LỜI NÓI ĐẦU

Trong các năm qua, nhiều tài liệu của ngành công nghệ thông tin đã được giới thiệu nhiều cho các cán bộ nghiên cứu, ứng dụng và sinh viên ở bậc đại học. Tuy nhiên các giáo trình của ngành học này chưa đáp ứng dược nhu cầu của sinh viên các trường đại học, đặc biệt đối với sinh viên khu vực miền Trung.

Vì vậy, chúng tôi biên soạn giáo trình "Trí tuệ nhân tạo", một môn cơ sở chuyên ngành trong chương trình đào tạo Cử nhân Tin học, ngoài mục đích xây dựng nhiều giáo trình trên một khung chương trình đào tạo, mà còn giúp cho sinh viên có tài liệu học tập phù hợp với hoàn cảnh thực tế của Đại học Huế.

Trong cuốn sách này, sinh viên được làm quen với một số kiến thức cơ bản nhất về các phương pháp tìm kiếm lời giải và các phương pháp xử lý tri thức. Ngoài ra, cuốn sách cũng giới thiệu một số chương trình cài đặt, nhằm giúp sinh viên có thể hiểu một cách tường tận các giải thuật, đồng thời tin tưởng rằng các giải thuật này có thể áp dung thực tế và cài đặt được trên máy tính một cách dễ dàng.

Các nội dung trình bày trong cuốn sách đã từng được giảng cho sinh viên ngành Công nghệ Thông tin tại Đại học Huế trong những năm vừa qua.
Cuốn sách ra đời dưới sự giúp đỡ về mặt vật chất cũng như tinh thần của Đại học Huế, Trường Đại học Khoa học và đặc biệt là Ban chủ nhiệm Khoa Công nghệ Thông tin và các đồng nghiệp thuộc Bộ môn Khoa học Máy tính. Chúng tôi xin gửi tới họ lòng biết ơn. Xin chân thành cám ơn các bạn bè đã cổ cũ và gíup cho cuốn sách sớm được hoàn thành.

Mặc dù đã hết sức cố gắng, tuy nhiên cuốn sách cũng không tránh khỏi những thiếu sót. Chúng tôi rất mong được sự góp ý của các độc giả, đặc biệt đối với các đồng nghiệp và sinh viên để cuốn sách ngày càng hoàn thiện.

Huế, tháng 7 năm 2004
Tác giả

MỤC LỤC

Chương 0 MỞ ĐẦU

1. Tổng quan về khoa học Trí tuệ nhân tạo.

Trong Công Nghệ Thông Tin, Trí Tuệ Nhân Tạo (Artificial Intelligence) là một ngành mới, nhưng phát triển rất mạnh mẽ và đem lại nhiều kết quả to lớn. Con người thường tự cho mình là sinh vật thông minh vì khả năng trí tuệ đóng vai trò quan trong trong cuộc sống. Trong văn học cũng đã từng có những câu chuyện đề cao về trí thông minh của con người.

Trí Tuệ Nhân Tạo chỉ mới hình thành từ năm 1956. Tuy nhiên, việc nghiên cứu trí tuệ đã có từ lâu. Trên 2000 năm trước, các nhà triết học đã tìm hiểu về cách thức nhìn nhận, học tập, nhớ và suy lý. Việc ra đời của máy tính điện tử vào những năm 50 của thế kỷ 20 đã sinh ra khuynh hướng đưa các lĩnh vực nghiên cứu trí tuệ về các vấn đề lý thuyết và thực nghiệm trên máy.

1.1. Đối tượng và mục tiêu nghiên cứu của trí tuệ nhân tạo.

Trí tuệ nhân tạo nghiên cứu về cách hành xử thông minh (intelligent behaviour) với mục tiêu là xây dựng lý thuyết đầy đủ về thông minh để có thể giải thích được hoạt động thông minh của sinh vật và áp dụng được các hiểu biết vào các máy móc nói chung, nhằm phục vụ cho con người.

- **Về mặt kỹ thuật**: Tạo ra các máy thông minh để giải quyết vấn đề thực tế dùng các kỹ thuật AI.

- **Khoa học:** Phát triển các khái niệm và thuật ngữ để hiểu được các hành xử thông minh của sinh vật.

1.2. Vai trò của Trí Tuệ Nhân Tạo.

Trí tuệ nhân tạo bao quát rất nhiều lĩnh vực nghiên cứu hẹp. Nó nghiên cứu từ các lĩnh vực tổng quát như máy nhận biết, suy luận logic, đến các bài toán như chơi cờ, chứng minh định lý. Thường thì các nhà khoa học ở các lĩnh vực

khác tìm đến với trí tuệ nhân tạo ở các kỹ thuật hệ thống hoá và tự động hoá các xử lý tri thức cũng như các phương pháp thuộc lĩnh vực mang tính người.

Trí tuệ nhân tạo nghiên cứu kỹ thuật làm cho máy tính có thể "suy nghĩ một cách thông minh" và mô phỏng quá trình suy nghĩ của con người khi đưa ra những quyết định, lời giải. Trên cơ sở đó, thiết kế các chương trình cho máy tính để giải quyết bài toán.

Sự ra đời và phát triển của Trí tuệ nhân tạo đã tạo ra một bước nhảy vọt về chất trong kỹ thuật và kỹ nghệ xử lý thông tin. Trí tuệ nhân tạo chính là cơ sở của công nghệ xử lý thông tin mới, độc lập với công nghệ xử lý thông tin truyền thống dựa trên văn bản giấy tờ. Điều này được thể hiện qua các mặt sau:

- Nhờ những công cụ hình thức hoá (các mô hinh logic ngôn ngữ, logic mờ,...), các tri thức thủ tục và tri thức mô tả có thể biểu diễn được trong máy. Do vậy quá trình giải bài toán được tiến hành hữu hiệu hơn.
- Mô hình logic ngôn ngữ đã mở rộng khả năng ứng dụng của máy tính trong lĩnh vực đòi hỏi tri thức chuyên gia ở trình độ cao, rất khó như: y học, sinh học, địa lý, tự động hóa.
- Một số phần mềm trí tuệ nhân tạo thể hiện tính thích nghi và tính mềm dẻo đối với các lớp bài toán thuộc nhiều lĩnh vực khác nhau.
- Khi máy tính được trang bị các phần mềm trí tuệ nhân tạo ghép mạng sẽ cho phép giải quyết những bài toán cỡ lớn và phân tán.

1.3. Các kỹ thuật Trí tuệ nhân tạo.

Có nhiều kỹ thuật nghiên cứu, phát triển ngành khoa học Trí tuệ nhân tạo. Tuy vậy, các kỹ thuật Trí tuệ nhân tạo thường khá phức tạp khi cài đặt cụ thể, lý do là các kỹ thuật này thiên về xử lý các ký hiệu tượng trưng và đòi hỏi phải sử dụng những tri thức chuyên môn thuộc nhiều lĩnh vực khác nhau.

Do vậy, các kỹ thuật Trí tuệ nhân tạo hướng tới khai thác những tri thức về lĩnh vực đang quan tâm được mã hoá trong máy sao cho đạt được mức độ tổng quát; dễ hiểu, dễ diễn đạt thông qua ngôn ngữ chuyên môn gần gũi với ngôn ngữ

tự nhiên; dễ sửa đổi, hiệu chỉnh, dễ sử dụng, khai thác nhằm thu hẹp các khả năng cần xét để đi tới lời giải cuối cùng.

Các kỹ thuật Trí tuệ nhân tạo cơ bản bao gồm :

- **Lý thuyết giải bài toán và suy diễn thông minh:** Lý thuyết giải bài toán cho phép viết các chương trình giải câu đố, chơi các trò chơi thông qua các suy luận mang tính người; các hệ thống chứng minh định lý. Ngoài ra các hệ thống hỏi đáp thông minh còn cho phép lưu trữ và xử lý khối lượng lớn các thông tin.
- **Lý thuyết tìm kiếm may rủi:** Lý thuyết này bao gồm các phương pháp và kỹ thuật tìm kiếm với sự hỗ trợ của thông tin phụ để giải bài toán một cách có hiệu quả.
- **Các ngôn ngữ về Trí tuệ nhân tạo:** Để xử lý các tri thức người ta không chỉ sử dụng các ngôn ngữ lập trình dùng cho các xử lý dữ liệu số, mà cần có ngôn ngữ khác. Các ngôn ngữ chuyên dụng này cho phép lưu trữ và xử lý thông tin ký hiệu. Một số ngôn ngữ được nhiều người biết đến là IPL.V,LISP, PROLOG.
- **Lý thuyết thể hiện tri thức và hệ chuyên gia:** Trí tuệ nhân tạo là khoa học về thể hiện và sử dụng tri thức. Mạng ngữ nghĩa, lược đồ, logic vị từ, khung là các phương pháp thể hiện tri thức thông dụng. Việc gắn liền cách thể hiện và sử dụng tri thức là cơ sở hình thành hệ chuyên gia.
- **Lý thuyết nhận dạng và xử lý tiếng nói:** Giai đoạn phát triển đầu của Trí tuệ nhân tạo gắn với lý thuyết nhận dạng. Các phương pháp nhận dạng chính gồm: nhận dạng hình học, nhận dạng dùng tâm lý học, nhận dạng theo phương pháp hàm thế, dùng máy nhận dạng. ứng dụng của phương pháp này trong việc nhận dạng chữ viết, âm thanh.
- **Người máy:** Cuối những năm 70, người máy trong công nghiệp đã đạt được nhiều tiến bộ. Người máy có bộ phận cảm nhận và các cơ chế hoạt

động được nối ghép theo sự điều khiển thông minh. Khoa học về cơ học và Trí tuệ nhân tạo được tích hợp trong khoa học người máy.

- **Tâm lý học xử lý thông tin :** Các kết quả nghiên cứu của tâm lý học giúp Trí tuệ nhân tạo xây dựng các cơ chế trả lời theo hành vi, có ý thức; nó giúp cho việc thực hiện các suy diễn mang tính người.
- Ngoài ra, **xử lý danh sách, kỹ thuật đệ quy, kỹ thuật quay lui và xử lý cú pháp hình thức** là những kỹ thuật cơ bản của tin học truyền thống có liên quan trực tiếp đến Trí tuệ nhân tạo.

2. Lịch sử phát triển của Trí Tuệ Nhân Tạo.

Lịch sử của Trí tuệ nhân tạo cho thấy ngành khoa học này có nhiều kết quả đáng ghi nhận. Theo các mốc phát triển, người ta thấy Trí tuệ nhân tạo được sinh ra từ những năm 50 với các sự kiện sau:

- Turing được coi là người khai sinh ngành Trí tuệ nhân tạo bởi phát hiện của ông về máy tính có thể lưu trữ chương trình và dữ liệu.
- Tháng 8/1956 J.Mc Carthy, M. Minsky, A. Newell, Shannon. Simon ,… đưa ra khái niêm "trí tuệ nhân tạo".
- Vào khoảng năm 1960 tại Đại học MIT (Massachussets Institure of Technology) ngôn ngữ LISP ra đời, phù hợp với các nhu cầu xử lý đặc trưng của trí tuệ nhân tạo - đó là ngôn ngữ lập trình đầu tiên dùng cho trí tuệ nhân tạo.
- Thuật ngữ Trí tuệ nhân tạo được dùng đầu tiên vào năm 1961 cũng tại MIT.
- Những năm 60 là giai đoạn lạc quan cao độ về khả năng làm cho máy tính biết suy nghĩ. Trong giai đoạn này người ta đã được chứng kiến máy chơi cờ đầu tiên và các chương trình chứng minh định lý tự động.

Cụ thể: 1961: Chương trình tính tích phân bất định

1963: Các chương trình Heuristics: Chương trình chứng minh các định lý hình học không gian có tên là "tương tự", chương trình chơi cờ của Samuel.

1964: Chương trình giải phương trình đại số sơ cấp, chương trình trợ giúp ELIZA (có khả năng làm việc giống như một chuyên gia phân tich tâm lý).

1966: Chương trình phân tích và tổng hợp tiếng nói

1968: Chương trình điều khiển người máy (Robot) theo đồ án "Mát – tay", chương trình học nói.

- Vào những năm 60, do giới hạn khả năng của các thiết bị, bộ nhớ và đặc biệt là yếu tố thời gian thực hiện nên có sự khó khăn trong việc tổng quát hoá các kết quả cụ thể vào trong một chương trình mềm dẻo thông minh.
- Vào những năm 70, máy tính với bộ nhớ lớn và tốc độ tính toán nhanh nhưng các phương pháp tiếp cận Trí tuệ nhân tạo cũ vẫn thất bại (do sự bùng nổ tổ hợp trong quá trình tìm kiếm lời giải các bài toán đặt ra).
- Vào cuối những năm 70 một vài kết quả như xử lý ngôn ngữ tự nhiên, biểu diễn tri thức và giải quyết vấn đề. Những kết quả đó đã tạo điều kiện cho sản phẩm thương mại đầu tiên của Trí tuệ nhân tạo ra đời đó là Hệ chuyên gia, được đem áp dụng trong các lĩnh vực khác nhau (Hệ chuyên gia là một phần mềm máy tính chứa các thông tin và tri thức về một lĩnh vực cụ thể nào đó, có khả năng giải quyết những yêu cầu của người sử dụng trong một mức độ nào đó, ở một trình độ như một chuyên gia con người có kinh nghiệm khá lâu năm).
- Một sự kiện quan trọng vào những năm 70 là sự ra đời ngôn ngữ Prolog, tương tự LISP nhưng nó có cơ sở dữ liệu đi kèm.

- Vào những năm 80, thị trường các sản phẩm dân dụng đã có khá nhiều sản phẩm ở trình đô cao như: máy giặt, máy ảnh,... sử dụng Trí tuệ nhân tạo. Các hệ thống nhận dạng và xử lý ảnh, tiếng nói.
- Những năm 90, các nghiên cứu nhằm vào cài đặt thành phần thông minh trong các hệ thống thông tin, gọi chung là cài đặt trí tuệ nhân tạo, làm rõ hơn các ngành của khoa học Trí tuệ nhân tạo và tiến hành các nghiên cứu mới, đặc biệt là nghiên cứu về cơ chế suy lý, về Trí tuệ nhân tạo phân tạo, về các mô hình tương tác.

Những đặc trưng của Trí tuệ nhân tạo

- Trí tuệ nhân tạo xử lý thông tin theo trật tự ký hiệu. Các thông tin gồm: khái niệm, luật, các đối tượng ? dùng cho suy lý. Khái niệm cơ bản trong Trí tuệ nhân tạo là sự thể hiện, suy lý, nhận biết, việc học và hệ thống cơ sở tri thức.
- Phương pháp may rủi hay được dùng trong Trí tuệ nhân tạo. Phương pháp này cho phép giải hai lớp bài toán khó. Thứ nhất là những bài toán chưa có thuật giải (bài toán nhận biết, ra quyết định). Thứ hai là các bài toán đã có thuật giải nhưng độ phức tạp lớn (chẳng hạn bài toán chơi cờ).
- Trí tuệ nhân tạo xét đến những thông tin không đầy đủ, không chính xác, có vẻ mâu thuẫn. Tuy vậy, các kết quả của Trí tuệ nhân tạo là cụ thể.
- Việc tương tác người- máy đi đôi với nhận biết tự động là cần thiết trong Trí tuệ nhân tạo. Các bài toán nhận dạng là ví dụ về yêu cầu này.
- Trí tuệ nhân tạo liên quan đến nhiều lĩnh vực, như các kỹ thuật mới, logic học, khoa học nhận biết, ngôn ngữ học, khoa học về tổ chức, thần kinh học. Trí tuệ nhân tạo còn nằm trong các lĩnh vực nghiên cứu nâng cao, các đề án nghiên cứu quan trọng.

3. Một số vấn đề Trí tuệ Nhân tạo quan tâm.

Những vấn đề chung

Khoa học Trí tuệ nhân tạo liên quan đến cảm giác, tri giác và cả quá trình tư duy thông qua các hành vi, giao tiếp. Nó có các định hướng nghiên cứu, ứng dụng sau:

1- Tìm và nghiên cứu các thủ tục giúp con người tiến hành các hoạt động sáng tạo. Công việc sáng tạo được thực hiện trên mô hình theo cấu trúc, chức năng và sử dụng công nghệ thông tin.

2- Dùng ngôn ngữ tự nhiên. Trước hết là ngôn ngữ được dùng để thể hiện tri thức, tiếp thu và chuyển hoá sang dạng có thể xử lý được.

3- Hình thức hoá các khía cạnh, các hành vi liên quan đến Trí tuệ nhân tạo. Do vậy có thể xây dựng các bài toán mang tính người và thông minh.

Các hoạt động lớn trong Trí tuệ nhân tạo bao gồm: chứng minh định lý, xử lý ngôn ngữ tự nhiên, hiểu tiếng nói, phân tích ảnh và hình, người máy và hệ chuyên gia. Về cài đặt hệ thống, khuynh hướng hiện tại của Trí tuệ nhân tạo là cài đặt các hệ Trí tuệ nhân tạo trong các hệ thống khác, đặc biệt là trong các hệ thống tin học.

Những vấn đề chưa được giải quyết trong Trí tuệ nhân tạo

Những thành tựu nghiên cứu và ứng dụng các kỹ thuật Trí tuệ nhân tạo đã khẳng định tính thực tiễn của các dự án xây dựng máy tính có khả năng suy nghĩ. Tuy vậy trong một số phạm vi, **máy tính còn thua xa so với hoạt động của hệ thần kinh con người:**

Sự khác nhau trong hoạt động giữa máy tính và bộ não con người, điều này thể hiện ưu thế của máy tính so với bộ não người vì khả năng tính toán rất lớn (nhất là trong các chương trình xử lý dữ liệu lớn).

Xử lý song song: mặc dù công nghệ điện tử hiện đại cho phép xây dựng các bộ đa xử lý, song máy tính không thể hoạt động song song như bộ não con người được.

Khả năng diễn giải: con người có thể xem xét cùng một vấn đề theo những phương pháp khác nhau, từ đó diễn giải theo cách dễ hiểu nhất. Ngược lại, sự linh hoạt này không thể mô phỏng được trong các hệ thống Trí tuệ nhân tạo.

Lôgic rời rạc và tính liên tục: một thách đố lớn với các hệ thống Trí tuệ nhân tạo là khả năng kết hợp các phương pháp xử lý thông tin trong môi trường liên tục với các thao tác xử lý thông tin rời rạc.

Khả năng học: mặc dù hiện nay máy tính có nhiều tính năng cao nhưng cũng không thể mô phỏng được hoàn toàn khả năng học giống bộ não con người.

Khả năng tự tổ chức: cho tới nay, người ta chưa thể tạo lập được các hệ thống Trí tuệ nhân tạo có khả năng tự tổ chức, tự điều khiển hoạt động của nó để thích nghi với môi trường.

Những vấn đề đặt ra trong tương lai của Trí tuệ nhân tạo.

Trong tương lai, những nghiên cứu và ứng dụng của Trí tuệ nhân tạo tập trung vào các vấn đề lớn sau:

Nghiên cứu và thử nghiệm các mạng Neuron, các hệ thống Trí tuệ nhân tạo mô phỏng chức năng hoạt động của bộ não với các khả năng học, tự tổ chức, tự thích nghi, tổng quát hoá, xử lý song song, có khả năng diễn giải, xử lý thông tin liên tục và rời rạc.

Nghiên cứu và tạo lập các hệ thống có giao tiếp thân thiện giữa người và máy trên cơ sở nghiên cứu nhận thức máy, thu thập và xử lý tri thức, xử lý thông tin hình ảnh, tiếng nói.

Nghiên cứu các phương pháp biểu diễn tri thức và các phương pháp suy diễn thông minh, các phương pháp giải quyết vấn đề đối với những bài toán phụ thuộc không gian, thời gian.

Ngày nay, thế giới đang chuyển mình trong những nghiên cứu về Trí tuệ nhân tạo. Chắc chắn rằng máy tính với trí tuệ như con người sẽ tác động mạnh đến cuộc sống xã hội.

4. Các khái niệm cơ bản:

Trí tuệ con người (Human Intelligence): Cho đến nay có hai khái niệm về trí tuệ con người được chấp nhận và sử dụng nhiều nhất, đó là:

- Khái niệm trí tuệ theo quan điểm của Turing

"Trí tuệ là những gì có thể đánh giá được thông qua các trắc nghiệm thông minh"

- Khái niệm trí tuệ đưa ra trong tụ điển bách khoa toàn thư:

"Trí tuệ là khả năng:

Phản ứng một cách thích hợp những tình huống mới thông qua hiệu chỉnh hành vi một cách thích đáng.

Hiểu rõ những mối liên hệ qua lại của các sự kiện của thế giới bên ngoài nhằm đưa ra những hành động phù hợp đạt tới một mục đích nào đó.

Những nghiên cứu các chuyên gia tâm lý học nhận thức chỉ ra rằng quá trình hoạt động trí tuệ của con người bao gồm 4 thao tác cơ bản:

1- Xác định tập đích (goals).

2- Thu thập các sự kiện (facts) và các luật suy diễn (inference rules) để đạt được đích đặt ra.

3- Thu gọn (pruning) quá trình suy luận nhằm xác định tập các suy diễn có thể sử dụng được.

4- Áp dụng các cơ chế suy diễn cụ thể (inference mechanisms) để đưa các sự kiện ban đầu đi đến đích.

Trí tuệ máy: cũng không có một định nghĩa tổng quat, nhưng cũng có thể nêu các đặc trưng chính:

1- Khả năng học.

2- Khả năng mô phỏng hành vi của con người.

3- Khả năng trừu tượng hoá, tổng quát hoá và suy diễn .

4- Khả năng tự giải thích hành vi.

5- Khả năng thích nghi tình huống mới kể cả thu nạp tri thức và dữ liệu.

6- Khả năng xử lý các biểu diễn hình thức như các ký hiệu tượng trưng.

7- Khả năng sử dụng tri thức heuristic.

8- Khả năng xử lý các thông tin không đầy đủ, không chính xác

5. Một số chuyên ngành của Trí tuệ nhân tạo:

1. Các phương pháp tìm kiếm lời giải.
2. Hệ chuyên gia
3. Máy nhìn và ngôn ngữ.
4. Lý thuyết nhận dạng.
5. Các mô hình thần kinh.
6. Người máy.

Chương 1

BIỂU DIỄN BÀI TOÁN TRONG KHÔNG GIAN TRẠNG THÁI

1. Đặt vấn đề.

Khi giải quyết bài toán bằng phương pháp tìm kiếm, trước hết ta phải xác định ***không gian tìm kiếm*** bao gồm tất cả các đối tượng trên đó thực hiện việc tìm kiếm.

