

$$
\begin{array}{|l}
a : A \\
\hline
\forall\, b : B \bullet P
\end{array}
$$

và $\exists b : B \bullet S$ là giản đồ :

$a : A$

$\exists\, b : B \bullet P$