Một phương pháp biểu diễn vấn đề phù hợp là sử dụng các khái niệm ***trạng thái*** (state) và ***toán tử*** (operator).

Phương pháp giải quyết vấn đề dựa trên khái niệm trạng thái và toán tử được gọi là cách tiếp cận giải quyết vấn đề nhờ không gian trạng thái.

2. Mô tả trạng thái

Giải bài toán trong không gian trạng thái, trước hết phải xác định dạng mô tả trạng thái bài toán sao cho bài toán trở nên đơn giản hơn, phù hợp bản chất vật lý của bài toán (Có thể sử dụng các xâu ký hiệu, véctơ, mảng hai chiều, cây, danh sách).

Mỗi trạng thái chính là mỗi hình trạng của bài toán, các tình trạng ban đầu và tình trạng cuối của bài toán gọi là trạng thái đầu và trạng thái cuối.

Ví dụ 1. Bài toán đong nước

Cho 2 bình có dung tích lần lượt là m và n (lit). Với nguồn nước không hạn chế, dùng 2 bình trên để đong k lit nước. Không mất tính tổng quát có thể giả thiết k <= min(m,n).

Tại mỗi thời điểm xác định, lượng nước hiện có trong mỗi bình phản ánh bản chất hình trạng của bài toán ở thời điểm đó.

- Gọi x là lượng nước hiện có trong bình dung tích m và y là lượng nước hiện có trong bình dung tích n. Như vậy bộ có thứ tự (x,y) có thể xem là trạng thái của bài toán. Với cách mô tả như vậy, các trạng thái đặc biệt của bài toán sẽ là:

- Trạng thái đầu: (0,0)
- Trạng thái cuối: (x,k) hoặc (k,y), $0 \le x \le m$, $0 \le y \le n$

Ví dụ 2. Bài toán trò chơi 8 số

Trong bảng ô vuông 3 hàng, 3 cột , mỗi ô chứa một số nằm trong phạm vi từ 1 đến 8 sao cho không có 2 ô có cùng giá trị, có một ô trong bảng bị trống (không chứa giá trị nào cả. Xuất phát từ một sắp xếp nào đó các số trong bảng, hãy dịch chuyển ô trống sang phải, sang trái, lên trên hoặc xuống dưới (nếu có thể được) để đưa về bảng ban đầu về bảng qui ước trước. Chẳng hạn Hình 1. dưới đây là bảng xuất phát và Hình 2. là bảng mà ta phải thực hiện các bước di chuyển ô trống để đạt được.

2	8	3
1	6	4
7		5

Hình 1.

1	2	3
8		4
7	6	5

Hình 2.

Giá trị các phần tử trong bảng xác định trạng thái bài toán. Vì vậy có thể mô tả trạng thái của bài toán bằng một ma trận $A_{3*3}= (a_{ij})$, $a_{ij} \in \{0..8\}$, $a_{ij} <> a_{kl}$, $\forall i<>k, j<> l$

- Trạng thái đầu của bài toán là ma trận

$$\begin{pmatrix} 2 & 8 & 3 \\ 1 & 6 & 4 \\ 7 & 0 & 5 \end{pmatrix}$$

- Trạng thái cuối là ma trận

$$\begin{pmatrix} 1 & 2 & 3 \\ 8 & 0 & 4 \\ 7 & 6 & 5 \end{pmatrix}$$

Có thể phát biểu dạng tổng quát của bài toán này (Trò chơi n^2-1 số)

Ví dụ 3. Bài toán tháp Hà Nội

Cho ba cọc 1, 2, 3. Ở cọc 1 ban đầu có n đĩa sắp xếp theo thứ tự to dần từ dưới lên trên. Hãy dịch chuyển n đĩa đó sang cọc thứ 3 sao cho:

- Mỗi lần chỉ chuyển một đĩa.
- Trong mỗi cọc không cho phép đĩa to nằm trên đĩa nhỏ hơn.

Bài toán xác định khi biết được từng đĩa đang nằm ở cọc nào. Hay nói cách khác, có hai cách xác định:

1- Cọc 1 hiện đang chứa những đĩa nào? Cọc 2 hiện đang chứa những đĩa nào? Và cọc 3 đang chứa những đĩa nào.

2- Đĩa lớn thứ i hiện đang nàm ở cọc nào? (i = 1 .. n)

Như vậy cách mô tả trạng thái bài toán không duy nhất, vấn đề là chọn cách mô tả nào để đạt được mục đích dễ dàng nhất.

Theo trên, với cách thứ nhất ta phải dùng 3 danh sách động vì số đĩa trên mỗi cọc là khác nhau trong từng thời điểm khác nhau.

Cách thứ hai, nhìn qua thì khó mô tả nhưng dựa vào khái niệm về bộ có thứ tự trong toán học, cách này mô tả bài toán hiệu quả hơn. Thật vậy, nếu gọi x_i là cọc chứa đĩa lớn thứ i, trong đó $x_i \in \{1, 2, 3\}$, $i \in \{1 .. n\}$. Khi đó bộ có thứ tự $(x_1, x_2, . . . ,x_n)$ có thể dùng làm dạng mô tả trạng thái đang xét của bài toán. Với cách mô tả này,

Trạng thái đầu là (1,1,. . .,1)

Trạng thái cuối là (3,3,. . .,3)

3. Toán tử chuyển trạng thái.

Toán tử chuyển trạng thái thực chất là các phép biến đổi đưa từ trạng thái này sang trạng thái khác. Có hai cách dùng để biểu diễn các toán tử:

- Biểu diễn như một hàm xác định trên tập các trạng thái và nhận giá trị cũng trong tập này.
- Biểu diễn dưới dạng các quy tắc sản xuất S? A có nghĩa là nếu có trạng thái S thì có thể đưa đến trạng thái A.

Ví dụ 1. Bài toán đong nước

Các thao tác sử dụng để chuyển trạng thái này sang trạng thái khác gồm: Đổ đầy một bình, đổ hết nước trong một bình ra ngoài, đổ nước từ bình này sang bình khác. Như vậy, nếu trạng thái đang xét là (x,y) thì các trạng thái kế tiếp có thể chuyển đến sẽ là:

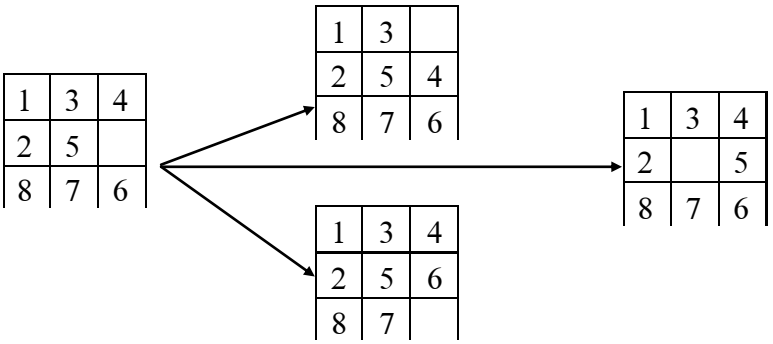
(x,y) ⟶

- **(m,y)**
- **(x,n)**
- **(0,y)**
- **(x,0)**
- **(0, x+ y) nếu x+y < = n**
- **(x+y -n,n) nếu x+y > n**
- **(x+ y,0) nếu x+y < = m**
- **(m, x+y-m) nếu x+y > m**

Ví dụ 2. Trò chơi 8 số

Các thao tác để chuyển trạng thái tương ứng với việc chuyển ô trống sang phải, sang trái, lên, xuống nếu có thể được.

- Biểu diễn theo quy tắc sản xuất:



- Biểu diễn theo một hàm

Gọi hàm f_u là hàm biểu diễn cho toán tử chuyển ô trống lên trên; gọi B (B= (b_{ij})) là trạng thái sau khi di chuyển ô trống ở trạng thái A (A= (a_{ij})) lên trên,

nghĩa là: B= $f_u(A)$, giả sử ô trống đang ở vị trí (i_0, j_0) (hay nói cách khác $a_{i0\ j0} = 0$) thì hàm f được xác định như sau:

$$f_u(a_{ij}) = \begin{cases} a_{ij} & \forall (i, j) \quad \text{nếu } i_0 = 1 \\ a_{ij} & \text{nếu } (i, j) \neq (i_0\text{-}1, j_0) \text{ và } (i, j) \neq (i_0, j_0) \text{ và } i_0 > 1 \\ a_{i0\text{-}1,\ j0} & \text{nếu } (i, j) = (i_0, j_0),\ i_0 > 1 \\ a_{i0,\ j0} & \text{nếu } (i, j) = (i_0\text{-}1, j_0),\ i_0 > 1 \end{cases}$$

Tương tự, có thể xác định các phép chuyển ô trống xuống dưới f_d, qua trái f_l, qua phải f_r như sau:

$$f_d(a_{ij}) = \begin{cases} a_{ij} & \forall (i, j) \quad \text{nếu } i_0 = 3 \\ a_{ij} & \text{nếu } (i, j) \neq (i_0\text{+}1, j_0) \text{ và } (i, j) \neq (i_0, j_0) \text{ và } i_0 < 3 \\ a_{i0\text{-}1,\ j0} & \text{nếu } (i, j) = (i_0, j_0),\ i_0 < 3 \\ a_{i0,\ j0} & \text{nếu } (i, j) = (i_0\text{+}1, j_0),\ i_0 < 3 \end{cases}$$

$$f_l(a_{ij}) = \begin{cases} a_{ij} & \forall (i, j) \quad \text{nếu } j_0 = 1 \\ a_{ij} & \text{nếu } (i, j) \neq (i_0, j_0\text{-}1) \text{ và } (i, j) \neq (i_0, j_0) \text{ và } j_0 > 1 \\ a_{i0\text{-}1,\ j0} & \text{nếu } (i, j) = (i_0, j_0),\ j_0 > 1 \\ a_{i0,\ j0} & \text{nếu } (i, j) = (i_0, j_0\text{-}1),\ j_0 > 1 \end{cases}$$

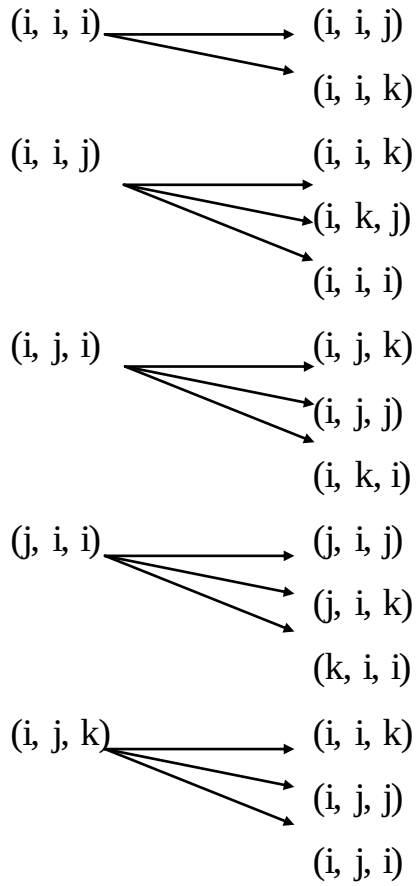
$$f_r(a_{ij}) = \begin{cases} a_{ij} & \forall (i, j) \quad \text{nếu } j_0 = 3 \\ a_{ij} & \text{nếu } (i, j) \neq (i_0, j_0\text{+}1) \text{ và } (i, j) \neq (i_0, j_0) \text{ và } j_0 < 3 \\ a_{i0\text{-}1,\ j0} & \text{nếu } (i, j) = (i_0, j_0),\ j_0 < 3 \\ a_{i0,\ j0} & \text{nếu } (i, j) = (i_0, j_0\text{+}1),\ j_0 < 3 \end{cases}$$

Ví dụ 3. Bài toán Tháp Hà Nội với n=3.

Mỗi trạng thái là một bộ ba (i, j, k). Có các trường hợp như sau:

- Ba đĩa cùng nằm trên một cọc: (i, i, i)
- Hai đĩa cùng nằm trên một cọc: (i, i, j), (i, j, i), (j, i, i)

- Ba đĩa nằm trên ba cọc phân biệt: (i, j, k)



4. Không gian trạng thái của bài toán.

Kkhông gian trạng thái là tập tất cả các trạng thái có thể có và tập các toán tử của bài toán.

Không gian trạng thái là một bộ bốn, Ký hiệu: K= (T, S, G, F). Trong đó,

T: tập tất cả các trạng thái có thể có của bài toán

S: trạng thái đầu

G: tập các trạng thái đích

F: tập các toán tử

Ví dụ 1. Không gian trạng thái của bài toán đong nước là bộ bốn T, S, G, F xác đinh như sau:

T = { (x,y) / 0 <= x <= m; 0 <= y <= n }

S = (0,0)

G = { (x,k) hoặc (k,y) / 0 <= x <= m; 0 <= y <= n}

F = Tập các thao tác đong đầy, đổ ra hoặc đổ sang bình khác thực hiện trên một bình.

Ví dụ 2. Không gian trạng thái của bài toán Tháp Hà nội với n = 3:

T = { (x1, x2, x3)/ xi ∈ {1, 2, 3} }

S = (1, 1, 1)

G = {(3, 3, 3)}

F = Tập các khả năng có thể chuyển đĩa đã xác định trong phần trước.

Ví dụ 3. Không gian trạng thái của bài toán trò chơi 8 số:

T = { $(a_{ij})_{3x3}$ / 0<= a_{ij} <= 8 và a_{ij} <> a_{kl} với i<> j hoặc k <> l}

S = Ma trận xuất phát của bài toán,

G = Ma trận cuối cùng của bài toán (các số nằm theo vị trí yêu cầu)

F = {f_l, f_r, f_u, f_d}

Tìm kiếm lời giải trong không gian trạng thái là quá trình tìm kiếm xuất phát từ trạng thái ban đầu, dựa vào toán tử chuyển trạng thái để xác định các trạng thái tiếp theo cho đến khi gặp được trạng thái đích.

5. Biểu diễn không gian trạng thái dưới dạng đồ thị

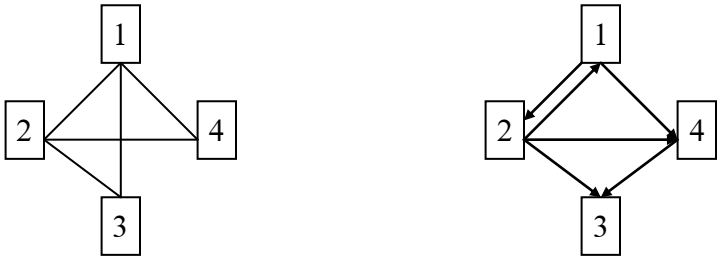
5.1. Các khái niệm

- Đồ thị G = (V,E) trong đó V:tập đỉnh, E: tập cung (E⊂V*V)

Chú ý

- G là đồ thị vô hướng thì (i, j) là một cạnh cũng như là (j, i) (tức là:(i, j)∈E thì (j,i)∈E)
- Nếu G là đồ thị có hướng thì cung (i, j) hoàn toàn khác với cung (j, i).

Ví dụ xét dồ thị vô hướng G_1 và đồ thị có hướng G_2



G_1 G_2

- Tập đỉnh kề:

$\forall n \in V$, $T(n)=\{m \in V / (n,m) \in E\}$ được gọi là tập các đỉnh kề của n

- Đường đi:

$p = (n_1,...,n_k)$ được gọi là đường đi từ đỉnh $n_1 \rightarrow n_k$ nếu $n_i \in V$, $\forall i=1,k$;

$(n_i, n_{i+1}) \in E$ $\forall i=\overline{1, k-1}$

- Cây là đồ thị có đỉnh gốc $n_0 \in V$ thoả:

Một đồ thị G=(V, E) gọi là cây nếu tồn tại một đỉnh $n_0 \in V$ có những tính chất sau:

- $\forall n \in V$, $n \in \hat{T}(n_0)$, trong đó $\hat{T}(n_0)$: tập các đỉnh thuộc dòng dõi của n_0 (n_0 là tổ tiên của n)
- $\forall n \in V$, $\exists! m \in V$ sao cho $n \in T(m)$; m được gọi là cha của n.

5.2. Biểu diễn không gian trạng thái bằng đồ thị

Theo ngôn ngữ đồ thị, không gian trạng thái tương ứng với một đồ thị định hướng trong đó: Các trạng thái tương ứng với các đỉnh trong đồ thị, nếu tồn tại toán tử chuyển trạng thái thì có cung (s, t)

Để thấy rõ mối tương quan, ta có bảng sau

KGTT	*Đồ thị*
Trạng thái	Đỉnh
Toán tử	Cung
Dãy các trạng thái liên tiếp	Đường đi

5.3. Biểu diễn đồ thị

Cho đồ thị G = (V,E) , giả sử V={1, 2,....,n}. Có hai cách thường dùng để biểu diễn đồ thị G lưu trữ trong máy tính.

i) Biểu diễn bằng ma trận kề

Đồ thị G được biểu diễn bởi ma trận kề $A=(a_{ij})_{nxn}$, với n là số đỉnh của đồ thị, trong đó:

$$a_{ij}= \begin{cases} 1 & \text{nếu } (i, j) \in E \\ 0 & \text{trong trường hợp ngược lại} \end{cases}$$

Nếu G là đồ thị vô hướng thì ma trận kề A là ma trận đối xứng.

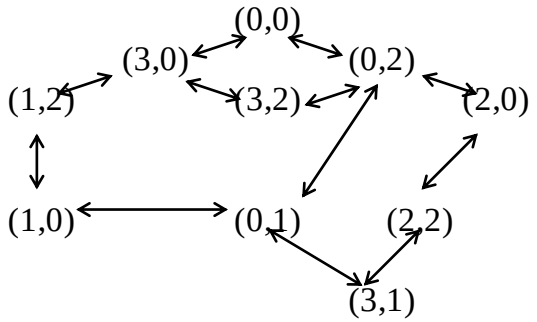
Ví dụ Với đồ thị vô hướng G_1 và đồ thị có hướng G_2 ở trên ta có các ma trận kề sau:

$$G_1: \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \qquad G_2: \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

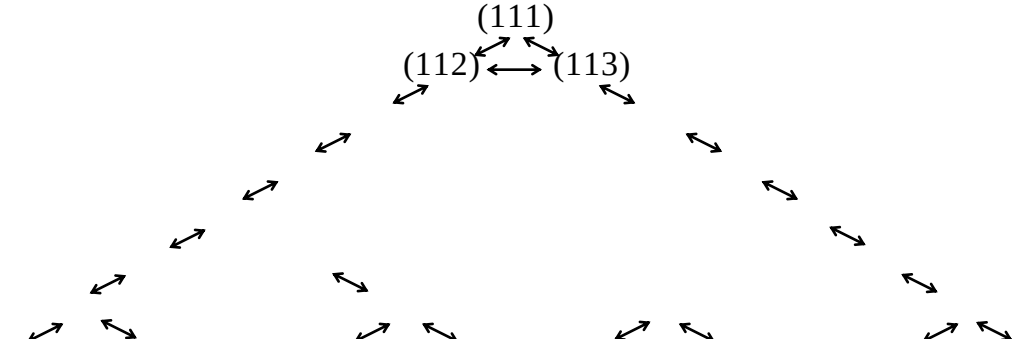
ii) Biểu diễn bằng danh sách kề.

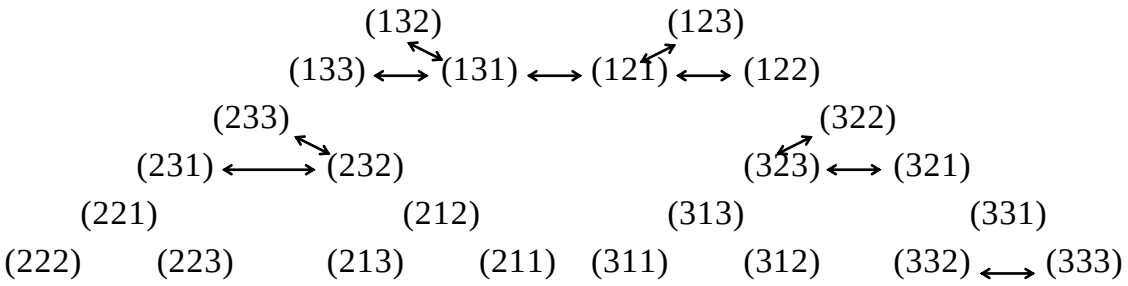
Với mỗi đỉnh i của đồ thị, ta có một danh sách tất cả các đỉnh kề với i, ta ký hiệu là List(i). Để thể hiện List(i) ta có thể dùng mảng, kiểu tập hợp hay kiểu con trỏ. Ví dụ với đồ thị G1, ta có List(1)= [2, 3, 4]

Ví dụ 1. Bài toán đong nước m=3, n=2, k=1



Ví dụ 2. Tháp Hà Nội với n = 3




(132)
(123)
(133)
(131)
(121)
(122)
(233)
(322)
(231)
(232)
(323)
(321)
(221)
(212)
(313)
(331)
(222)
(223)
(213)
(211)
(311)
(312)
(332)
(333)

6. BÀI TẬP

Xây dựng không gian trạng thái đối với các bài toán sau:

6.1. Cho n thành phố đánh số từ 1 đến n. Giao thông đường bộ giữa hai thành phố i và j được cho bởi giá trị a_{ij} như sau: a_{ij} = -1 có nghĩa là không có đường bộ đi từ thành phố i sang thành phố j và a_{ij} = 1 nếu có đường đi trực tiếp từ thành phố i sang thành phố j. Tìm đường đi từ thành phố i_0 sang thành phố j_0.

6.2. Cho k và n là 2 số nguyên dương. Có 2^k viên sỏi, được phân bố trong n đống, đống thứ nhất có a_1 viên, đống thứ 2 có a_2 viên, …, đống thứ n có a_n viên và tất nhiên $a_1 + a_2 + \ldots + a_n = 2^k$. Người ta cần san sẻ lượng sỏi từ các đống để dồn sỏi trở về 1 đống. Quy tắc san sỏi như sau: mỗi lần san áp dụng cho 2 đống sỏi, giả sử 1 đống có a viên và đống kia có b viên (không giảm tổng quát, có thể giả thiết $a \geq b$) thì san sỏi từ đống có a viên sang đống có b viên để thành một đống có a-b viên và đống kia 2*b viên.

Hướng dẫn: Trạng thái của bài toán phải xác định được số sỏi hiện có trong mỗi đống.

6.3. Một dãy các số nguyên dương $a_1, a_2, \ldots, a_n$ được gọi là hợp lý nếu thoả mãn hai điều kiện:

- a_n là số nguyên tố.
- $a_{i+1} = a_i + 1$ hoặc $2*a_i$

Cho trước số a_1, hãy tìm dãy hợp lý $a_1, a_2, \ldots, a_n$.

6.4 Bài toán người đưa hàng.

Người đưa hàng cần phải xác định được hành trình ngắn nhất sao cho mỗi thành phố đi đến đúng một lần và quay trở lại thành phố xuất phát. Giả sử thành phố xuất phát là thành phố 1, có tất cả n thành phố đánh số từ 1 đến n.

Hướng dẫn:

- Mỗi trạng thái cho bởi danh sách các thành phố đã đi qua cho đến thời điểm hiện tại, trong đó không cho phép một thành phố nào được xuất hiện nhiều hơn một lần trừ thành phố 1 sau khi đã liệt kê tất cả các thành phố còn lại.
- Các toán tử tương ứng với các hành động
  1- đi tới thành phố 1
  2- đi tới thành phố 2
  3- đi tới thành phố 3.
  . . .

6.5. Bài toán phân tích cú pháp.

Văn phạm G là bộ bốn G = (N, T, P, S), N là tập ký hiệu không kết thúc, T là tập ký hiệu kết thúc, $S \in N$ là ký hiệu đầu và P là tập sản xuất có dạng $\alpha \rightarrow \beta$, ở đây $\alpha, \beta \in (NUT)$. Ngôn ngữ sinh ra bởi văn phạm G được định nghĩa bởi:\

$L(G) = \{ \omega \in T \mid S \Rightarrow \omega \}$, $S \Rightarrow \omega$ có nghĩa là $\exists \omega_1, \ldots, \omega_n \in (NUT)$ sao cho $\omega_i \Rightarrow \omega_{i+1}$, $\omega_1 = S$ và $\omega_n = \omega$.

Bài toán phân tích cú pháp được phát biểu : ch trước văn phạm G với xâu $\omega \in T$ đã cho hãy xác định xem $\omega \in L(G)$ hay không ?

Chương 2

CÁC PHƯƠNG PHÁP TÌM KIẾM LỜI GIẢI TRONG KHÔNG GIAN TRẠNG THÁI

Quá trình tìm kiếm lời giải của bài toán được biểu diễn trong không gian trạng thái được xem như quá trình dò tìm trên đồ thị, xuất phát từ trạng thái ban đầu, thông qua các toán tử chuyển trạng thái, lần lượt đến các trạng thái tiếp theo cho đến khi gặp được trạng thái đích hoặc không còn trạng thái nào có thể tiếp tục được nữa. Khi áp dụng các phương pháp tìm kiếm trong không gian trạng thái , người ta thường quan tâm đến các vấn đề sau:

- Kỹ thuật tìm kiếm lời giải
- Phương pháp luận của việc tìm kiếm
- Cách thể hiên các nút trong quá trình tìm kiếm (mô tả trạng thái bài toán)
- Việc chọn các toán tử chuyển trạng thái nào để áp dung và có khả năng sử dụng .phương pháp may rủi trong quá trình tìm kiếm.

Tuy nhiên, không phải các phương pháp này có thể áp dụng cho tất cả các bài toán phức tạp mà cho từng lớp bài toán. Việc chọn chiến lược tìm kiếm cho bài toán cụ thể phụ thuộc nhiều vào các đặc trưng của bài toán.

Các thủ tục tìm kiếm điển hình bao gồm:

- Tìm kiếm theo chiều rộng (Breadth – First Search)
- Tìm kiếm theo chiều sâu (Depth – First Search)
- Tìm kiếm sâu dần (Depthwise Search)
- Tìm kiếm cực tiểu hoá giá thành (Cost minimization Search).
- Tìm kiếm với tri thức bổ sung (Heuristic Search).

1. Phương pháp tìm kiếm theo chiều rộng.

1.1. Kỹ thuật tìm kiếm rộng.

Kỹ thuật tìm kiếm rông là tìm kiếm trên tất cả các nút của một mức trong không gian bài toán trước khi chuyển sang các nút của mức tiếp theo.

Kỹ thuật tìm kiếm rộng bắt đầu từ mức thứ nhất của không gian bài toán, theo hướng dẫn của luật trọng tài, chẳng hạn "đi từ trái sang phải". Nếu không thấy lời giải tại mức này, nó chuyển xuống mức sau để tiếp tục … đến khi định vị được lời giải nếu có.

1.2. Giải thuật.

Input:

Cây/Đồ thị G = (V,E) với đỉnh gốc là n_0 (trạng thái đầu)

Tập đích Goals

Output:

Một đường đi p từ n_0 đến một đỉnh $n_* \in$ Goals

Method:

Sử dụng hai danh sách hoạt động theo nguyên tắc FIFO (queue) MO và DONG

Procedure BrFS; (Breadth First Search)

```
Begin
    Append(MO,n0)
    DONG=null;
    While MO <> null do
            begin
                n:= Take(MO);
                if n∈ DICH then exit;
                Append(DONG, n);
                For m∈ T(n) and m∉DONG+MO do
                    Append(MO, m);
```

```
                end;
        Write ('Không có lời giải');
End;
```

Chú ý: Thủ tục Append(MO,n_0) bổ sung một phần tử vào queue MO.

Hàm Take(MO) lấy một phần tử trong queue MO.

- Nếu G là cây thì không cần dùng danh sách DONG.

1.3. Đánh giá độ phức tạp của giải thuật tìm kiếm rộng.

Giả sử rằng, mỗi trạng thái khi được xét sẽ sinh ra k trạng thái kế tiếp. Khi đó ta gọi k là nhân tố nhánh. Nếu bài toán tìm được nghiệm theo phương pháp tìm kiếm rộng có độ dài d. Như vậy, đỉnh đích sẽ nằm ở mức d+1, do đó số đỉnh cần xét lớn nhất là:

$1 + k + k^2 + \ldots + k^d$.

Như vậy độ phức tạp thời gian của giải thuật là $O(k^d)$. Độ phức tạp không gian cũng là $O(k^d)$, vì tất cả các đỉnh của cây tìm kiếm ở mức d+1 đều phải lưu vào danh sách.

1.4. Ưu và nhược điểm của phương pháp tìm kiếm rộng.

1.4.1. Ưu điểm.

- Kỹ thuật tìm kiếm rộng là kỹ thuật vét cạn không gian trạng thái bài toán vì vậy sẽ tìm được lời giải nếu có.
- Đường đi tìm được đi qua ít đỉnh nhất.

1.4.2. Nhược điểm.

- Tìm kiếm lời giải theo thuật toán đã định trước, do vậy tìm kiếm một cách máy móc; khi không có thông tin hổ trợ cho quá trình tìm kiếm, không nhận ra ngay lời giải.
- Không phù hợp với không gian bài oán kích thước lớn. Đối với loại bài toán này, phương pháp tìm rộng đối mặt với các nhu cầu:

- Cần nhiều bộ nhớ theo số nút cần lưu trữ.
- Cần nhiều công sức xử lý các nút, nhất là khi các nhánh cây dài, số nút tăng.
- Dễ thực hiện các thao tác không thích hợp, thừa, đưa đến việc tăng đáng kể số nút phải xử lý.

• Không hiệu qủa nếu lời giải ở sâu. Phương pháp này không phù hợp cho trường hợp có nhiều đường dẫn đến kết quả nhưng đều sâu.

• Giao tiếp với người dùng không thân thiện. Do duyệt qua tất cả các nút, việc tìm kiếm không tập trung vào một chủ đề.

1.5. Các ví dụ.

Ví dụ 1. Bài toán đong nước với m = 5, n= 4, k =3

Mức 1: Trạng thái đầu (0;0)

Mức 2: Các trạng thái (5;0), (0;4), Mức 3: (5;4), (1;4), (4,0)

Mức 4: (1;0), (4;4) Mức 5: (0;1), (5;3)

Ở mức 5 ta gặp trạng thái đích là (5;3) vì vậy có được lời giải như sau:

(0;0)→ (0;4) → (4;0) → (4;4) → (5;3)

Để có được lời giải này ta phải lưu lại vết của đường đi, có thể trình bày quá trình tìm kiếm dưới dạng bảng sau:

i	T(i)	↑MO ↓	DONG
		(0;0)	
(0;0)	(5;0) **(0;4)**	(5;0) (0;4)	(0;0)
(5;0)	(5;4) (0;0) (1;4)	(0;4) (5;4) (1;4)	(0;0) (5;0)
(0;4)	(5;4) (0;0) **(4;0)**	(5;4) (1;4) (4;0)	(0;0) (5;0) (0;4)
(5;4)	(0;4) (5;0)	(1;4) (4;0)	(0;0) (5;0) (0;4) (5;4)
(1;4)	(5;4) (0;4) (1;0) (5;0)	(4;0) (1;0)	(0;0) (5;0) (0;4) (5;4) (1;4)
(4;0)	(5;0) (4;4) (0;0) **(0;4)**	(1;0) (4;4)	(0;0) (5;0) (0;4) (5;4) (1;4) (4;0)

(1;0)	(5;0) (1;4) (0;1)	(4;4) (0;1)	(0;0) (5;0) (0;4) (5;4) (1;4) (4;0) (1;0)
(4;4)	(5;4) (0;4) (4;0) **(5;3)**	(0;1) (5;3)	(0;0) (5;0) (0;4) (5;4) (1;4) (4;0) (1;0) (4;4)
(0;1)	(5;1) (0;4) (0;0) (1;0)	(5;3) (5;1)	(0;0) (5;0) (0;4) (5;4) (1;4) (4;0) (1;0) (0;1)
(5;3)			

Ví dụ 2. Bài toán trò chơi 8 số

Bảng xuất phát

2	8	3
1	6	4
7		5

Bảng kết thúc

1	2	3
8		4
7	6	5

Mức 1: Có một trạng thái

2	8	3
1	6	4
7		5

Mức 2: Có ba trạng thái

2	8	3
1		4
7	6	5

2	8	3
1	6	4
	7	5

2	8	3
1	6	4
7	5	

Mức 3: Có năm trạng thái

2	8	3
	1	4
7	6	5

2	8	3
1	4	
7	6	5

2		3
1	8	4
7	6	5

2	8	3
	6	4

2	8	3
1	6	

1	7	5

7	5	4

Mức 4: Có mười trạng thái

	8	3
2	1	4
7	6	5

2	8	3
7	1	4
	6	5

2	8	
1	4	3
1	7	5

2	8	3
1	4	5
7	6	

	2	3
1	8	4
7	6	5

2	3	
1	8	4
7	6	5

	8	3
2	6	4
1	7	5

2	8	3
6		4
1	7	5

2	8	
1	6	3
7	5	4

2	8	3
1	6	4
7	5	

Mức 6: Có 12 trạng thái

1	2	3
	8	4
7	6	5

2	3	4
1	8	
7	6	5

8		3
2	1	4
7	6	5

2	8	3
7	1	4
6		5

2		8
1	4	3
7	6	5

2	8	3
1	4	5
7		6

8		3
2	6	4
1	7	5

2		3
6	8	4
1	7	5

2	8	3
6	7	4
1		5

2		8
1	6	3
7	5	4

2		3
1	8	3
7	5	4

2	8	3
1	5	6
7		4

Mức 6: Có 24 trạng thái

1	2	3
8		4
7	6	5

1	2	3
7	8	4
	6	5

. . .

Ở mức này ta gặp được trạng thái đích.

1	2	3
8		4
7	6	5

2. Phương pháp tìm kiếm theo chiều sâu.

2.1. Kỹ thuật tìm kiếm sâu.

Tìm kiếm sâu trong không gian bài toán được bắt đầu từ một nút rồi tiếp tục cho đến khi hoặc đến ngõ cụt hoặc đến đích. Tại mỗi nút có luật trong tài, chẳng hạn, "đi theo nút cực trái", hướng dẫn việc tìm. Nếu không đi tiếp đượoc, gọi là đến ngõ cụt, hệ thống quay lại một mức trên đồ thị và tìm theo hướng khác, chẳng hạn, đến nút "sát nút cực trái". Hành động này gọi là quay lui.

Thuật toán tìm kiếm theo chiều sâu được hình dung như việc khảo sát một cây bắt đầu từ gốc đi theo mọi cành có thể được, khi gặp cành cụt thì quay lại xét cành chưa đi qua.

- Ở bước tổng quát, giả sử đang xét đỉnh i, khi đó các đỉnh kề với i có các trường hợp:

+ Nếu tồn tại đỉnh j kề i chưa được xét thì xét đỉnh này (nó trở thành đỉnh đã xét) và bắt đầu từ đó tiếp tục quá trình tìm kiếm với đỉnh này..

+ Nếu với mọi đỉnh kề với i đều đã được xét thì i coi như duyệt xong và quay trở lại tìm kiếm từ đỉnh mà từ đó ta đi đến được i.

2.2. Giải thuật.

Input:

Cây/Đồ thị G = (V,E) với đỉnh gốc là n_0 (trạng thái đầu)

Tập đích Goals

Output:

Một đường đi p từ n_0 đến một đỉnh $n_* \in$ Goals

Method:

Sử dụng hai danh sách hoạt động theo nguyên tắc LIFO (Stack) MO và DONG

Procedure DFS; (Depth First Search)

```
Begin
    Push (MO,n_o)
    DONG=null;
    While MO <> null do
            begin
                n:=pop (MO);
                if n∈ DICH then exit;
                push (DONG, n);
                For m∈ T(n) and m∉DONG+MO do
                    Push (MO, m);
            end;
    Write ('Không có lời giải');
End;
```

Chú ý: Thủ tục Push(MO,n_0) thực hiện việc bổ sung n_0 vào stack MO

Hàm Pop(MO) lấy phần tử đầu tiên trong Stack MO.

2.3. Đánh giá độ phức tạp của thuật toán tìm kiếm sâu.

Gải sử nghiệm của bài toán là đường đi có độ dài d, cây tìm kiếm có nhân tố nhánh là k. Có thể xãy ra nghiệm là đỉnh cuối cùng được xét ở mức d+1 theo luật trọng tài. Khi đó độ phức tạp thời gian của thuật toán tìm kiếm theo chiều sâu trong trường hợp xấu nhất là $O(k^d)$.

Để đánh giá độ phức tạp không gian của thuật toán tìm kiếm sâu ta có nhận xét ràng: Khi xét đỉnh j, ta chỉ cần lưu các đỉnh chưa được xét mà chúng là những đỉnh con của những đỉnh nằm trên đường đi từ đỉnh gốc đến j. Vì vậy chỉ cần lưu tối đa la k*d. Do đó độ phức tạp không gian của thuật toán là O(k*d).

2.4. Ưu và nhược điểm của phương pháp tìm kiếm sâu.

2.4.1. Ưu điểm.

- Nếu bài toán có lời giải, phương pháp tìm kiếm sâu bảo đảm tìm ra lời giải.
- Kỹ thuật tìm kiếm sâu tập trung vào đích, con người cảm thấy hài lòng khi các câu hỏi tập trung vào vấn đề chính.
- Do cách tìm của kỹ thuật này, nếu lời giải ở rất sâu, kỹ thuật tìm sâu sẽ tiết kiệm thời gian.

2.4.2. Nhược điểm.

- Tìm sâu khai thác không gian bài toán để tìm lời giải theo thuật toán đơn giản một cách cứng nhắc. Trong quá trình tìm nó không có thông tin nào hỗ trợ để phát hiện lời giải. Nếu chọn nút ban đầu không thích hợp có thể không dẫn đến đích của bài toán.
- Không phù hợp với không gian bài toán lớn, kỹ thuật tìm kiếm sâu có thể không đến lời giải trong khoảng thời gian vừa phải.

2.5. Các ví dụ.

Ví dụ 1. Bài toán đong nước với m = 5, n = 4, k = 3

Nếu ta chọn nhánh ưu tiên đổ đầy bình thứ hai thì sẽ tìm thấy lời giải rất nhanh. Quá trình tìm kiếm có thể trình bày bằng bảng dưới đây

i	T(i)	MO ↓↑	DONG
		(0;0)	
(0;0)	(5;0) **(0;4)**	(5;0) (0;4)	(0;0)
(0;4)	(5;4) (0;0) **(4;0)**	(5;0) (5;4) (4;0)	(0;0) (0;4)

(4;0)	(5;0) (4;4) (0;0) **(0;4)**	(5;0) (5;4) (4;4)	(0;0) (0;4) (4;0)
(4;4)	(5;4) (0;4) (4;0) **(5;3)**	(5;0) (5;4) (5;3)	(0;0) (0;4) (4;0) (4;4)
(5;3)			

Lời giải tìm được: (0;0) → (0;4) → (4;0) → (4;4) → (5;3)

Ví dụ 2. Bài toán Tháp Hà nội với n = 3.

Nhắc lại, dùng bộ ba (x_1; x_2; x_3) biểu diễn trạng thái bài toán, với x_i là cọc chứa đĩa lớn thứ i.

i	T(i)	MO ↓↑	DONG
		(1;1;1)	
(1;1;1)	(1;1;2) (1;1;3)	(1;1;2) (1;1;3)	(1;1;1)
(1;1;3)	(1;1;1)(1;1;2) (1;2;3)	(1;1;2)(1;2;3)	(1;1;1)(1;1;3)
(1;2;3)	(1;1;3) (1;2;1) (1;2;2)	(1;1;2)(1;2;1)(1;2;2)	(1;1;1)(1;1;3)(1;2;3)
(1;2;2)	(1;2;3) (1;2;1) (3;2;2)	(1;1;2)(1;2;1)(3;2;2)	(1;1;1)(1;1;3)(1;2;3)(1;2;2)
(3;2;2)	(1;2;2) (3;2;3) (3;2;1)	(1;1;2)(1;2;1)(3;2;1)	(1;1;1)(1;1;3)(1;2;3)(1;2;2) (3;2;2)
(3;2;1)	(3;2;2) (3;2;3) (3;3;1)	(1;1;2)(1;2;1)(3;3;1)	(1;1;1)(1;1;3)(1;2;3)(1;2;2) (3;2;2) (3;2;1)
(3;3;1)	(3;2;1) (3;3;2) (3;3;3)	(1;1;2)(1;2;1)(3;3;3)	(1;1;1)(1;1;3)(1;2;3)(1;2;2) (3;2;2) (3;2;1) (3;3;3)
(3;3;3)			

Lời giải của bài toán:

(1;1;1) → (1;1;3) → (1;2;3) → (1;2;2) → (3;2;2) → (3;2;1) → (3;3;1) → (3;3;3)

- Cả hai ví dụ trên, chúng ta đều thấy, tìm kiếm theo chiều sâu đều cho lời giải tốt và nhanh.

Ví dụ 3. Bài toán tìm dãy hợp lý với số hạng đầu $a_1 = 26$

Nhắc lại: Dãy $a_1, a_2, \ldots, a_n$ được gọi là hợp lý nếu thoả hai điều kiện:

- a_n là số nguyên tố
- $a_{k+1} = a_k+1$ hoặc $2*a_k$

Như vậy, khi biết a_k thì ta xác định được a_{k+1}. Vì vậy có thể mô tả trạng thái bài toán tương ứng với giá trj a_k tại thòi điểm đang xét. Ta có thể chỉ ra một cách tìm kiếm theo chiều sâu như sau

I	T(i)	MO ↓↑	DONG
		26	
26	27 52	27 52	26
52	53 104	27 53 104	26 52
104	105 208	27 53 105 208	26 52 104
208	209 416	27 53 105 209 416	26 52 104 208
. . .			

Với cách tìm kiếm theo theo thuật toán một cách máy móc như vậy thì rõ ràng không bao giờ đạt được đích. Trong khi chúng ta dễ dàng nhận được lời giải, chăng hạn:

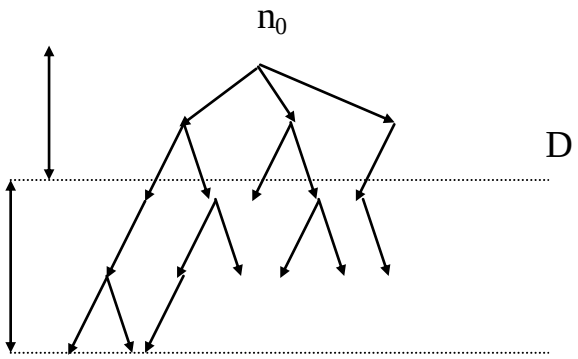
$a_1 = 26$; $a_2 = 52$; $a_3 = 53$. Như vậy n =3

3. Tìm kiếm sâu dần

3.1. Kỹ thuật tìm kiếm sâu dần.

Kỹ thuật tìm kiếm sâu dần là thực hiện việc tìm kiếm với độ sâu ở mức giuới hạn d nào đó. Nếu không tìm ra nghiệm ta tăng độ sâu lên d+1 và lại tìm kiếm theo độ sâu tới mức d+1. Quá trình trên được lặp lại với d lần lượt là 1, 2,...đến độ sâu max nào đó.

Kỹ thuật tìm kiếm sâu dần thường được thực hiện khi cây tìm kiếm chứa nhánh vô hạn, và nếu sử dụng tìm kiếm theo độ sâu ta có thể mắc kẹt ở một nhánh nào đó (thuật toán không dừng) và không tìm ra nghiệm.



3.2. Giải thuật.

Thuật toán tìm kiếm sâu dần sử dụng thuật toán tìm kiếm sâu hạn chế như thủ tục con. Đó là thủ tục tìm kiếm theo chiều sâu nhưng chỉ tới độ sâu d nào đó rồi quay lên.

Thủ tục tìm kiếm sâu hạn chế (depth_limitedsearch)

```
Procedure Depth_limited_search(d);        {d là tham số độ sâu}
    Begin
        Push (MO,n0);
        Depth(n0)=0;                      {hàm depth ghi lại độ sâu mỗi đỉnh}
        DONG=null;
        While  MO <> null  do
            begin
                n:=pop (MO);
                if n∈ DICH then exit;
                push (DONG, n);
                if depth(n)<=d then
                    For  m∈ T(n) and m∉DONG do
                     begin
                        Push (MO, m);
                        depth(m)=depth(n)+1;
                     end;
            end;
        Write ('Không có lời giải');
    End
```

Thuật toán tìm kiếm sâu dần (Depth_deepening_search) sẽ sử dụng thủ tục tìm kiếm sâu hạn chế như thủ tục con:

Procedure Depth_deepening_search;

```
Begin
    For d:=0 to max do
        Depth_limited_search(d);
    If thành công then exit;
End;
```

3.3. Nhận xét.

- Luôn tìm ra nghiệm (nếu bài toán có nghiệm), miễn là chọn max đủ lớn (giống như tìm kiếm theo chiều rộng)
- Có độ phức tạp thời gian là $O(k^d)$ (giống tìm kiếm rộng)
- Có độ phức tạp không gian là O(k*d) (giống tìm kiếm sâu)
- Giải thuật tìm kiếm sâu dần thương áp dụng cho các bài toán có không gian trạng thái lớn và độ sâu của nghiệm không biết trước.

4. Phương pháp tìm kiếm tốt nhất đầu tiên (Best First Search).

Cả hai kỹ thuật tìm kiếm rộng và tìm kiếm sâu đều là phương pháp cơ bản để khai thác không gian bài toán. Chúng đều vét cạn không gian để tìm ra lời giải theo thủ tục xác định trước. Mặc dù có sử dụng tri thức về trạng thái của bài toán để hướng dẫn tìm kiếm nhưng không phổ biến. Cho dù có một số ưu điểm, nhưng chúng là những kỹ thuật thực hiện một cách máy móc. Chính vì vậy chúng bị gán tên là "*kỹ thuật tìm kiếm mù*".

4.1. Kỹ thuật tìm kiếm tốt nhất đầu tiên.

Kỹ thuật tìm kiếm tốt nhất đầu tiên tìm lời giải có dùng tri thức về bài toán để hướng dẫn. Tri thức này hướng việc tìm kiếm về nút lời giải trong không gian bài toán.

Tại mỗi nút được xem xét, người ta sẽ quyết định việc tìm kiếm tiếp tục theo nhánh nào tin tưởng sẽ dẫn đến lời giải.

Trong các chương trình trí tuệ nhân tạo, kỹ thuật tìm kiếm tốt nhất đầu tiên sử dụng hàm đánh giá. Hàm này dùng các thông tin hiện tại về mức độ quan trọng của bài toán tại nút đó để gán giá trị cho nút này, gọi là trọng số của nút. Giá trị này được xem xét trong lúc tìm kiếm. Thông thường, nút có trọng số nhỏ (lớn) nhất sẽ được chọn trong quá trình tìm kiếm.

4.2. Hàm đánh giá

Trong nhiều vấn đề, ta có thể sử dụng kinh nghiệm, tri thức của chúng ta về vấn đề đó để đánh giá các trạng thái của vấn đề.

Với mỗi trạng thái u, ta sẽ xác dịnh một giá trị số h(u), số này đánh giá "sự gần đích" của trạng thái u. Hàm ***h(u) được gọi là hàm đánh giá***.

Phương pháp tìm kiếm kinh nghiệm là phương pháp tìm kiếm có sử dụng đến hàm đánh giá. Trong quá trình tìm kiếm, tại mỗi bước ta sẽ chọn trạng thái kế tiếp là trạng thái có nhiều hứa hẹn dẫn tới đích nhiều nhất.

Quá trình tìm kiếm trong không gian trạng thái có sử dụng hàm đánh giá bao gồm các bước cơ bản sau:

- Biểu diễn thích hợp các trạng thái và các toán tử chuyển trạng thái
- Xây dựng hàm đánh giá
- Thiết kế chiến lược chọn trạng thái ở mỗi bước

4.3. Ưu và nhược điểm của phương pháp tìm kiếm tốt nhất đầu tiên.

4.3.1. Ưu điểm.

- Phương pháp tìm kiếm tốt nhất đầu tiên tổ hợp các ưu điểm của phương pháp tìm kiếm rộng và tìm kiếm sâu.

- Ưu điểm chủ yếu của phương pháp tìm kiếm tốt nhất đầu tiên là dùng tri thức để dẫn dắt việc tìm kiếm. Tri thức này giúp người ta bắt đầu từ đâu là tốt nhất và cách tốt nhất để tiến hành tìm lời giải.

- Tìm kiếm tốt nhất đầu tiên tuân theo cách suy lý của một chuyên gia. Do đó có thể thấy rõ đường đi hơn tìm kiếm rộng và tìm kiếm sâu.

4.3.2. Nhược điểm.

- Quá trình tìm kiếm có thể đi xa khỏi lời giải. Kỹ thuật này chỉ xét một phần của không gian và coi đó là phần hứa hẹn hơn cả.

4.4. Giải thuật.

Dữ liệu tương tự như giải thuật tìm kiếm rông và sâu, sử dụng danh sách MO để lưu các đỉnh sẽ xét.

```
Procedure BFS; {Best First Search}
Begin
    Push(MO,n0);
    while  MO <> null do
    begin
        i := Pop(MO);
        if  i ∈ Goals then
            exit;
        for j ∈ T(i) do
            Push(MO,j);
        Sort(MO); {theo thứ tự của hàm đánh giá}
    end;
write('Khong co loi giai');
end;
```

4.5. Các ví dụ.

Ví dụ 1 Trong bài toán tìm kiếm đường đi trên bản đồ giao thông, ta có thể lấy độ dài của đường chim bay từ một thành phố đang xét tới một thành phố đích làm giá trị của hàm đánh giá của thành phố đang xét.

Ví dụ 2 Bài toán 8 số. Chúng ta có thể đưa ra hai cách đánh giá

u =

3	2	8
	6	4
7	1	5

đích =

1	2	3
8		4
7	6	5

- Hàm h_1: Với mỗi trạng thái u thì $h_1(u)$ là số quân không nằm đúng vị trí của nó trong trạng thái đích.

Với ví dụ trên, ta có **$h_1(u)=4$**

- Hàm h_2: Gọi $h_2(u)$ là là tổng khoảng cách giữa vị trí của các quân trong trạng thái u và vị trí của nó trong trạng thái đích. Ở đây, khoảng cách được hiểu là số lần dịch chuyển ít nhất theo hàng hoặc cột để đưa một quân ở vị trí của hiện tại tới trạng thái đích.

Với ví dụ trên, ta có:**$h_2(u)=2+3+1+3= 9$** (vì quân 3 cần ít nhất 2 dịch chuyển, quân 8 cần ít nhất 3 dịch chuyển, quân 6 cần ít nhất 1 dịch chuyển và quân 1 cần ít nhất 3 dịch chuyển)

5. Tìm kiếm đường đi có giá thành cực tiểu - Thuật toán AT

Cho đồ thị G= (V, E) biểu diễn bài toán với đỉnh xuất phát n_0 và tập đích DICH xác định.

Với mỗi phép chuyển trạng thái $n_i \rightarrow n_{i+1}$ tốn chi phí $c(n_i, n_{i+1})$ ký hiệu c(u) với $u= (n_i, ni_{+1}) \in E$

$$n_i \xrightarrow{c(u)} n_{i+1}$$

Vấn đề:

Tìm đường đi p: $n_0 \rightarrow n^* \in$ DICH sao cho $c(p) = \sum_{u \in p} c(u) \rightarrow \min$

Chẳng hạn trong bài toán tìm đường đi trong bản đồ giao thông, giá của cung (i,j) chính là độ dài của đường nối thành phố i với thành phố j. Độ dài đường đi được xác định là tổng độ dài các cung trên đường đi. Vấn đề đặt ra là tìm đường đi ngắn nhất từ trạng thái ban đầu đến trạng thái đích.

- **Phương pháp giải**

1) Nếu $c(u) = k(const) \quad \forall u \in E$ thì $c(p) \rightarrow \min \Leftrightarrow \# p \rightarrow \min \Rightarrow$ Dùng phương pháp tìm kiếm theo chiều rộng.

2) Gọi g(n) là giá của đường đi cực tiểu từ đỉnh n_0 đến n, khi đó bài toán có thể phát biểu như sau:

Tìm đường đi từ đỉnh $n_0 \rightarrow n_k \in DICH$ **sao cho:** $g(n_k) = \min\{g(n) / n \in DICH\}$

Lúc đó, ta có: $g(n_0) = 0$

$$g(m) = \min_{(n,m)\in E}\{g(n) + c(n,m)\}$$

Dùng 2 danh sách MO, DONG như trên. Tại mỗi thời điểm chọn đỉnh n trong MO ra xét là đỉnh thoả.

- **Thuật toán AT**

Input:

Đồ thị G = (V,E), Đỉnh xuất phát n_0

Hàm chi phí c: $E \to R^+$

c(i,j): xác định chi phí chuyển từ đỉnh i sang đỉnh j với (i,j) $\in$ E

Tập các đỉnh đích DICH

Output:

Đường đi từ đỉnh n_0 đến đỉnh $n_* \in$ DICH sao cho $g(n_*) = c(p) = \min\{g(n) | n \in DICH\}$.

Procedure AT;

{ Dùng $g^0(n)$ là chi phí cực tiểu của đường đi từ đỉnh xuất phát đến đỉnh n tại thời điểm đang xét và xem như hàm g}

```
Begin
    g(n0):= 0;
    push(MO, n0);
    While MO<>null do
        begin
            g(n) := min_{m∈MO} g(m)
            if n∈DICH then
                exit {xay dung duong di cuc tieu}
            push(DONG, n);
            if T(n) <>null then
                for  m∈T(n) do
```

```
                    if m∉MO+DONG then
                        begin
                            push(MO,m);
                            g(m):=g(n)+c(n,m);
                            cha(m):=n;
                        end
                    else
                        if g(m) >g(n)+c(n,m) then
                            begin
                                g(m):=g(n)+c(n,m);
                                cha(m):=n;
                            end;
            end;
            writeln('Khong co duong di');
End;
```

Ví dụ 1. Bài toán Tháp Hà Nội -với chi phí chuyển đĩa như sau:

Chi phí chuyển đĩa nhỏ giữa 2 cọc gần	1
Chi phí chuyển đĩa nhỏ giữa 2 cọc xa	3
Chi phí chuyển đĩa vừa giữa 2 cọc gần	2
Chi phí chuyển đĩa vừa giữa 2 cọc xa	5
Chi phí chuyển đĩa lớn giữa 2 cọc gần	4
Chi phí chuyển đĩa lớn giữa 2 cọc xa	8

Xuất phát từ đỉnh (1,1,1), ta có g(1,1,1) = 0.

Khi xét đỉnh **(1,1,1)** ta có các đỉnh kề và chi phí tương ứng :

g(1,1,2) = 1; g(1,1,3) = 3; như vậy đỉnh **(1,1,2)** được chọn

Các đỉnh kề của (1,1,2) có giá trị hàm g:

g(1,1,3) = 2 (ở đây giá của đỉnh (1,1,3) được tính lại); g(1,3,2) = 5; chọn đỉnh **(1,1,3)**, ta lại tính tiếp giá trị hàm g của các đỉnh kề với đỉnh này:

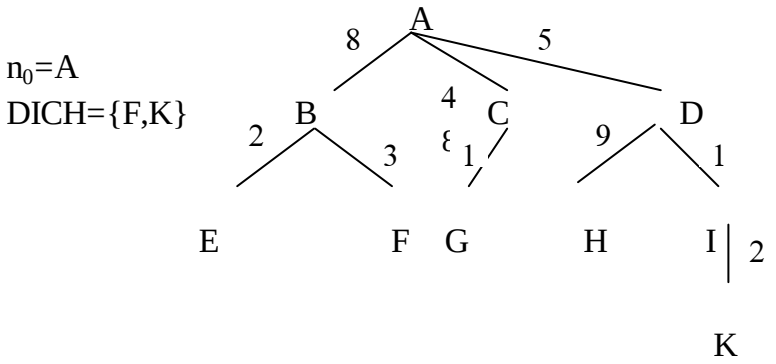
g(1,2,3) = 2; lại chọn đỉnh **(1,2,3)**; chi phí của các đỉnh kề với nó:

g(1,2,1) = 2 + 3 = 5; g(1,2,2) = 2 + 1 = 3; chọn đỉnh **(1,2,2)**

g(1,2,1) = 3 +1 = 4 (được tính lại); g(3,2,2) = 3 + 8 = 11, chọn đỉnh (1,2,1)

Cứ tiếp tục như vậy cho đến khi xét đỉnh (3,3,3).

Ví dụ 2



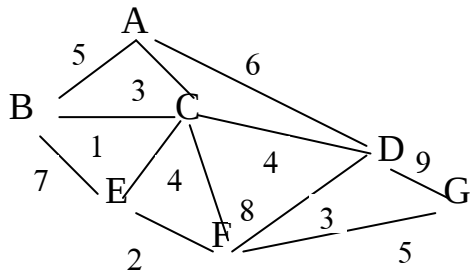
Có thể trình bày quá trình tìm kiếm bằng bảng dưới đây. Ký hiệu giá trị g(n) là chỉ số dưới tương ứng đỉnh n: $n_{g(n)}$

i	**T(i)**	**MO**	**DONG**
		A_0	
A	B C D	B_8 C_4 D_5	**A**
C	G	B_8 D_5 G_5	**A C**
D	H I	B_8 G_5 H_{14} I_6	**A C D**
G		B_8 H_{14} I_6	**A C D G**
I	K	B_8 H_{14} K_8	**A C D G**
B	E F	H_{14} K_8 E_{10} F_{11}	**A C D G B**
K			

Lời giải của bài toán là A→ D → I → K và chi phí của đường đi tìm được là 8

Ví dụ 3.

n_0 = A; DICH = {G}



i	T(i)	MO	DONG
		A_0	
A	B C D	B_5 C_3 D_6	A
C	A B E F D	B_4 D_6 E_7 F_{11}	A C
B	A C E	D_6 E_7 F_{11}	A C B
D	A C F G	E_7 F_9 G_{15}	A C B D
E	B C F	F_9 G_{15}	A C B D E
F	C D E G	G_{14}	A C B D E F
G			

Đường đi tìm được p: A → D → F → G. Chi phí của đường đi là 14.

6. Tìm kiếm cực tiểu sử dụng hàm đánh giá - Thuật toán A*

Đối với nhiều bài toán, việc tìm kiếm đường đi cực tiểu sẽ được định hướng tập trung xung quanh đường đi tốt nhất; nếu sử dụng các thông tin đặc tả về bài toán gọi là các heuristic.

Đối với việc tìm kiếm đường đi với chi phí cực tiểu, người ta sử dụng hàm đánh giá heuristic như sau:

Gọi g(n): giá cực tiểu đường đi từ $n_0 \rightarrow n$. Tại đỉnh n, g(n) xác định được.

Gọi h(n): giá cực tiểu đường đi từ n→DICH, h(n) không xác định được ⇒ người ta tìm cách ước lượng giá trị này.

Đặt **$f^0(n)=g^0(n)+h^0(n)$:** dự đoán chi phí cực tiểu của đường đi từ $n_0 \rightarrow$DICH có đi qua đỉnh n.

$g^0(n)$ là chi phí của đường đi từ đỉnh xuất phát đến đỉnh n tại thời điểm đang xét. $h^0(n)$ là ước lượng (dự đoán) chi phí đường đi từ đỉnh n đến đích. Việc chọn giá trị xấp xỉ $h^0(n)$ của h(n) không có một phương pháp tổng quát và được xem như một nghệ thuật. Giá trị này sẽ do các chuyên gia đưa ra.

Lúc này giải thuật tìm kiếm cực tiểu sẽ thay việc xét hàm g bởi hàm f. Tuy nhiên, người ta cũng chứng minh được 2 kết quả như sau:

Kết quả 1: Nếu $h^0(n)$ có tính chất: $0 \le h^0(n) \le h(n)\ \forall n$ và $c(u) > 0\ \forall u \in E$ thì thủ tục TKCT sử dụng hàm $f^0(n)$ để chọn phần tử trong MO ra xét (thay g(n)) sẽ cho đường đi từ $n_0 \to n_* \in DICH$ sao cho $g(n_*) = \min_{n \in DICH} g(n)$

Kết quả 2: Giả sử dùng 2 hàm ước lượng h_1^0 và h_2^0 thoả tính chất: $h_2^0(m) - h_2^0(n) \le h(m,n)$ (giá cực tiểu của đường đi từ m→n) và $\forall n \in N, 0 \le h_1^0(n) \le h_2^0(n) \le h(n)$. Khi đó $\#DONG_2 \le \#DONG_1$

Nhận xét:

$h^0 \equiv h \Rightarrow$ phương án tốt nhất

$h^0 \equiv 0 \Rightarrow$ **phương án tồi nhất**

Thuật toán A*

Input:

Đồ thị G = (V,E), Đỉnh xuất phát n_0

Hàm chi phí c: $E \to R^+$

c(i,j): xác định chi phí chuyển từ đỉnh i sang đỉnh j với (i,j) $\in$ E

h: V $\to$ R^+; h(n) xác định dự đoán chi phí tối ưu của đường đi từ đỉnh n đến đích. (ký hiệu h thay cho h^0, (tương tự g))

Tập các đỉnh đích DICH

Output:

Đường đi từ đỉnh n_0 đến đỉnh $n_* \in$ DICH

Procedure A* ;

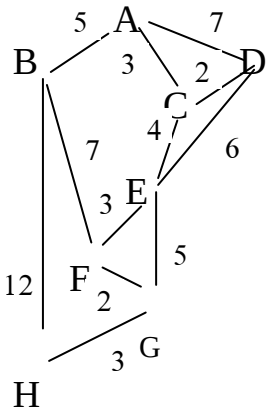
Begin

```
g(n0):= 0;
push(MO, n0);
While MO<>null do
    begin
        f(n) := min_{m∈MO} f(m)
        if n∈DICH then
            exit {xay dung duong di cuc tieu}
        push(DONG, n);
        if T(n) <>null then
            for  m∈T(n) do
                if m∉MO+DONG then
                    begin
                        push(MO,m);
                        tính f(m);
                        cha(m):=n;
                    end
                else
                    if f_mới(m) > f_cũ(n) then
                        begin
                            f(m):= f_mới(m);
                            cha(m):=n;
                        end;
    end;
    writeln('Khong  co duong di');
End;
```

Ví dụ 1. Cho đồ thị biểu diễn bài toán và giá trị dự đoán h^0 như sau:

n	A	B	C	D	E	F	G	H

$h^0(n)$	14	10	10	5	5	4	4	0



Tìm đường đi từ đỉnh A đến đỉnh H.

Trước tiên đỉnh A được đưa vào danh sách MO

g(A) = 0; h(A) = 14; f(A) = 14

Xét đỉnh A, (đưa A vào danh sách DONG) ta có các đỉnh kề B, C, D:

g(B) = 5; **f(B) = 15**; g(C) = 3; **f(C) = 13**; g(D) = 7; **f(D) = 12** → chọn đỉnh **D**.

Xét đỉnh D (đưa D vào danh sách DONG) có các đỉnh kề A, C, E. Đỉnh A đã ở trong danh sách DONG, ta tính lại f(C) và tính f(E):

f(C) không thay đổi; f(E) = g(D) +c(D,E) + H(E) = 7 + 6 + 5 = 18; **f(E) = 18**, chọn đỉnh **C**, có các đỉnh kề A, D, E. Tính lại **f(E) = 12**, chọn **E.** Các đỉnh kề của E là C, D, F, G. Tính **f(F) = 14; f(G) = 16**, chọn **F**. Các đỉnh kề của F là E, G, B và f(B), f(E), f(G) không đổi, chọn **B.** Các đỉnh kề của B là F, H. **f(H) = 17**, chọn **G**. Tính lại f(H) = 15 và dừng.

Đường đi tìm được là p: A → C → E → G → H với chi phí đường đi là 15

7. Phương pháp tìm kiếm leo đồi (hill-climbing search)

7.1. Kỹ thuật tìm kiếm leo đồi.

Tìm kiếm leo đồi là tìm kiếm theo độ sâu được hướng dẫn bởi hàm đánh giá. Song khác với tìm kiếm theo độ sâu, khi phát triển một đỉnh u thì bước tiếp theo ta chọn trong số các đỉnh con của u, đỉnh có hứa hẹn nhiều nhất để phát triển, đỉnh này được xác định bởi hàm đánh giá.

7.2. Giải thuật.

Input:

Đồ thị G = (V,E), đỉnh xuất phát n_0.

Hàm đánh giá h(n) đối với mỗi đỉnh n.

Tập đỉnh đích DICH

Output:

Đường đi từ đỉnh n_0 đến DICH

```
Procedure HLC; {Hill Climbing  Search}
begin
    Push(MO,n0);
    while  MO <> null do
    begin
        i = Pop(MO);
        if T(i) ∩ DICH <> null then
        begin
            L:= null;
            for  j ∈ T(i) do
                if j chưa xét then
                    đưa j vào danh sách L
            sắp xếp L theo thứ tự hàm đánh giá;
            chuyển danh sách L vào đầu danh sách MO;
    end;
    write('Khong co loi giai');
end;
```

7.3. Nhận xét.

Phương pháp tìm kiếm leo đồi chú trọng tìm hướng đi dễ dẫn đến trạng thái đích nhất. Cách đó được đưa ra nhằm làm giảm công sức tìm kiếm. Thuật

toán tìm kiếm leo đồi thực chất là thuật toán tìm kiếm theo chiều sâu, song tại mỗi bước ta sẽ ưu tiên chọn một trạng thái có hứa hẹn nhanh tới đích nhất để phát triển trước. Vấn đề quan trọng là biết khai thác kheo léo thông tin phản hồi để xác định hướng đi tiếp và đẩy nhnah quá trình tìm kiếm. Thông thường ta gán mỗi trạng thái của bài toán với một số đo (hàm đánh giá) nào đó nhằm đánh giá mức độ gần đích của nó. Điều đó có nghĩa là nếu trạng thái hiện thời là u thì trạng thái v sẽ được phát triển tiếp theo nếu v kề với u và hàm đanh giá của v đạt giá trị max (hoặc min).

Tuy nhiên phương pháp này không được cải thiện so với các phương pháp khác trong một số trường hợp sau:

- Cực trị địa phương: nút đang xét tốt hơn các nút lân cận, nhưng đó không phải là phương án tốt nhất trong toàn thể, ví vậy có thể phải quay lui về nút trước để đi theo hướng khác. Giải pháp này đòi hỏi ghi nhớ lại nhiều đường đi.
- Cao nguyên bằng phẳng: Các giá trị của các phương án như nhau, không xác định được ngay hướng nào là tốt hơn trong vùng lân cận.

7.4. Các ví dụ.

Ví dụ 1. Bài toán trò chơi 8 số**.**

trạng thái đầu

2	**8**	**3**
1	**6**	**4**
7		**5**

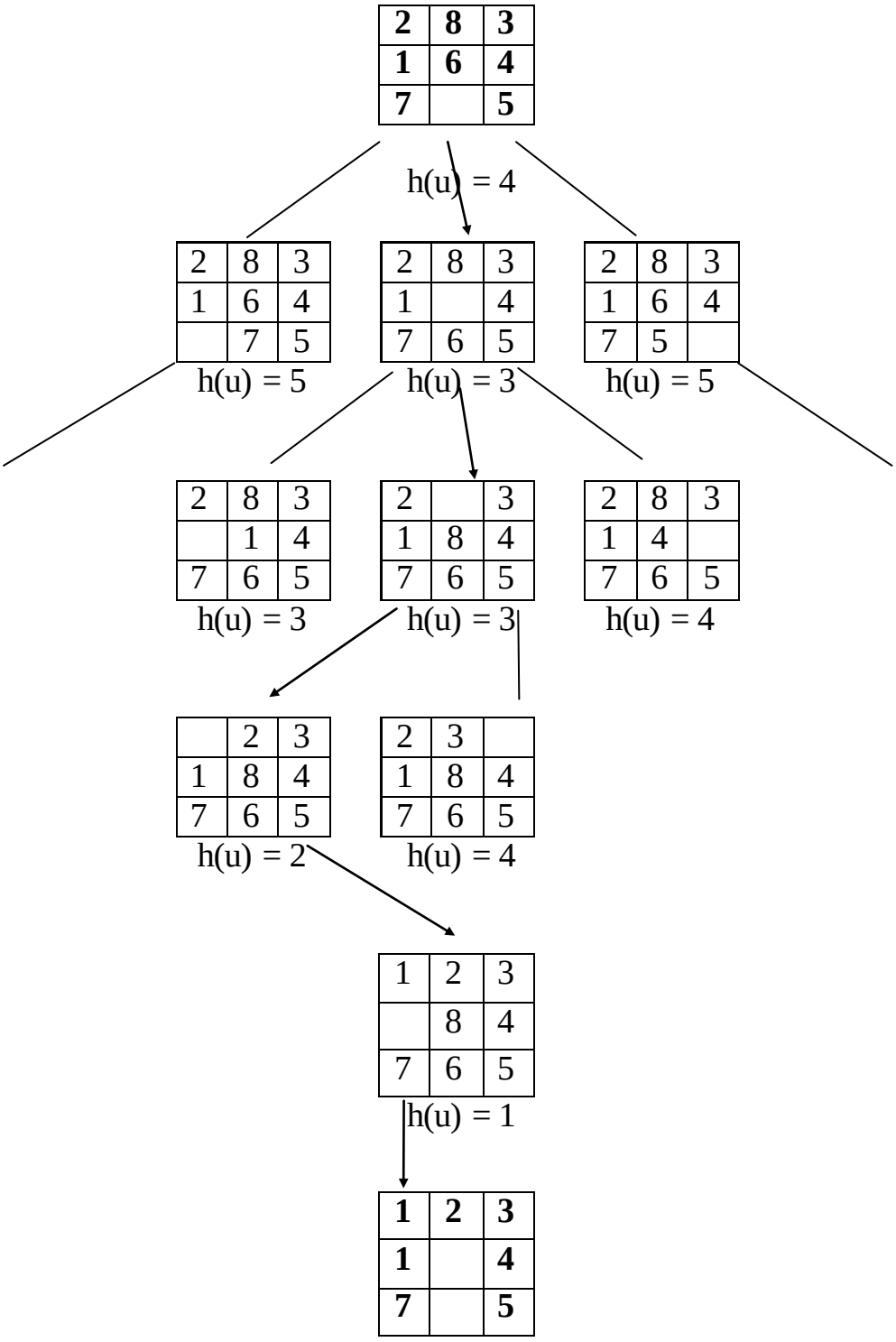
trạng thái đích

1	**2**	**3**
8		**4**
7	**6**	**5**

Trong bài toán này ta sử dụng hàm đánh giá, ký hiệu là h với ý nghĩa: h(u) cho biết số các chữ số trong trạng thái u không trùng với vị trí cú nó trong trạng thái đích. Trạng thái có tiềm năng dẫn đến đích nhanh nhất (được ưu tiên phát triển trước) là trạng thái có hàm đánh giá h đạt giá trị min..

Minh hoạ cây tìm kiếm cho trò chơi này theo giải thuật leo đồi ở trang sau

Trạng thái được chọn đi tiếp ở hướng mũi tên. Ở mức 3 chúng ta thấy có hai trạng thái cùng giá trị hàm đánh giá (= 3). Đây là trương hợp "cao nguyên băng phẳng" như nhận xét trên, nếu ta chọn phương án kia thì chắc chắn quá trình tìm kiếm sẽ khác đi nhiều. Trường hợp này dành cho độc giả.



8. Phương pháp sinh và thử.

Chiến lược này đơn giản, gồm ba bước:

- Trước hết tạo ra một giải pháp. Trong vài bài toán cụ thể đó là việc chọn một lời giải trong không gian các lời giải hay tạo ra một đường đi.
- Thứ hai, thử xem lời giải đó có thích hợp không bằng cách so sánh phương án khác hay so sanh với điểm cuối cần suy diễn.
- Tiếp theo, nếu lời giải đạt được thì dừng, ngược lại, lặp lại từ bước đầu với nút khác.

Với phương pháp này nếu bài toán có llời giải thì sẽ đưa đến đích. Tuy nhiên kích thước bài toán lớn sẽ tăng khối lượng tính toán. Việc tạo lời giải ban đầu có thể thực hiện ngẫu nhiên, và cũng hy vọng ngẫu nhiên mà đạt được lời giải, bởi vậy, không thể không tính đến chỉ một vài hướng đi được cảm nhận là tốt, và loại trừ trước các hướng không dẫn đến lời giải.

Ví dụ1. Tìm số có 6 chữ số mà tổng bình phương các chữ số chia hết cho 3.

Giai đoạn sinh: tạo ra số có 6 chữ số và ta gọi các chữ số từ trái qua phải lần lượt là a, b, c, d, e, f thì $0 < a <= 9$, $0 <= b, c, d, e, f <= 9$.

Giai đoạn thử: nểu a*a + b*b + c*c + d*d + e*e + f*f chia hết cho 3 thì chon, ngược lại, tạo ra số khác.

Ví dụ 2. Một xâu nhị phân được gọi là thưa nếu trong xâu không có hai chữ số 1 đứng kề nhau. Tìm xâu nhị phân thưa có chiều dài n.

Giai đoạn sinh: Tạo ra một xâu nhị phân S có chiều dài n.

Giai đoạn thử: Kiểm tra có phải xâu thưa không? (Pos(‘11’,S) = 0).

Trong hai ví dụ trên, sinh viên có thể lập trình để tìm tất cả các lời giải của bài toán, chẳng hain tìm tất cả các xâu nhị phân thưa có chiều dài n cho trước.

Ví dụ 3. Một bệnh nhân có một vài triệu chứng, chẳng hạn: sốt cao về buổi chiều, ho và mệt mỏi ,…. Bác sĩ có chẩn đoán nghi bị lao phổi, người ta sẽ cho làm ngay xét nghiệm, nếu đúng là dương tính thì kết luận và điều trị bệnh lao

phổi, ngược lai, băt buộc bác sĩ phải chuyển hướng suy nghĩ sang một bệnh khác, v.v…

9. Phương pháp thoả mãn ràng buộc.

Phương pháp thoả mãn ràng buộc hỗ trợ cho phương pháp sinh và thử, khi chú ý tới một số ràng buộc áp đặt lên các nút trong không gian bài toán. Mục đích đặt ra là xác định đường đi trong đồ thị không gian bài toán, đường đi từ trạng thái đầu đến trạng thái cuối đáp ứng một vài ràng buộc nào đó. Do vậy quá trình tìm kiếm lời giải bao gồm hai phần liên quan chặt chẽ với nhau:

- Tìm kiếm trong không gian các ràng buộc.
- Tìm kiếm trong không gian các bài toán ban đầu.

Nội dung của phương pháp như sau: Thực hiện các bước từ a) đến e) dưới đây cho đến khi tìm được lời giải đày đủ của bài toán hoặc tất cả các đương đều đã duyệt qua nhưng không cho kết quả.

a) Chọn một đỉnh chưa được xét trong đồ thị tìm kiếm.
b) Áp dụng các luật suy diễn trên các ràng buộc đối với đỉnh đã chọn để tạo ra tập các ràng buộc mới.
c) Nếu tập các ràng buộc mới có mâu thuẫn thì đưa ra thông báo đường đi hện thời tới nút đang xét dẫn tới bế tắc.
d) Nếu tập ràng buộc mô tả lời giải đầy đủ của bài toán thì dừng và đưa ra thông báo thành công. Ngược lai, sang bước sau.
e) Áp dụng các luật biến đổi không gian trạng thái tương ứng để tạo ra lời giải bộ phận, tương hợp với tập các ràng buộc hiện thời. Thêm các lời giải bộ phận này vào đồ thị tìm kiếm.

Ví dụ. Xét bài toán điền các chữ số phân biệt thay cho các chữ cái S, E, N, D, M, O, R, Y sao cho phép cộng sau là đúng:

```
  SEND
+ MORE
------
 MONEY
```

Các ràng buộc ban đầu:

- Các chữ cái khác nhau không nhận cùng một giá trị.

- Các ràng buộc số học (cộng có nhớ hoặc không có nhớ.

Gọi C1, C2, C3, C4 lần lượt lá số nhớ của các cột từ phải sang trái. Khi đó ta xây dựng các ràng buộc cụ thể như sau:

E, N, D,O, R, Y thuộc tập {0 ..9} (1)

S, M thuộc tập {1..9} (2)

C1, C2, C3, C4 thuộc tập {0,1} (3)

D + E = Y + 10*C1 (4)

N + R + C1 = E + 10*C2 (5)

E + O + C2 = N + 10*C3 (6)

S + M + C3 = O + 10*C4 (7)

M = C4 (8)

Từ ràng buộc (2) và (8) suy ra **M = 1** và **C4=1** (9)

Từ ràng buộc (7) và (9) suy ra S +C3 = O + 9, lúc này có hai phương án để lựa chọn:

Phương án 1:

C3 = 0, khi đó ta có S = O + 9, như vậy **S = 9** và **O = 0** (10-1)

Từ ràng buộc (6) ta có E + C2 = N, suy ra **C2 = 1** và

E + 1 = N (11-1)

Từ ràng buộc (5), ta có R + C1 = 9, như vậy **R = 8** và **C1 = 1** (12-1) (do kết hợp với các ràng buộc (2) và (10-1).

Từ ràng buộc (4) ta có D + E = Y +10. Đến bước này ta có thể khẳng định các giá trị của D, E, Y chỉ có thể nhận trong tập {2, 3, 4, 5, 6, 7}. Ngoài ra D + E >= 12. Vì vậy chỉ có các khả năng sau có thể xãy ra:

- D = 5 và E = 7
- D = 7 và E = 5 (hai trường hợp này Y = 2)
- D = 6 và E = 7
- D = 7 và E = 6 (hai trường hợp sau Y = 3)

Xét khả năng thứ nhất. Từ ràng buộc (11-1) ta suy ra N = 8 mâu thuẫn với (12-1) nên bị loại

Xét khả năng thứ hai. Từ ràng buộc (11-1) ta suy ra N= 6. Kiểm tra điều kiện bài toán đều thoả mãn. Vậy ta có nghiệm là:

S = 9, E = 5, N= 6, D = 7, M= 1, O = 0, R = 8, Y = 2.

Xét khả năng thứ ba. Từ ràng buộc (11-1) suy ra N = 8 mâu thuẫn với (12-1)

Xét khả năng thứ tư. Từ ràng buộc (11-1) suy ra N=7 = D mâu thuẫn.

Phương án 2.

C3 = 1. Từ ràng buộc (7) ta có S = O + 8, suy ra **S = 8** và **O = 0**(10-2) (vì M=1 và S <= 9).

Từ ràng buộc (6) ta có E = N +10 mâu thuẫn với ràng buộc (1). vậy phương án 2 không có lời giải.

10.Cài đặt một số giải thuật.

Một số quy ước:

- Giả sử đồ thị G được cho bởi ma trận kề A.
- Các danh sách MO và DONG được lưu trong cùng một mảng, với các chỉ số riêng.
- Mảng logic Dau dùng để đánh dấu các đỉnh đã xét (nằm trong danh sách DONG

10.1. Tìm kiếm rộng.

- Danh sách MO và DONG được lưu trong mảng Q, d và c là chỉ số của phần tử đầu và cuối của queue Q.
- V = { 1..n}
- Thủ tục Duyet_rong(i) đánh dấu tất cả các đỉnh từ i có thể đến được đỉnh đó.

Procedure Khoitao;

Begin

```
        Fillchar (Dau, n, True);
End;
Procedure Duyet_rong (i:byte);
Var    Q: array [1..100] of byte;
                d, c, j, k: byte;
Begin
        d:=1; {Khởi tạo hàng đợi rỗng }
        c:=1;
        Q[c]:= i;
        Dau[i]:= false;
        While d<=c do
                begin
                        j:= Q[d];
                        inc(d);
                        for k:=1 to n do
                                if (A[j,k]=1) and Dau[k] then
                                        begin
                                                inc(c);
                                                Q[c]:=k;
                                                Dau[k]:=false;
                                        end;
                end;
End;
```

Ví dụ 1. Tìm đường đi từ đỉnh i_0 đến đỉnh j_0 của đồ thị G.

Dữ liệu được lưu vào file Text có cấu trúc như sau:

- Dòng đầu tiên chứa 3 số n, i_0, j_0 (n là số đỉnh của đồ thị)
- n dòng tiếp theo lần lượt chứa giá trị n dòng của ma trận A.

Tên file được nhập từ bàn phím khi thực hiện chương trình.

Giá trị của mảng Truoc tạ vị trí j Truoc[j] xác định đỉnh đứng trước j trong đường đi tìm được.

```
Program đuongdi;
Var
    A: array[1..50,1..50] of byte;
    Dau: array[1..50] of boolean;
    Truoc: array[1..50] of byte;
    n, i0, j0: byte;
Procedure khoitao;
Var
    i, j : byte;
    f:text;
    tenfile:string;
Begin
    write('ten file');
    readln(tenfile);
    assign(f, tenfile);
    reset(f);
    readln(f,n,i0,j0);
    for i:=1 to n do
        begin
            for j:=1 to n do
                read(f, A[i,j]);
            readln(f);
        end;
    close(f);
    Fillchar (Dau,n,true);
End;
```

```
Procedure  BFS(i:byte);
Var
        Q: array [1..50] of byte;
        d,c,j,k: byte;
Begin
        d:=1;
        c:=1;
        Q[c]:=i;
        Dau[i]:=false;
        Truoc[i] := 0;
        While d<=c do
        begin
                j:=Q[d];
                inc(d);
                for  k:=1 to n do
                        if  (A[j,k]=1) and Dau[k] then
                        begin
                                inc(c) ;
                                Q[c]:=k;
                                Dau[k]:=false;
                                Truoc[k] := j;
                        end;
      end;
End;
Procedure inkq(j: byte);
Begin
        if Truoc[j] <> 0 then
                inkq(truoc[j]);
```

```
        write(j:4);
End;
Procedure Duyet;
Var  i:byte;
Begin
        BFS(i0);
        if Dau[j0] then
                inkq(j0)
        else
                writeln(‘  Khong co duong di’);
End;
BEGIN {main}
        Khoitao;
        Duyet;
        Readln;
END.
```

Ví dụ 2. Tìm số thành phần liên thông của một đồ thị .

Dữ liệu được lưu vào file Text có cấu trúc như sau:

- Dòng đầu tiên chứa số n (số đỉnh của đồ thị)
- n dòng tiếp theo lần lượt chứa giá trị n dòng của ma trận A.
- Tên file được nhập từ bàn phím khi thực hiện chương trình.

```
Program lienthong;
Var
        A: array[1..50,1..50] of byte;
        Dau: array [1..50] of boolean;
        n, So:byte;
Procedure khoitao;
```

```
Var
        i, j : byte;
        f:text;
        tenfile:string;
Begin
        write('ten file');
        readln(tenfile);
        assign(f, tenfile);
        reset(f);
        readln(f,n);
        for i:=1 to n do
                begin
                        for j:=1 to n do
                                read(f, A[i,j]);
                        readln(f);
                end;
        close(f);
        Fillchar (Dau,n,true);
        So:=0;
End;
Procedure BFS(i:byte);
Var
        Q: array [1..50] of byte;
        d,c,j,k: byte;
Begin
        d:=1;
        c:=1;
        Q[c]:=i;
```

```
        Dau[i]:=false;
        While d<=c do
        begin
                j:=Q[d];
                inc(d);
                for  k:=1 to n do
                        if  (a[j,k]=1) and Dau[k] then
                        begin
                                inc(c) ;
                                Q[c]:=k;
                                Dau[k]:=false;
                        end;
      end;
End;
Procedure  Duyet;
Var
        i:byte;
Begin
        For  i:=1 to n do
                If  Dau[i] then
                        begin
                                inc(So);
                                BFS (i);
                        end;
        writeln('So thành phần liên thông:', So);
End;
```

```
BEGIN {main}
        Khoitao;
        Duyet;
        Readln;
END.
```

10.2. Tìm kiếm sâu.

Với giả thiết như duyệt rộng, do MO hoạt động như stack nên dùng thủ tục đệ quy.

```
Procedure  DFS (i:byte);  {Depth First Search}
{Xuất phát từ đỉnh i, đánh dấu các đỉnh được xét khi tìm kiếm theo chiều sâu}
Var
    j: byte;
Begin
    Dau[i]:=false;
    For  j:=1 to n do
        If  (a[i,j]=1)  and Dau[j] then
            DFS (j);
End;
```

Ví dụ 1. Tìm đường đi từ đỉnh i0 đến đỉnh j0.

```
Program  Duong_di;
Var
    A: array[1..50,1..50] of byte;
    Truoc: array [1..50] of byte;
    Dau: array [1..50] of boolean;
    n, i0, j0: byte;
Procedure  Khoitao;
Var
    i, j : byte;
    f:text;
    tenfile:string;
Begin
    write('ten file');
```

```
        readln(tenfile);
        assign(f, tenfile);
        reset(f);
        readln(f,n,i0,j0);
        for  i:=1 to n do
                begin
                        for  j:=1 to n do
                                read(f, a[i,j]);
                        readln(f);
                end;
        close(f);
        Fillchar  (Dau,n,true);
End;

Procedure  DFS (i:byte);
Var
        j: byte;
Begin
        Dau[i]:=false;
        For  j:=1 to n do
                If  (a[i,j]=1)  and Dau[j] then
                        DFS (j);
End;
Procedure inkq(j: byte);
Begin
        if Truoc[j] <> 0 then
                inkq(truoc[j]);
        write(j:4);
```

```
End;
Procedure duyet;
Begin
        DFS(i0);
        if  dau[j0] then
                inkq(j0)
        else
                writeln('Khong  co duong di');
End;

BEGIN {main}
        Khoitao;
        Duyet;
        Readln;
END.
```

Ví dụ 2. Tìm tất cả hoán vị của (1,2,...n)

```
Program hoanvi;
Var
        A:array  [1..50] of byte;
        n:byte;
        Dau: array [1..50] of boolean;

Procedure Khoitao;
Begin
        write('n = ');
        readln(n);
        Fillchar(Dau,n,  true);
```

```
End;
```

```
Procedure DFS (i:byte);
Begin
    if i<n then
        for j:=1 to n do
            if Dau[j] then
                begin
                    A[i]:=j;
                    Dau[j]:=false;
                    DFS (j);
                    Dau[j]:=true;
                end
    else
        begin
            j:=1;
            While not Dau[j] do
                inc (j);
            a[n]:=j;
            begin
                for j:=1 to n do
                    write (A[j]:4);
                witeln;
            end;
        end;
End;
BEGIN {main}
    Khoitao;
    DFS (1);
    Readln;
```

END.

10.3. Thuật toán AT – Tìm kiếm cực tiểu.

Giả thiết dữ liệu lưu trữ như tìm kiếm rộng.

cp là mảng chi phí của đồ thị G.

Đồ thị G được lưu trữ bởi ma trận chi phí cp, trong đó cp[i,j] = Vocung có nghĩa là không có cung (i,j).

```
Procedure ddcuctieu;
Const
    Vocung = 70000;
Var
    A: array[1..50,1..50] of byte;
    Truoc: array [1..50] of byte;
    Dau: array [1..50] of boolean;
    cp: array[1..50, 1..50] of word;
    n, i0, j0: byte;

Procedure Khoitao;
Var
    i, j : byte;
    f:text;
    tenfile:string;
Begin
    write('ten file');
    readln(tenfile);
    assign(f, tenfile);
    reset(f);
    readln(f,n,i0,j0);
```

```
        for  i:=1 to n do
                begin
                        for  j:=1 to n do
                                read(f, cp[i,j]);
                        readln(f);
                end;
        close(f);
        Fillchar  (Dau,n,true);
End;

Procedure inkq(j: byte);
Begin
        if Truoc[j] <> 0 then
                inkq(truoc[j]);
        write(j:4);
End;

Procedure  Timkiem;
Begin
        g[i0]:=0;
        truoc[i0]:=0;
        d:=1;  c:=1;
        Q[c]:=i0;
        While  d<=c do
                begin
                        k:=d;
                        for  l:=d+1 to c do
                                if  g[q[l]]<g[q[k]] then
```

```
                    k:=l;
                tam:=q[d];
                q[d]:=q[k];
                q[k]:=tam;
                m:=q[d];
                inc(d);
                if m=j0 then
                    inkq(j0)
                else
                    for  l:=1 to n do
                        if (cp[m,l]< vocung) then
                            if dau[l] then
                                begin
                                    inc(c);
                                    q[c]:=l;
                                    dau[l]:=false;
                                    g[l]:=g[m]+cp[m,n];
                                    truoc[l]:=m;
            end
                            else
                                if g[l]>g[m]+cp[m,n] then
                                    begin
                                        g[l]:= g[m]+cp[m,n];
                                        truoc[l]:=m;
                                    end;
    end;
End;
```

Begin

Khoitao;

Timkiem;

End;

10.4. Tìm kiếm leo đồi.

Trong chương trình cài đặt này, chúng ta quy ước nếu đỉnh đang xét là đỉnh u, thì **đỉnh kề với u có khả năng đến đích nhất là đỉnh có khoảng cách với u lớn nhất.**

Khi đó giải thuật leo đồi có thể trình bày lại như sau.

Leodoi(i,j): Thực hiện giải thuật leo đồi từ đỉnh i đến đỉnh j.

- Nếu (i, j) $\in$E : d=c[i,j], push(i,j,k), exit
- Nếu (i,j) $\notin$E: Tìm k sao cho c[i,k]=max {c[l,k]/ l$\in$T[i] and dau[i,l]}:

Nếu có (d=c[l,k]): dau[i,l]=false, push(i,j,d), Leodoi(k,j)

Ngược lại (d=0): pop(k,j,d), leodoi(k,j)

Dữ liệu được thiết kế như sau:

- Mảng A lưu danh sách các cung của đồ thị G
- S là stack lưu danh sách các đỉnh sẽ được xét và Top là đỉnh của S
- i0, j0 là đỉnh xuất phát và đỉnh kết thúc
- Toàn bộ thông tin được lưu trong file dạng Text có cấu trúc như sau: dòng đầu lưu m (số cung của đồ thị), i0, j0; m dòng tiếp theo mỗi dòng chứa thông tin của mộtcung đồ thị G (đỉnh đầu, đỉnh cuối và độ dài cung).

Procedure Leodoi;

Type

cung = record

dau, cuoi: byte;

```
                kc: word;
            end;
Var
    S, A: array[1..50] of cung;
    B: array[1..50] of boolean;
    m,i0,j0, Top: byte;
Procedure Khoitao;
Var
    f: text;
    l: byte;
    d: word;
    tenfile: string;
begin
    write('Nhap ten file: ');
    readln(tenfile);
    assign(f,tenfile);
    reset(f);
    readln(f,m,i0,j0);
    for l:=1 to m do
        with A[l] do
            readln(f,dau, cuoi, kc);
    fillchar(B, l, false);
    Top:= 0;
end;
```

Procedure Pop(Var i,j: byte; var d: word);

{Lấy một bản ghi (i,j,d) từ S}

```
begin
```

```
        with S[Top] do
                begin
                        i:= dau;
                        j:= cuoi;
                        d:= kc;
                end;
        dec(Top);
end;
```

Procedure TimKiem(i: byte; Var j: byte; var d: word);

{ Tìm cung (i,j) có c[i,j] lớn nhất, nếu có thì d = c[i,j] và đánh dấu cung ơi,jư là true, ngược lại d = 0 }

```
Var
        l,p: byte;
begin
        d:=0;
        for l:= 1 to m do
                if (A[l].dau = i) and (A[l].kc > d) and not B[l] then
                        begin
                                j:= A[l].cuoi;
                                p:= l;
                                d:= A[l].kc;
                        end;
                B[l]:= true;
end;
```

Function DenDich(i,j: byte; var d:word): boolean;

```
Var
```

```
        l: byte;
begin
        for l:= 1 to m do
                if (A[l].dau = i) and (A[l].cuoi = j) then
                        begin
                                d:= A[l].kc;
                                DenDich:= true;
                        end
                else
                        begin
                                DenDich:= false;
                                d:= 0;
                        end;
end;
```

Procedure Inkq(j:byte);

```
Var
        d:word;
        k: byte;
begin
        d:=0;
        for k:= 1 to Top do
                begin
                        write(S[k].dau);
                        d:=d + S[k].kc;
                end;
        writeln(j);
        writeln(‘  Chi phi: ‘,d);
```

```
end;

Procedure Duongdi(i,j: byte);
Var
        k,d: byte;
Begin
        if Dendich(i,j) then
                begin
                        push(i,j,d);
                        inkq(j);
                        exit;
                end;
        Timkiem(i,k,d);
        if d > 0 then
        begin
                push(i,j,d);
                duongdi(k,j);
        end
        else
                if Top > 0 then
                        begin
                                pop(i,j,d);
                                duongdi(i,j);
                        end
                else
                        writeln('Khong co duong di');
end;
```

```
Begin {leo doi}
        Khoitao;
        Duongdi(i0,j0);
end;
```

11. Bài tập.

Bài tập 1. Cho ma trận kề A= (a_{ij}) biểu diễn một đồ thị vô hướng G = (V,E) dưới đây:

$$\begin{pmatrix} 0 & 4 & \infty & \infty & \infty & \infty \\ 4 & 0 & 7 & \infty & \infty & 3 \\ \infty & 7 & 0 & 3 & 8 & 6 \\ \infty & \infty & 3 & 0 & 5 & \infty \\ \infty & \infty & 8 & 5 & 0 & 2 \\ \infty & 3 & 6 & \infty & 2 & 0 \end{pmatrix}$$

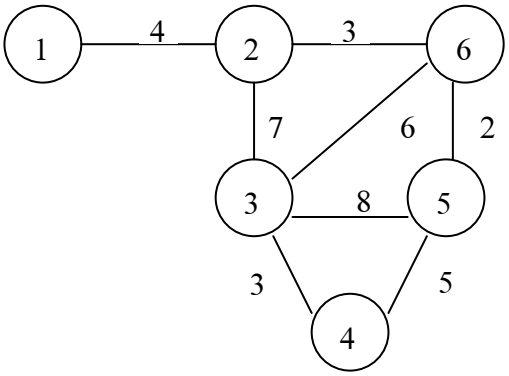
trong đó $a_{ij}= \infty$ nếu $(i,j) \notin E$, ngược lại a_{ij} là chi phí để đi từ đỉnh i sang đỉnh j.

a. Hãy tìm đường đi từ đỉnh 1 sang đỉnh 4 theo các phương pháp tìm kiếm rộng và tìm kiếm sâu.

b. Tìm đường đi ngắn nhất từ đỉnh 1 sang đỉnh 4

LỜI GIẢI

- **Vẽ đồ thị G được biểu diễn bởi ma trận kề A ở trên**



- **Phương pháp duyệt rộng**

n	**t(n)**	**↑open ↓**	**close**
		1	
1	2	2	1
2	1, 3, 6	3, 6	1, 2
3	2, 4, 5, 6	6, 4, 5	1, 2, 3
6	2, 3, 5	4, 5	1, 2, 3, 6
4 ∈dich→dừng			

Đường đi từ đỉnh 1 đến đỉnh 4 theo phương pháp duyệt rộng là: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$

- **Phương pháp duyệt sâu**

n	**t(n)**	**open ↑↓**	**close**
		1	
1	2	2	1
2	1, 3, 6	3, 6	1, 2
6	2, 3, 5	3, 5	1, 2, 6
5	3, 4, 6	3, 4	1, 2, 6, 5
4 ∈dich→dừng			

Đường đi từ đỉnh 1 đến đỉnh 4 theo phương pháp duyệt sâu là: $1 \rightarrow 2 \rightarrow 6 \rightarrow 5 \rightarrow 4$

- **Phương pháp tìm kiếm cực tiểu**

n	**t(n)**	**open**	**close**
		1_0	
1	2	2_4	1
2	1, 3, 6	3_{11}, 6_7	1, 2
6	2, 3, 5	3_{11}, 5_9	1, 2, 6
5	3, 4, 6	3_{11}, 4_{14}	1, 2, 6, 5
3	2, 4, 5, 6	4_{14}	1, 2, 6, 5, 3
4∈DICH →			

dừng			

VẬY ĐƯỜNG ĐI NGẮN NHẤT: 1→ 2→ 6→ 5→ 4 VỚI CHI PHÍ 14

BÀI TẬP 2. N**GƯỜI TA SỬ DỤNG HAI BÌNH CHỨA CÓ DUNG TÍCH LẦN LƯỢT LÀ 3(LÍT) VÀ 4(LÍT) ĐỂ ĐONG 2(LÍT) NƯỚC. GIẢ SỬ LƯỢNG NƯỚC ĐƯỢC LẤY TỪ VÒI KHÔNG HẠN CHẾ VÀ CÔNG ĐỂ LẤY NƯỚC TỪ VÒI CHO ĐẦY MỘT BÌNH LÀ 3, CÔNG ĐỂ ĐỔ NƯỚC TRONG MỘT BÌNH RA NGOÀI LÀ 2 VÀ ĐỔ NƯỚC TỪ BÌNH NÀY SANG BÌNH KHÁC THÌ TỐN CÔNG LÀ 5.**

HÃY CHỈ RA QUÁ TRÌNH TÌM KIẾM LỜI GIẢI BẰNG PHƯƠNG PHÁP TÌM KIẾM THEO CHIỀU RỘNG VÀ TÌM KIẾM LEO ĐỒI.

Lời giải

- **Phương pháp tìm kiếm theo chiều rộng**

n	**t(n)**	**↑open ↓**	**close**
		(0,0)	
(0,0)	(0,4), (3,0)	(0,4), (3,0)	(0,0)
(0,4)	(0,0), (3,4), (3,1)	(3,0), (3,4), (3,1)	(0,0), (0,4)
(3,0)	(0,0), (3,4), (0,3)	(3,4), (3,1), (0,3)	(0,0),(0,4),(3,0)
(3,4)	(0,4), (3,0)	(3,1), (0,3)	(0,0),(0,4),(3,0), (3,4)
(3,1)	(0,1), (3,0), (3,4),	(0,3), (0,1)	(0,0),(0,4),(3,0), (3,4), (3,1)
(0,3)	(0,4) (0,0), (0,4), (3,3), (3,0)	(0,1), (3,3)	(0,0),(0,4),(3,0), (3,4), (3,1), (0,3)
(0,1)	(0,0), (3,1), (0,4), (1,0)	(3,3), (1,0)	(0,0),(0,4),(3,0), (3,4), (3,1), (0,3), (0,1)
(3,3)	(0,3), (3,0), (2,4)	(1,0), (2,4)	(0,0),(0,4),(3,0), (3,4), (3,1), (0,3), (0,1), (3,3)

(1,0) (2,4)	(0,0), (3,0), (1,4), (0,1)	(2,4), (1,4),	(0,0),(0,4),(3,0), (3,4), (3,1), (0,3), (0,1), (3,3), (1,0)

QUÁ TRÌNH ĐONG NƯỚC THEO PHƯƠNG PHÁP DUYỆT RỘNG LÀ:

(0,0)→(3,0)→(0,3)→(3,3)→(2,4)

- **Phương tìm kiếm leo đồi** (giả thiết đỉnh kề với đỉnh đang xáe và có khoảng cách đên đỉnh đó lớn nhất là đỉnh có triển vọng đến đích nhất)

((0,0), (2,4))$\notin$E $\rightarrow$ k= (3,0) c[(0,0), (3,0)]=5

((3,0), (2,4)) $\notin$E $\rightarrow$ k= (0,3) c[(3,0), (0,3)]=5

((0,3), (2,4)) $\notin$e $\rightarrow$ k= (3,3) c[(0,3), (3,3)]=5

((3,3), (2,4)) $\in$E $\rightarrow$ dừng c[(3,3), (2,4)]=5

QUÁ TRÌNH ĐONG NƯỚC THEO PHƯƠNG PHÁP LEO ĐỒI LÀ:

(0,0)→(3,0)→(0,3)→(3,3)→(2,4) VỚI CHI PHÍ 20

Bài tập 3. Đại dương được xem như là một mặt phẳng toạ độ trên đó có n hòn đảo với toạ độ lần lượt là (x_1, y_1), (x_2, y_2), …, (x_n, y_n). Một chiếc ca nô xuất phát từ đảo d_1 muốn tuần tra đến đảo d_2. bình xăng của ca nô chỉ chứa đủ xăng để đi được một quãng đường dài không quá m (km). Trên đường đi ca nô có thể ghé một số đảo nào đó để tiếp thêm xăng, lúc này ca nô được tiếp thêm xăng đầy bình chứa. Hãy chỉ ra một đường đi từ đảo d_1 đến đảo d_2 sao cho số lần ghé đảo trung gian để tiếp thêm xăng là ít nhất.

HƯỚNG DẪN

TA XEM HAI ĐẢO LÀ KỀ NHAU NẾU KHOẢNG CÁCH GIỮA CHÚNG KHÔNG VƯỢT QUÁ M (KM). BÀI TOÁN CẦN TÌM ĐƯỜNG ĐI TỪ ĐẢO D_1 ĐẾN ĐẢO D_2 THÔNG QUA CÁC ĐẢO KỀ NHAU. THUẬT TOÁN TÌM KIẾM THEO CHIỀU RỘNG CHO PHÉP TÌM ĐƯỜNG TÌM RA ĐƯỜNG ĐI NỐI HAI ĐẢO QUA ÍT CẠNH TRUNG GIAN NHẤT (TỨC LÀ ÍT ĐẢO TRUNG GIAN NHẤT).

DỮ LIỆU VÀO LƯU TRONG FILE DẠNG TEXT, DÒNG ĐẦU CHỨA SỐ ĐẢO N, DÒNG THỨ HAI CHỨA KHOẢNG CÁCH LỚN NHẤT CANO CÓ THỂ ĐI LIÊN TỤC, N DÒNH TIẾP THEO MỖI DÒNG CHỨA HAI GIÁ TRỊ TƯƠNG ỨNG VỚI TOẠ ĐỘ CỦA MỖI ĐẢO.

BÀI TẬP 4**. MỘT MẠNG LƯỚI GIAO THÔNG GIỮA N THÀNH PHỐ (CÁC THÀNH PHỐ ĐƯỢC ĐÁNH SỐ TỪ 1 ĐẾN N) ĐƯỢC CHO BỞI MA TRẬN $A=(A_{IJ})_{N*N}$, TRONG ĐÓ:**

$$\begin{cases} \mathbf{0,\ NẾU\ KHÔNG\ CÓ\ ĐƯỜNG\ ĐI\ TRỰC\ TIẾP\ TỪ\ I\ ĐẾN\ J} \end{cases}$$

A_{IJ}= 1, NẾU CÓ ĐƯỜNG ĐI TRỰC TIẾP TỪ I ĐẾN J VÀ LÀ ĐƯỜNG ĐI AN TOÀN

2, NẾU CÓ ĐƯỜNG ĐI TRỰC TIẾP TỪ I ĐẾN J NHƯNG PHẢI QUA MỘT CHẶNG ĐƯỜNG NGUY HIỂM

QUY ƯỚC: A_{II}=1, ∀I =1..N

CHO TRƯỚC HAI THÀNH PHỐ I_0, I_1. HÃY TÌM MỘT ĐƯỜNG ĐI TỪ I_0 ĐẾN I_1 SAO CHO SỐ CHẶNG ĐƯỜNG NGUY HIỂM PHẢI ĐI QUA LÀ ÍT NHẤT.

HƯỚNG DẪN: **TRƯỚC HẾT PHẢI XÁC ĐỊNH ĐỒ THỊ BIỂU DIỄN BÀI TOÁN. Ở ĐÂY DỄ THẤY RẰNG MỖI THÀNH PHỐ TƯƠNG ỨNG VỚI MỘT ĐỈNH CỦA ĐỒ THỊ, VẤN ĐỀ CHỈ CÒN XÁC ĐỊNH TẬP CUNG E CĂN CỨ VÀO GIẢ THIẾT CỦA BÀI TOÁN.**

Bài tập 5. Cho bảng vuông gồm m*n ô. Trên mỗi ô ghi số 0 hay 1.

a. Từ một ô nào đó có thể chuyển sang ô chứa số 1 có chung cạnh với nó. giả sử đang ở ô (h,c). Hãy tìm xem có cách di chuyển từ ô này ra một ô ở mép bảng hay không? Tìm cách chuyển qua ít ô nhất.

b. Một miền của bảng là tập hợp các ô có chung cạnh và có cùng giá trị. hãy đếm xem bảng có bao nhiêu miền. miền lớn nhất có bao nhiêu ô.

c. Cho phép thay đổi giá trị tất cả các ô trong cùng một miền. Hãy xác định miền cần thay đổi để số miền giảm nhiều nhất.

d. Hãy xác định miền cần thay đổi để thu được một miền mới lớn nhất.

***Hướng dẫn*:** Mỗi ô tương ứng với một đỉnh của đồ thị. Hai đỉnh kề nhau khi và chỉ khi hai ô tương ứng có thể chuyển sang nhau. Mỗi miền của bảng tương ứng với một miền liên thông của đồ thị.

Bài tập 6. Lập chương trình đối với bài toán đong nước, với các số m, n, k là các số dương bất kỳ được nhập từ bàn phím khi thực hiện chương trình.
Hướng dẫn: Sử dụng thuật toán tìm kiếm rộng sẽ cho số lần thao tác là ít nhất.

Bài tập 7. Một toà lâu đài được mô tả bằng một hình chữ nhật có m*n ô. Giữa các ô có một số bức tường ngăn cách chia lâu đài thành các phòng. Như vậy, mỗi phòng tương ứng với tập các ô thông nhau. Tại ô (i,j), cho biết thông tin có tường ngăn giữa ô này với bốn ô kề với nó không bởi giá trị a_{ij} là một số nhị phân 4 chữ số tương ứng ô (i,j) có (1) hoặc không có (0) tường ở phía Tây, Bắc, Đông, Nam. Ví dụ a_{ij} = 1001 có nghĩa là ô (i,j) có tường ở phía Tây và Nam, nhưng không có tường ở phía Bắc và Đông. Hãy viết chương trình thực hiện các yêu cầu sau:

A. **ĐẾM SỐ PHÒNG CỦA TOÀ LÂU ĐÀI.**
B. **CHO BIẾT PHÒNG LỚN NHẤT CÓ DIỆN TÍCH LÀ BAO NHIÊU Ô.**
C. **CHO BIẾT NÊN PHÁ BỨC TƯỜNG NGĂN HAI PHÒNG NÀO ĐỂ ĐƯỢC MỘT PHÒNG MỚI CÓ DIỆN TÍCH LỚN NHẤT.**

HƯƠNG DẪN: **GIÁ TRỊ A_{IJ} CÓ THỂ NHẬN TƯƠNG ỨNG VỚI SỐ THẬP PHÂN TỪ 0 ĐẾN 15. VÌ VẬY TA LƯU DỮ LIỆU TRONG FILE DẠNG TEXT CÓ CẤU TRÚC NHƯ SAU: DÒNG ĐẦU CHỨA HAI SỐ M,N. TỪ DÒNG THỨ HAI ĐẾN DÒNG THỨ M+1, CHỨA CÁC HÀNG CỦA MA TRẬN A = (A_{IJ}). KẾT QUẢ ĐƯA RA FILE DẠNG TEXT CÓ CẤU TRÚC**

NHƯ SAU: DÒNG ĐẦU CHỨA SỐ PHÒNG, DÒNH HAI CHỨA DIỆN TÍACH PHÒNG LỚN NHẤT VÀ DONG BA CHỨA HÀNG, CỘT, HƯỚNG CỦA BỨC TƯỜNG CẦN PHÁ.

CHẲNG HẠN DỮ LIỆU VÀO LÀ

4 6

11 6 11 6 3 10 6

7 9 6 13 5 15 5

1 10 12 7 13 7 5

13 11 10 8 10 12 13

DỮ LIỆU RA SẼ LÀ:

5

9

4 1 DONG

BÀI TẬP 8. **MỘT SÂN CHƠI HÌNH CHỮ NHẬT GỒM M*N Ô ĐƠN VỊ. TRÊN MỖI Ô (I,J) CÓ ĐÓNG CÁC TRỤ BÊ TÔNG CHIỀU CAO A_{IJ}. GIẢ THIẾT NƯỚC KHÔNG THẤM QUA ĐƯỢC CÁC CẠNH GIỮA HAI TRỤ BÊ TÔNG KỀ NHAU. SAU MỘT TRẬN MƯA ĐỦ LỚN, HÃY TÍNH NƯỚC ĐỌNG LẠI TRÊN SÂN.**

HƯỚNG DẪN

CHIA NƯỚC THÀNH TỪNG TẦNG CÓ CHIỀU CAO BẰNG 1. TÍNH THỂ TÍCH NƯỚC ĐỌNG TRÊN MỖI TẦNG THEO THUẬT TOÁN LOANG TÌM THÀNH PHẦN LIÊN THÔNG.

BÀI TẬP 9. **TÌM 2 CHỮ SỐ PHÂN BIỆT A VÀ B SAO CHO THOÃ MÃN HAI ĐIỀU KIỆN SAU:**

A. A2B CHIA HẾT CHO 3

B. $\overline{A2B}$ - $\overline{AB}$=110

BÀI TẬP 10. **GẢII BÀI TOÁN ĐOÁN CHỮ SAU**

DONALD CROSS

+

GERALD

+

ROADS

ROBERT DANGER

BÀI TẬP 11. **CHO SỐ CÓ HAI CHỮ SỐ. NẾU VIẾT THÊM HAI CHỮ SỐ VỀ BÊN PHẢI SỐ ĐÓ THÌ ĐƯỢC SỐ MỚI LỚN HƠN SỐ ĐÃ HO LÀ 1986 ĐƠN VỊ. HÃY TÌM SỐ ĐÃ CHO VÀ HAI CHỮ SỐ VIẾT THÊM ĐÓ.**

BÀI TẬP 12. **GIẢI BÀI TOÁN ĐOÁN CHỮ SAU:**

T

+ TH

THA

THAN

_____

4321

CHƯƠNG TRÌNH THAM KHẢO

```
PROGRAM CANO_DI_TUAN; { BÀI TẬP 3}
USES CRT;
TYPE
    DAO =    RECORD
                X,Y: INTEGER;
             END;
VAR
    N,D1,D2,SO: BYTE;
    M: WORD;
    A: ARRAY[BYTE] OF DAO;
    B: ARRAY[BYTE] OF BOOLEAN;
    TR: ARRAY[BYTE] OF BYTE;

PROCEDURE NHAP;
VAR
    F: TEXT;
    S: STRING[20];
```

```
        I: BYTE;
BEGIN
    CLRSCR;
    WRITE('TEN FILE DU LIEU:');
    READLN(S);
    ASSIGN(F,S);
    RESET(F);
    READLN(F,N);
    READLN(F,M);
    FOR I:=1 TO N DO
       WITH A[I] DO READLN(F,X,Y);
    CLOSE(F);
END;

PROCEDURE INDULIEU;
VAR
        I: BYTE;
BEGIN
    WRITELN('SO DAO:',N);
    WRITELN('GIOI HAN KHOANG CACH:',M);
    FOR I:=1 TO N DO
       WITH A[I] DO WRITELN('TOA DO DAO ',I,' : (',X,',',Y,')');
```

```
END;

PROCEDURE KHOITAO;
VAR
    I: BYTE;
BEGIN
  FOR I:=1 TO N DO
    B[I]:= TRUE;
END;

FUNCTION KC(I,J: BYTE):REAL;
BEGIN
  KC:= SQRT(SQR(A[I].X-A[J].X)+SQR(A[I].Y-A[J].Y));
END;

PROCEDURE BFS(I: BYTE);
VAR
    J,K,D,C: BYTE;
    Q: ARRAY[BYTE] OF BYTE;
BEGIN
  D:=1;
  C:=1;
```

```
    Q[1]:=I;
    B[I]:= FALSE;
    WHILE D<=C DO
      BEGIN
         J:= Q[D];
         D:=D+1;
         FOR K:=1 TO N DO
            IF B[K] AND (KC(K,J) <= M) THEN
              BEGIN
                 C:=C+1;
                 Q[C]:=K;
                 B[K]:= FALSE;
                 TR[K]:=J;
              END;
      END;
END;

PROCEDURE INKETQUA;
VAR
       I:BYTE;
BEGIN
   WRITE('DUONG DI GHE DAO IT NHAT NHU SAU: ');
```

```
    WRITE(D1);
    I:=D1;
    SO:=0;
    WHILE I<>D2 DO
      BEGIN
         WRITE('-->',TR[I]);
         SO:=SO+1;
         I:=TR[I];
      END;
    WRITELN;
    WRITELN('DUONG DI TREN GHE QUA ',SO-1,' DAO');
END;

PROCEDURE TIMDUONGDI;
BEGIN
   WRITE('DAO XUAT PHAT: ');
   READLN(D1);
   WRITE('DAO KET THUC: ');
   READLN(D2);
   BFS(D2);
   IF B[D2] THEN WRITE('KHONG CO DUONG DI TU ',D1,' DEN ',D2)
   ELSE INKETQUA;
```

```
  READLN;
END;

BEGIN
    NHAP;
    KHOITAO;
    INDULIEU;
    TIMDUONGDI;
END.

PROGRAM LAUDAI; { BÀI TẬP 7}
USES CRT;
TYPE
        SIZE = 0..100;
VAR
        M,N,SO,P,HANG,COT: SIZE;
        A:ARRAY[SIZE,SIZE] OF 0..15;
        PH: ARRAY[SIZE,SIZE] OF WORD;
        S: ARRAY[SIZE] OF WORD;
        DT: WORD;
        HUONG: STRING[4];
```

```
PROCEDURE NHAP;
VAR
        F: TEXT;
        I,J: SIZE;
BEGIN
  CLRSCR;
  ASSIGN(F,'INPUT.PAS');
  RESET(F);
  READ(F,M,N);
  FOR I:=1 TO M DO
     FOR J:=1 TO N DO READ(F,A[I,J]);
  CLOSE(F);
END;

PROCEDURE KHOITAO;
VAR
        I,J: SIZE;
BEGIN
    FOR I:=1 TO M DO
       FOR J:=1 TO N DO
              PH[I,J]:=0;
    SO:=0;
```

```
END;



PROCEDURE BFS(I,J: SIZE);
VAR
        QH,QC: ARRAY[SIZE] OF SIZE;
        D,C,K,L: SIZE;
BEGIN
    PH[I,J]:= SO;
    D:=1;
    C:=1;
    QH[1]:=I;
    QC[1]:=J;
    WHILE D<=C DO
        BEGIN
            K:= QH[D];
            L:= QC[D];
            D:=D+1;
            IF A[K,L]>=8 THEN
               S[K,L]:=A[K,L]-8
            ELSE
               IF (K<M) AND (PH[K+1,L]=0) THEN
```

```
      BEGIN
         C:=C+1;
         QH[C]:=K+1;
         QC[C]:=1;
         PH[K+1,L]:=SO;
      END;
IF A[K,L]>=4 THEN
   A[K,L]:=A[K,L]-4
ELSE
   IF (L<N) AND (PH[K,L+1] = 0) THEN
     BEGIN
       C:=C+1;
       QH[C]:=K;
       QC[C]:=L+1;
       PH[K,L+1]:=SO;
     END;
IF A[K,L]>=2 THEN
  A[K,L]:= A[K,L]-2
ELSE
  IF (K>1) AND (PH[K-1,L] = 0) THEN
     BEGIN
        C:=C+1;
```

```
                QH[C]:=K-1;
                QC[C]:=L;
                PH[K-1,L]:=SO;
            END;
        IF A[K,L] >=1 THEN
          A[K,L]:= A[K,L]-1
        ELSE
          IF (L>1) AND (PH[K,L-1]=0) THEN
            BEGIN
              C:=C+1;
              QH[C]:=K;
              QC[C]:=L-1;
              PH[K,L-1]:=SO;
            END;
      END;
END;

PROCEDURE DEMPHONG;
VAR
    I,J: SIZE;
BEGIN
  FOR I:=1 TO M DO
```

```
      FOR J:=1 TO N DO
        IF PH[I,J] = 0 THEN
          BEGIN
            SO:= SO+1;
            BFS(I,J);
          END;
END;

PROCEDURE SMAX;
VAR
        I: WORD;
        J,K: SIZE;
BEGIN
    DT:=0;
    FOR I:=1 TO SO DO
      BEGIN
        S[I]:=0;
        FOR J:=1 TO M DO
          FOR K:=1 TO N DO
            IF PH[J,K]=I THEN S[I]:= S[I]+1;
              IF  S[I] > DT THEN DT:= S[I];
      END;
```

```
END;
```

PROCEDURE PHATUONG;

```
{ CHỈ CẦN PHÁ PHÍA ĐÔNG HOẶC PHÍA NAM, PHÍA TÂY CỦA Ô
(I,J) TƯƠNG ỨNG LÀ PHÍA ĐÔNG CỦA Ô (I,J-1), TƯƠNH TỰ, PHÍA
BẮC CỦA Ô (I,J) TƯƠNG ỨNG PHÍA NAM CỦA Ô (I-1,J)}
VAR
      I,J: SIZE;
      MAX,TG: WORD;
BEGIN
 MAX:=0;
 FOR I:=1 TO M DO
   FOR J:=1 TO N DO
    BEGIN
     IF I< M THEN
      IF PH[I,J] <> PH[I+1,J] THEN
        BEGIN
          TG:= S[PH[I,J]] + S[PH[I+1,J]];
          IF TG >= MAX THEN
            BEGIN
              HANG :=I;
              COT:=J;
              HUONG:= 'NAM';
```

```
            MAX:= TG;
          END;
        END;
      IF J<N THEN
       IF PH[I,J]<> PH[I,J+1] THEN
        BEGIN
         TG:=  S[PH[I,J]] + S[PH[I,J+1]];
         IF TG >= MAX THEN
          BEGIN
           HANG:=I;
           COT:=J;
           HUONG:= 'DONG';
           MAX:= TG;
          END;
        END;
   END;
END;

PROCEDURE INKQ;
VAR
      I,J: SIZE;
      F: TEXT;
```

```
BEGIN
  ASSIGN(F,'OUT.PAS');
  REWRITE(F);
  WRITELN(F,SO);
  WRITELN(F,DT);
  WRITELN(F,HANG,' ',COT,' ',HUONG);
  CLOSE(F);
END;
BEGIN
    NHAP;
    KHOITAO;
    DEMPHONG;
    SMAX;
    PHATUONG;
   INKQ;
END.
```

Chương 4

BIỂU DIỄN BÀI TOÁN BẰNG LOGIC VÀ CÁC PHƯƠNG PHÁP CHỨNG MINH

Như ta đã biết, không thể có phương pháp giải quyết vấn đề tổng quát cho mọi bài toán. Có thể phương pháp này phù hợp cho bài toán này, nhưng lại không phù hợp cho lớp bài toán khác. Điều này có nghĩa là khi nói tới một bài toán, ta phải chú ý đến ***phương pháp biểu diễn nó*** cùng với các ***phương pháp tìm kiếm trong không gian bài toán nhận được***.

1. Biểu diễn bài toán nhờ không gian trạng thái (có các chiến lược tìm kiếm trên đồ thị biểu diễn vấn đề)
2. Quy về các bài toán con
3. Biểu diễn vấn đề nhờ logic hình thức (có các phương pháp suy diễn logic)

....

và trong phần này sẽ trình bày phương pháp biểu diễn vấn đề nhờ logic hình thức và các phương pháp giải quyết vấn đề trên cách biểu diễn này.

Logic hình thức thường dùng để thu gọn quá trình tìm kiếm lời giải. Trước khi giải quyết vấn đề, nhờ phân tích logic, có thể chứng tỏ rằng một bài toán nào đó có thể giải được hay không?.

Ngoài ra, các kết luận logic rất cần ngay cả trong cách tiếp cận dựa trên không gian trạng thái và quy bài toán về bài toán con. Chẳng hạn, trong các phương pháp dựa trên không gian trạng thái, các kết luận logic dùng để kiểm tra một trạng thái nào đó có phải là trạng thái đích hay không?,....

Ngoài ra, logic hình thức có thể được sử dụng để giải quyết những bài toán chứng minh logic, chẳng hạn như chứng minh một khẳng định nào đó là đúng khi biết những tiền đề ban đầu và các luật suy diễn. Đây là một dạng quen thuộc nhất và được các chuyên gia TTNT quan tâm ngay từ đầu.

VÍ DỤ

Ta có thể dùng các biểu thức logic để mô tả mối quan hệ của các thành phần trong 1 tam giác như sau:

1) $a \wedge b \wedge c \Rightarrow p$

2) $b \wedge p \wedge c \Rightarrow a$

3) $a \wedge p \wedge c \Rightarrow b$

4) $a \wedge b \wedge p \Rightarrow c$

5) $S \wedge c \Rightarrow hc$

6) $a \wedge b \wedge C \Rightarrow c$

7) $a \wedge b \wedge C \Rightarrow S$

8) $a \wedge b \wedge c \wedge p \Rightarrow S$

9) $S \wedge hc \Rightarrow c$

(***Trong đó:*** *a, b, c là ký hiệu các cạnh, A, B, C là ký hiệu các góc tương ứng, p là ký hiệu nữa chu vi, và hc là đường cao xuất phát từ đỉnh C của tam giác*)

Giả sử ta biết các cạnh a, b và một góc C. Ta có thể có kết luận về đường cao hc không?

1. BI ẾU DI ỄN VẤN ĐỀ NHỜ LOGIC HÌNH THỨC

1.1. Logic mệnh đề

Đây là kiểu biểu diễn tri thức đơn giản nhất và gần gũi nhất đối với chúng ta.

a) ***Mệnh đề*** là một khẳng định, một phát biểu mà giá trị của nó chỉ có thể hoặc là đúng hoặc là sai.

Ví dụ

phát biểu "1+1=2" (có giá trị đúng)

phát biểu "Trời mưa"

(Giá trị của mệnh đề không chỉ phụ thuộc vào bản thân mệnh đề đó. Có những mệnh đề mà giá trị của nó luôn đúng hoặc sai bất chấp thời gian nhưng cũng có những mệnh đề mà giá trị của nó lại phụ thuộc vào thời gian, không gian và

nhiều yếu tố khác quan khác. Chẳng hạn như mệnh đề : "Con người không thể nhảy cao hơn 5m với chân trần" là đúng khi ở trái đất , còn ở những hành tinh có lực hấp dẫn yếu thì có thể sai.)

b) Biểu thức logic

- Ta ký hiệu mệnh đề bằng những chữ cái la tinh như **a, b, c, ...** và các ký hiệu này được *gọi là biến mệnh đề*

- *Biểu thức logic* được định nghĩa đệ quy như sau:

- Các hằng logic (True, False) và các biến mệnh đề là các biểu thức logic
- Các biểu thức logic kết hợp với các toán tử logic (phép tuyển ($\vee$), phép hội ($\wedge$), phủ định ($\neg$, ~, ¯), phép kéo theo ($\Rightarrow$, $\rightarrow$), phép tương đương ($\Leftrightarrow$, $\equiv$)) là các biểu thức logic.

Tức là nếu E và F là các biểu thức logic thì $E \wedge F$, $E \vee F$, $E \rightarrow F$, $E \equiv F$ cũng là các biểu thức logic

Thứ tự ưu tiên của các phép toán logic: $\neg, \wedge, \vee, \rightarrow, \equiv$

VÍ DỤ MỘT SỐ BIỂU THỨC LOGIC:

1) True
2) $\neg p$
3) $p \wedge (p \vee r)$

.....

- *Biểu thức logic dạng chuẩn:* là biểu thức được xây dựng từ các biến mệnh đề và các phép toán $\neg, \wedge, \vee$.

VÍ DỤ $P \wedge (\neg P \vee R)$

(Chúng ta đã từng sử dụng logic mệnh đề trong chương trình rất nhiều lần (như trong cấu trúc lệnh IF ... THEN ... ELSE) để biểu diễn các tri thức "cứng" trong máy tính !)

c) Bảng chân trị (bảng chân lý) Dùng để dánh giá giá trị của biểu thức logic.

p	q	¬p	p ∨ q	p ∧ q	¬p ∨ q	p → q	p ≡ q
T	T	F	T	T	T	T	T
T	F	F	T	F	F	F	F
F	T	T	T	F	T	T	F
F	F	T	F	F	T	T	T

Nhận xét

- Mọi biểu thức logic đều có thể chuyển về các biểu thức logic dạng chuẩn nhờ vào:

$$p \to q \equiv \neg p \vee q$$

- Nếu có n biến mệnh đề trong biểu thức logic thì bảng chân trị sẽ có 2^n trường hợp khác nhau đối với các biến mệnh đề.

d) Đồng nhất đúng

Một đồng nhất đúng là một biểu thức logic luôn luôn có giá trị True với bất kỳ giá trị nào của các biến mệnh đề trong biểu thức logic đó.

VÍ DỤ (CÓ THỂ KIỂM TRA BẰNG CÁCH DÙNG BẢNG CHÂN TRỊ)

1) $p \vee \neg p$

2) $0 \to p$

3) $(p \vee q) \wedge (\neg p \vee r) \to q \vee r$

Ta thấy rằng biểu thức có dạng VT→VP luôn có giá trị True (T) với mọi giá trị của a, b; chỉ có một trường hợp để a →b có giá trị False (F) là a: True và b: False. Như vậy, để chứng minh biểu thức 3) là một đồng nhất đúng, ta chỉ cần chứng minh nếu b: F thì a: F, không có trường hợp a: T và b: F.

Thật vậy, giả sử VP: F nghĩa là q: F và r: F. Xét 2 trường hợp của p:

- Nếu p: T thì VT: F
- Nếu p: F thì VT: F

Do đó biểu thức 3) là một đồng nhất đúng

Bài tập. *Biểu thức nào trong số các biểu thức sau đây là đồng nhất đúng?*

1) $p \wedge q \wedge r \rightarrow p \vee q$

2) $(p \rightarrow q) \rightarrow p$

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

1.2. Một số luật đại số

Sau đây là một số đồng nhất đúng thường gặp

a) Luật phản xạ (cho phép tương đương): $p \equiv p$

b) Luật giao hoán

- phép tương đương: $p \equiv p$
- phép hội: $p \wedge q \equiv q \wedge p$
- phép tuyển: $p \vee q \equiv q \vee p$

c) Luật bắc cầu:

- phép kéo theo: $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow (p \rightarrow r)$
- phép tương đương: $(p \equiv q) \wedge (q \equiv r) \rightarrow (p \equiv r)$

d) Luật kết hợp:

- phép hội: $p \wedge (q \wedge r) \equiv (p \wedge q) \wedge r$
- phép tuyển: $p \vee (q \vee r) \equiv (p \vee q) \vee r$

e) Luật phân phối:

- phép $\wedge$ trên phép $\vee$: $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
- phép $\vee$ trên phép $\wedge$: $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

f) Phần tử trung hoà:

- 0 (False) là phần tử trung hoà cho phép $\vee$: $p \vee 0 \equiv p$
- 1 (true) là phần tử trung hoà cho phép $\wedge$: $p \wedge 1 \equiv p$

g) Triệt tử

- 0 (False) là triệt tử cho phép $\wedge$: $p \wedge 0 \equiv 0$
- 1 (true) là triệt tử cho phép $\vee$: $p \vee 1 \equiv 1$

h) Tính luỹ đẳng

- của phép $\wedge$: $p \wedge p \equiv p$
- của phép $\vee$: $p \vee p \equiv p$

i) Luật Demorgan

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

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

j) Một số luật khác cho phép kéo theo

- $(p \rightarrow q) \wedge (q \rightarrow p) \equiv (p \equiv q)$
- $(p \equiv q) \rightarrow (p \rightarrow q)$
- $p \rightarrow q \equiv \neg p \vee q$

k) $\neg (\neg p) \equiv p$

1.3. Logic vị từ

Biểu diễn tri thức bằng mệnh đề gặp phải một trở ngại cơ bản là ta không thể can thiệp vào cấu trúc của một mệnh đề. Hay nói một cách khác là mệnh đề *không có cấu trúc* . Điều này làm hạn chế rất nhiều thao tác suy luận .

Do đó, người ta đã đưa vào ***khái niệm vị từ và lượng từ*** (**$\forall$:với mọi, $\exists$: tồn tại**) để tăng cường tính cấu trúc của một mệnh đề.

Trong logic vị từ, một mệnh đề *được cấu tạo bởi* hai thành phần là các ***đối tượng tri thức*** và ***mối liên hệ giữa chúng*** *(gọi là vị từ)*. Các mệnh đề sẽ được biểu diễn dưới dạng:

Vị từ (<đối tượng 1>, <đối tượng 2>, …, <đối tượng n>)

VÍ DỤ

Để biểu diễn vị của các trái cây, các mệnh đề sẽ được viết lại thành :

Cam có vị Ngọt $\Rightarrow$Vị (Cam, Ngọt)

Cam có màu Xanh $\Rightarrow$ Màu (Cam, Xanh)

...

Kiểu biểu diễn này có hình thức tương tự như hàm trong các ngôn ngữ lập trình, các đối tượng tri thức chính là các tham số của hàm, giá trị mệnh đề chính là kết quả của hàm (thuộc kiểu BOOLEAN).

Với vị từ, ta có thể biểu diễn các tri thức dưới dạng các mệnh đề tổng quát, là những mệnh đề mà giá trị của nó được xác định thông qua các đối tượng tri thức cấu tạo nên nó.

Ví dụ

1) Chẳng hạn tri thức : *"A là bố của B nếu B là anh hoặc em của một người con của A"* có thể được biểu diễn dưới dạng vị từ như sau :

Bố (A, B) = Tồn tại Z sao cho : Bố (A, Z) và (Anh(Z, B) hoặc Anh(B,Z))

Trong trường hợp này, mệnh đề Bố(A,B) là một mệnh đề tổng quát

Như vậy nếu ta có các mệnh đề cơ sở là :

a) Bố ("An", "Bình") có giá trị đúng (Anh là bố của Bình)

b) Anh("Tú", "Bình") có giá trị đúng (Tú là anh của Bình)

thì mệnh đề **c)** Bố ("An", "Tú") sẽ có giá trị là đúng. (An là bố của Tú).

Rõ ràng là nếu chỉ sử dụng logic mệnh đề thông thường thì ta sẽ không thể tìm được một mối liên hệ nào giữa c và a,b bằng các phép nối mệnh đề $\wedge$, $\vee$, $\neg$. Từ đó, ta cũng không thể tính ra được giá trị của mệnh đề **c.** Sở dĩ như vậy vì ta không thể thể hiện tường minh tri thức *"(A là bố của B) nếu có Z sao cho (A là bố của Z) và (Z anh hoặc em C)"* dưới dạng các mệnh đề thông thường. Chính đặc trưng của vị từ đã cho phép chúng ta thể hiện được các tri thức dạng tổng quát như trên.

2) Câu cách ngôn *"Không có vật gì là lớn nhất và không có vật gì là bé nhất!"* có thể được biểu diễn dưới dạng vị từ như sau :

LớnHơn(x,y) = x>y

NhỏHơn(x,y) = x<y□

$\forall$x,□ $\exists$y : LớnHơn(y,x) và $\forall$x, $\exists$y : NhỏHơn(y,x)

3) Câu châm ngôn *"Gần mực thì đen, gần đèn thì sáng"* được hiểu là "chơi với bạn xấu nào thì ta cũng sẽ thành người xấu" có thể được biểu diễn bằng vị từ như sau :

NgườiXấu (x) = $\forall$y : Bạn(x,y) và NgườiXấu(y)

Sử dụng vị từ làm toán hạng nguyên tử thay vì các biến mệnh đề đã đưa ra một ngôn ngữ mạnh mẽ hơn so với các biểu thức chỉ chứa mệnh đề. Thực sự, logic vị từ đủ khả năng diễn tả để tạo cơ sở cho một số ngôn ngữ lập trình rất có ích như Prolog (Programing Logic) và ngôn ngữ SQL. Logic vị từ cũng được sử dụng trong các hệ thống suy luận hoặc các hệ chuyên gia chẳng hạn các chương trình chẩn đoán tự động y khoa, các chương trình chứng minh định lý tự động

1.3.1. Cú pháp và ngữ nghĩa của logic vị từ

a. Cú pháp

- ***Các ký hiệu***

- **Hằng**: được biểu diễn bằng chuỗi ký tự bắt đầu bằng chữ cái thường hoặc các chữ số hoặc chuỗi ký tự đặt trong bao nháy. Ví dụ: a,b, c, "An", "Ba",...
- **Biến**: tên biến luôn bắt đầu bằng chữ cái viết hoa. Ví dụ: X, Y, Z, U, V,...
- **Vị từ**: được biểu diễn bằng chuỗi ký tự bắt đầu bằng chữ cái thường. Ví dụ: p, q, r, s, like,...

Mỗi vị từ là vị từ của n biến (n≥0). Các ký hiệu vị từ không có biến là các ký hiệu mệnh đề

Ví dụ: like(X,Y) là vị từ của hai biến
u(X) là vị từ một biến
r là vị từ không biến

- **Hàm:** f, g, cos, sin, mother,...

Mỗi hàm là hàm của n biến (n≥1). Ví dụ: cos, sin là hàm một biến

- **Lượng từ:** $\forall$(với mọi), $\exists$ (tồn tại).

Ví dụ: $\forall X$, p(X) nghĩa là với mọi giá trị của biến X đều làm cho biểu thức p đúng.

$\exists X$, p(X) nghĩa là có ít nhất một giá trị của biến X để làm cho biểu thức p đúng.

- Các ký hiệu kết nối logic: $\wedge$ (hội), $\vee$ (tuyển), $\neg$(phủ định), $\Rightarrow$ (kéo theo), $\Leftrightarrow$ (kéo theo nhau).

- Các ký hiệu ngăn cách: dấu phẩy, dấu mở ngoặc và dấu đóng ngoặc.

- ***Các hạng thức***

Các hạng thức (term) là các biểu thức mô tả các đối tượng. Các hạng thức được xác định đệ quy như sau:

- Các ký hiệu hằng và các ký hiệu biến là hạng thức

- Nếu t_1, t_2, t_3,...,t_n là n hạng thức và f là một ký hiệu hàm n biến thì f(t_1, t_2, t_3,...,t_n) là hạng thức. Một hạng thức không chứa biến được gọi là một hạng thức cụ thể (ground term).

Ví dụ: An là một ký hiệu hằng, mother là ký hiệu hàm một biến thì mother("An") là một hạng thức cụ thể

- ***Các công thức phân tử***

Chúng ta sẽ biểu diễn các tính chất của đối tượng, hoặc các quan hệ giữa các đối tượng bởi các công thức phân tử (câu đơn)

Các công thức phân tử được xác định đệ quy như sau

- Các ký hiệu vị từ không biến (các ký hiệu mệnh đề) là công thức phân tử

- Nếu t_1, t_2, t_3,...,t_n là n hạng thức và p là vị từ của n biến thì p(t_1, t_2, t_3,...,t_n) là công thức phân tử.

Ví dụ: Hoa là một ký hiệu hằng, love là một vị từ hai biến, husband là hàm của một biến thế thì love("Hoa", husband("Hoa")) là một công thức phân tử.

- ***Các công thức***

Từ công thức phân tử, sử dụng các kết nối logic và các lượng từ, ta xây dựng nên các công thức (các câu)

Các công thức được xác định đệ quy như sau:

- Các công thức phân tử là công thức

- Nếu G và H là các công thức thì các biểu thức ($G \wedge H$), ($G \vee H$), ($\neg G$), ($G \Rightarrow H$), ($G \Leftrightarrow H$) là công thức

- Nếu G là một công thức và X là biến thì các biểu thức $\forall x$ (G), $\exists x$ (G) là công thức

Các công thức không phải là công thức phân tử sẽ được gọi là các câu phức hợp. Các công thức không chứa biến sẽ được gọi là công thức cụ thể. Khi viết các công thức ta sẽ bỏ đi các dấu ngoặc không cần thiết, chẳng hạn các dấu ngoặc ngoài cùng.

Lượng từ phổ dụng (universal quantfier) cho phép mô tả tính chất của cả một lớp các đối tượng chứ không phải của một đối tượng mà không cần phi liệt kê ra tất cả các đối tượng trong lớp. Chẳng hạn sử dụng vị từ elephant(X) (đối tượng X là con voi) và vị từ color(X, "Gray") (đối tượng X có màu xám) thì câu "tất cả các con voi đều có màu xám" có thể biểu diễn bởi công thức: $\forall X$ (elephant(X) $\Rightarrow$ color(X, "Gray"))

Lượng từ tồn tại (existantial quantifier) cho phép ta tạo ra các câu nói đến một đối tượng nào đó trong một lớp đối tượng mà nó có một tính chất hoặc thõa mãn một quan hệ nào đó. Chẳng hạn bằng cách sử dụng các câu đơn student(X) (X là sinh viên) và inside(X, "P301") (X ở trong phòng 301), ta có thể biểu diễn câu "Có một sinh viên ở phòng 301" bởi biểu thức: $\exists x$ (student(X) $\wedge$ inside(X, "P301"))

Một công thức là công thức phân tử hoặc phủ định công thức phân tử được gọi là literal. Chẳng hạn, play(X, "Football"), $\neg$like("Lan", "Rose") là các

literal. Một công thức là tuyển của các literal sẽ được gọi là câu tuyển. Chẳng hạn, male(X)$\vee$ $\neg$ like(X,"Football") là câu tuyển.

Trong công thức $\forall$X (G), hoặc $\exists$X (G) trong đó G là một công thức nào đó thì mỗi xuất hiện của biến X trong công thức G được gọi là xuất hiện buộc. Một công thức mà tất cả các biến đều là xuất hiện buộc thì được gọi là công thức đóng.

Ví dụ: Công thức $\forall$X, p(X, f(a,X)) $\wedge$ $\exists$Y, q(Y) là công thức đóng, còn công thức $\forall$X, p(X, f(Y,X)) không phải là công thức đóng vì sự xuất hiện của biến Y trong công thức này không chịu ràng buộc bởi một lượng tử nào cả (sự xuất hiện của Y gọi là sự xuất hiện tự do)

b. Ngữ nghĩa

Cũng như trong logic mệnh đề, nói đến ngữ nghĩa là chúng ta nói đến ý nghĩa của các công thức trong một thế giới hiện thực nào đó mà chúng ta sẽ gọi là một minh họa. Để xác định một minh họa, trước hết ta cần xác định một miền đối tượng (nó bao gồm tất cả các đối tượng trong thế giới thực mà ta quan tâm).

Trong một minh họa, các ký hiệu hằng sẽ được gắn với các đối tượng cụ thể trong miền đối tượng, các ký hiệu hàm sẽ được gắn với một hàm cụ thể nào đó. Khi đó mỗi hạng thức cụ thể sẽ chỉ định một đối tượng cụ thể trong miền đối tượng. Chẳng hạn nếu An là một ký hiệu hằng, father là một ký hiệu hàm, nếu trong minh họa An ứng với một người cụ thể nào đó, còn father(X) gắn với hàm ứng với mỗi X là cha của nó, thì hạng thức father("An") sẽ chỉ người cha của An.

- ***Ngữ nghĩa của các câu đơn***

Trong một minh họa, các ký hiệu vị từ sẽ được gắn với một thuộc tính hoặc một quan hệ cụ thể nào đó. Khi đó, mỗi công thức phân tử (không chứa biến) sẽ chỉ định một sự kiện cụ thể. Đương nhiên sự kiện này có thể là đúng (true) hoặc sai (false). Chẳng hạn, nếu minh họa, ký hiệu hằng Lan ứng với một cô gái cụ

thể nào đó còn student(X) ứng với thuộc tính X là sinh viên thì câu student ("Lan") có giá trị chân lý là true hoặc false tùy thuộc trong thực tế Lan có phi sinh viên hay không.

- ***Ngữ nghĩa của các câu phức hợp***

Khi đã xác định được ngữ nghĩa của các câu đơn, ta có thể xác định được ngữ nghĩa của các câu phức hợp (được tạo thành từ các câu đơn bằng các liên kết các câu đơn bởi các kết nối logic) như trong logic mệnh đề

Ví dụ: Câu student("Lan") $\wedge$ student("An") nhận giá trị true nếu cả hai câu student(Lan) và student(An) đều có giá trị true, tức là cả Lan và An đều là sinh viên.

- ***Ngữ nghĩa của các câu chứa các lượng từ***

Ngữ nghĩa của các câu $\forall$X(G), trong đó G là một công thức nào đó được xác định là ngữ nghĩa của công thức là hội của tất cả các công thức nhận được từ công thức G bằng cách thay X bởi một đối tượng trong miền đối tượng. Chẳng hạn, nếu miền đối tượng gồm ba người {Lan, An, Hoa} thì ngữ nghĩa của câu $\forall$X, student(X) được xác định là ngữ nghĩa của câu student("Lan") $\wedge$ student("An") $\wedge$ student("Hoa"). Câu này đúng khi và chỉ khi cả ba câu thành phần đều đúng, tức là cả Lan, Hoa, An đều là sinh viên. Như vậy, công thức $\forall$X (G) là đúng nếu và chỉ nếu mọi công thức nhận được từ G bằng cách thay X bởi một đối tượng bất kỳ trong miền đối tượng đều đúng, tức là G đúng cho tất cả đối tượng X trong miền đối tượng

Ngữ nghĩa của công thức $\exists$X (G) được xác định như là ngữ nghĩa của công thức là tuyển của tất c các công thức nhận được từ công thức G bằng cách thay X bởi một đối tượng trong miền đối tượng. Chẳng hạn, nếu ngữ nghĩa của câu younger(X, 20) là "X trẻ hn 20 tuổi" và miền đối tượng gồm ba người { Lan, Hoa, An} thì ngữ nghĩa của câu $\exists$X younger(X, 20) là ngữ nghĩa của câu younger("Lan", 20) $\vee$ younger("Hoa", 20) $\vee$ younger("An", 20). Câu này nhận

giá trị True nếu và chỉ nếu ít nhất một trong ba người Lan, Hoa, An trẻ hơn 20 tuổi. Như vậy, công thức ∃X(G) là đúng nếu và chỉ nếu một trong các công thức nhận được từ G bằng cách thay X bởi một đối tượng trong miền đối tượng là đúng.

Các công thức tương đương

Cũng như trong logic mệnh đề, ta nói hai công thức G và H tương đương (G≡H) nếu chúng cùng đúng hoặc cùng sai trong mọi minh hoạ. Ngoài các tương đương đã biết trong logic mệnh đề, trong logic vị từ còn có các tương đương khác liên quan tới các lượng từ. Giả sử G là một công thức, cách viết G(X) nói rằng công thức G có chứa các xuất hiện của biến X. Khi đó công thức G(Y) là công thức nhận được từ G(X) bằng cách thay tất cả các xuất hiện của X bởi Y. Ta nói G(Y) là công thức nhận được từ G(X) bằng cách đặt tên lại (biến X đổi tên lại là Y).

Chúng ta có các tương đương sau đây:

1. $\forall X\,(G(X)) \equiv \forall Y\,(G(Y))$

   $\exists X\,(G(X)) \equiv \exists Y\,(G(Y))$

   Đặt tên lại biến đi sau lượng từ phổ dụng (tồn tại), ta nhận được công thức tương đương.

2. $\neg\,(\forall X\,(G(X))) \equiv \exists X\,(\neg G(X))$

   $\neg\,\exists X\,(G(X)) \equiv \forall X\,(\neg G(X))$

3. $\forall X\,(G(X) \wedge H(X)) \equiv \forall X\,(G(X)) \wedge \forall X\,(H(X))$

   $\exists x\,(G(X) \vee H(X)) \equiv \exists X\,(G(X)) \vee \exists X\,(H(X))$

Bài tập

1) Giả sử ta thiết lập vị từ có 3 đối số csg(C,S,G) biểu diễn câu: “Sinh viên S nhận điểm G trong học phần C”. Vậy với các giá trị cụ thể của vị từ chẳng hạn như csg(“Anhvan”, “An”, 8) thì có thể được diễn giải như thế nào?

2) Giả sử ta có vị từ loves(X,Y) được diễn giải là: "X yêu Y", như vậy các câu sau (biểu diễn trong logic vị từ) được hiểu như thế nào?

$\forall X (\exists Y (loves(X,Y))$

$\exists Y (\forall X (loves(X,Y))$

3) Giả sử ta có các vị từ:

dog(X) ("X là chó"), cat(Y) ("Y là mèo"), animal(Z) ("Z là động vật"). Hãy biểu diễn câu sau trong logic vị từ: "chó, mèo đều là động vật".

1.3.2. Chuẩn hoá các công thức

Từ các câu phân tử, bằng cách sử dụng các kết nối logic và các lượng từ, ta có thể tạo ra các câu phức hợp có cấu trúc rất phức tạp. Để dễ dàng cho việc lưu trữ các câu trong bộ nhớ và thuận lợi cho việc xây dựng các thủ tục suy diễn, chúng ta cần chuẩn hoá các câu bằng cách đưa chúng về dạng chuẩn tắc hội (hội của các câu tuyển).

Trong mục này, chúng ta sẽ trình bày thủ tục chuyển một câu phức hợp thành một câu ở dạng chuẩn tắc hội tương đương.

Thủ tục chuẩn hoá các công thức bao gồm các bước sau:

a. Loại bỏ các kéo theo

Để loại bỏ các kéo theo, ta chỉ cần thay công thức $P \Rightarrow Q$ bởi công thức tương đương $\neg P \vee Q$, thay $P \Leftrightarrow Q$ bởi $(\neg P \vee Q) \wedge (P \vee \neg Q)$

b. Chuyển các phủ định tới các phân tử

Điều này được thực hiện bằng cách thay công thức ở vế trái bởi công thức ở vế phải trong các tương đương sau đây:

$$\neg (\neg P) \equiv P$$

$$\neg (P \wedge Q) \equiv \neg P \vee \neg Q$$

$$\neg (P \vee Q) \equiv \neg P \wedge \neg Q$$

$$\neg (\forall X (P)) \equiv \exists X (\neg P)$$

$$\neg (\exists X (P)) \equiv \forall X (\neg P)$$

c. Loại bỏ các lượng từ tồn tại

Giả sử p(X,Y) là các vị từ có nghĩa rằng "Y lớn hơn X" trong miền các số. Khi đó công thức $\forall x\ (\exists y\ (P(x,y)))$ có nghĩa là "với mọi số X, tồn tại Y sao cho số Y lớn hơn X". Ta có thể xem Y trong công thức đó là hàm của đối số X, chẳng hạn f(X) và loại bỏ lượng tử $\exists Y$, công thức đang xét trở thành $\forall X\ (P(X,f(X)))$. Ví dụ: f(X)=X+1, khi đó $\forall X\ (P(X,f(X))) \equiv \forall X\ (P(X, X+1))$ nghĩa là với mọi giá trị của X thì X+1 lớn hơn X.

Một cách tổng quát, giả sử $\exists Y(G)$ là một công thức con của công thức đang xét và nằm trong miền tác dụng của các lượng từ $\forall X_1,...,\forall X_n$. Khi đó ta có thể xem Y là hàm của n biến $X_1,...,X_n$, chẳng hạn $f(X_1,...,X_{n)}$. Sau đó ta thay các xuất hiện của Y trong công thức G bởi hạng thức $f(X_1,...,X_n)$ và loại bỏ các lượng từ tồn tại. Các hàm f được đưa vào để loại bỏ các lượng từ tồn tại được gọi là hàm Scholem.

Ví dụ: Xét công thức sau

$$\forall X\ (\exists Y\ (P(X,Y)) \vee \forall U\ (\exists V\ (Q(a,V) \wedge \exists Y\ (\neg\ R(X,Y)))) \quad (1)$$

Công thức con $\exists Y\ (P(X,Y)$ nằm trong miền tác dụng của lượng từ $\forall X$, ta xem Y là hàm của X: f(X). Các công thức con $\exists V\ (Q(a,V))$ và $\exists Y\ (\neg\ R(X,Y))$ nằm trong miền tác dụng của các lượng tử $\forall X$, $\forall U$ ta xem V là hàm g(X,U) và Y là hàm h(X,U) của hai biến X, U. Thay các xuất hiện của Y và V bởi các hàm tương ứng, sau đó loại bỏ các lượng từ tồn tại, từ công thức (1) ta nhận được công thức:

$$\forall X\ (P(X,f(X)) \vee \forall U\ (Q(a,g(X,U)) \wedge \neg\ R(X,h(X,U)))) \quad (2)$$

d. Loại bỏ các lượng từ phổ dụng

Sau bước 3 trong công thức chỉ còn lại các lượng từ phổ dụng và mọi xuất hiện của biến đều nằm trong miền tác dụng của các lượng từ phổ dụng. Ta có thể loại bỏ tất cả lượng từ phổ dụng, công thức (2) trở thành công thức:

$$P(X,f(X)) \vee Q(a,g(X,U)) \wedge \neg\ R(X,h(X,U))) \quad (3)$$

Cần chú ý rằng, sau khi được thực hiện bước này tất cả các biến trong công thức được xem là chịu tác dụng của các lượng tử phổ dụng.

e. Chuyển các tuyển tới các literal

Bước này được thực hiện bằng cách thay các công thức dạng: $P \vee (Q \wedge R)$ bởi $(P \vee Q) \wedge (P \vee R)$ và thay $(P \wedge Q) \vee R$ bởi $(P \vee Q) \wedge (P \vee R)$. Sau bước này công thức trở thành hội của các câu tuyển nghĩa là ta nhận được các công thức ở dạng chuẩn tắc hội. Chẳng hạn, công thức (3) được chuyển thành công thức sau:

$$(P(X,f(X)) \vee Q(a,g(X,U))) \wedge (P(X,f(X)) \vee \neg R(X,h(X,U))) \qquad (4)$$

f. Loại bỏ các hội

Một câu hội là đúng nếu và chỉ nếu tất cả các thành phần của nó đều là đúng. Do đó công thức ở dạng chuẩn tắc hội tương đương với tập các thành phần. Chẳng hạn, công thức (4) tong đong với tập hai câu tuyển sau:

$$P(f(X)) \vee Q(a,g(X,U))$$
$$P(f(X)) \vee \neg R(X,h(X,U)) \qquad (5)$$

g. Đặt tên lại các biến

Đặt tên lại các biến sao cho các biến trong các câu khác nhau có tên khác nhau, chẳng hạn hai câu (5) có hai biến cùng tên là X, ta cần đổi tên biến X trong câu hai thành Z, khi đó các câu (5) tương đương với các câu sau:

$$P(f(X)) \vee Q(a,g(X,U))$$
$$P(f(Z)) \vee \neg R(Z,h(Z,U)) \qquad (5')$$

Như vậy, khi tri thức là một tập hợp nào đó các công thức trong logic vị từ, bằng cách áp dụng thủ tục trên ta nhận được cơ sở tri thức chỉ gồm các câu tuyển (tức là ta luôn luôn có thể xem mỗi câu trong cơ sở tri thức là tuyển của các literal). Hoàn toàn tương tự như trong logic mệnh đề, mỗi câu tuyển có thể biểu diễn dưới dạng một kéo theo, vế trái của kéo theo là hội của các câu phân

tử, còn vế phải là tuyển của các câu phân tử. Dạng câu này được gọi là câu Kowlski, một trường hợp quan trọng của câu Kowlski là câu Horn (luật if- then)

Bài tập

1) Gọi vị từ nt(X) có nghĩa là "X là số nguyên tố" và vị từ sl(X) có nghĩa là "X là số lẻ".

Hãy diễn giải ý nghĩa của công thức: $\exists X$ (nt(X) and sl(X)).

Viết lại công thức trên sau khi lấy phủ định và diễn gii ý nghĩa của công thức đó.

2)Biến đổi công thức sau về dạng chuẩn tắc hội

$$\exists X \exists Y ((b(X) \wedge c(X)) \vee (d(Y) \wedge b(Y))$$

3) Gọi p(X,Y,Z) có nghĩa là: Z=X*Y, là một vị từ 3 biến trên tập số thực. Khi đó tính chất giao hoán của phép nhân X*Y=Y*X được diễn tả như sau:

$$\forall X,Y\ (p(X,Y,Z)) \Rightarrow p(Y,X,Z)$$

Hãy chuẩn hóa công thức trên (đưa về dạng chuẩn tắc hội)

1.3.3. Các luật suy diễn

Tất cả các luật suy diễn đã được đưa ra trong logic mệnh đề đều đúng trong logic vị từ cấp một. Bây giờ ta đưa ra một luật suy diễn quan trọng trong logic vị từ liên quan tới lượng tử phổ dụng

a. Luật thay thế phổ dụng

Giả sử G là một câu, câu $\forall X(G)$ là đúng trong một minh hoạ nào đó nếu và chỉ nếu G đúng với tất cả các đối tượng nằm trong miền đối tượng của minh hoạ đó. Mỗi hạng thức t ứng với một đối tượng, vì thế nếu câu $\forall X$ (G) đúng thì khi thay tất cả các xuất hiện của biến X bởi hạng thức t ta nhận được câu đúng. Công thức nhận được từ công thức G bằng cách thay tất ar các xuất hiện của x bởi t được ký hiệu là G[X/t]. Luật thay thế phổ dụng (universal instatiation) phát biểu rằng, từ công thức $\forall X$ (G) suy ra công thức G[X/t].

$$\frac{\forall X\,(G)}{G[X/t]}$$

Chẳng hạn, từ câu ∀X, like(X, "Football") (mọi người đều thích bóng đá), bằng cách thay X bởi An ta suy ra câu like("An", "Football") (An thích bóng đá).

b. Hợp nhất

Trong luật thay thế phổ dụng, ta cần sử dụng phép thế các biến bởi các hạng thức để nhận được các công thức mới từ công thức chứa các lượng tử phổ dụng. Ta có thể sử dụng phép thế để hợp nhất các câu phân tử (tức là để các câu trả lời thành đồng nhất). Chẳng hạn xét hai câu phân tử like("An",Y) và ∀x, like(X, "Football") mà để cho đơn giản ta bỏ đi các lượng tử phổ dụng. Sử dụng phép thế [X/An. Y/Football] hai câu trên trở thành đồng nhất like("An", "Football"). Trong các suy diễn, ta cần sử dụng phép hợp nhất các câu bởi các phép thế. Chẳng hạn, cho trước hai câu

friend(X,"Ba")⇒ good(X) (mọi bạn của Ba đều là tốt)

friend("Lan",Y) (Lan là bạn của của tất cả mọi người)

Ta có thể hợp nhất hai câu friend(X,"Ba")⇒ good(X) và friend("Lan",Y) bởi phép thay thế [X/Lan, Y/Ba]

friend("Lan","Ba")⇒ good("Lan")

friend("Lan","Ba")

Từ hai câu này, theo luật Modus Ponens, ta suy ra câu good("Lan") (Lan là người tốt)

Một cách tổng quát, một phép thế θ là một dãy các cặp X_i/t_i, θ = [X_1/t_1 X_2/t_2.... X_n/t_n] trong đó các X_i là các biến khác nhau, các t_i là các hạng thức và các X_i không có mặt trong t_i (i=1,....,n). Áp dụng phép thế θ vào công thức G, ta nhận được công thức G_0, đó là công thức nhận được từ công thức G bằng cách thay mỗi sự xuất hiện của các X_i bởi t_i. Chẳng hạn, nếu G=p(X,Y,f(a,X)) và θ=[X/b, Y/g(Z)] thì G_0=P(b,g(Z),f(a,b)).

Với hai câu phân tử G và H mà tồn tại phép thế θ sao cho G_0 và H_0 trở thành đồng nhất (G_0=H_0) thì G và H được gọi là hợp nhất được, phép thế θ được gọi là hợp nhất tử của G và H. Chẳng hạn, hai câu Like(An,y) và Like(x, Football) là hợp nhất được bởi hợp nhất tử [X/An, Y/Football]. Vấn đề đặt ra là với hai câu phân tử bất kỳ G và H, chúng có hợp nhất được không và nếu có thì làm thế nào tìm được hợp nhất tử?.

c. Luật Modus Ponens tổng quát

Giả sử P_i, P_i' (i=1,...,n) và Q là các công thức phân tử sao cho tất cả các cặp câu P_i, P_i' hợp nhất được bởi phép thế θ, tức là P_{i0}=P_{i0}' (i=1,...,n). Khi đó ta có luật:

$$\frac{(P_i \wedge ... \wedge P_n \Rightarrow Q),\ P_i',...,P_n'}{Q'}$$

Trong đó Q'=Q_θ

Ví dụ: Giả sử ta có các câu student(X) $\wedge$ male(X) $\Rightarrow$ like(X,"Football") và student("Anh"), male("Anh"). Với phép thế θ = [X/Anh], các cặp câu student(X), student("Anh") và male(X), male("Anh") hợp nhất được. Do đó ta suy ra câu like("Anh", "Football").

d. Luật phân giải tổng quát

- ***Luật phân giải trên các câu tuyển***

Giả sử ta có hai câu tuyển $A_1 \vee ... \vee A_m \vee C$ và $B_1 \vee ... \vee B_n \vee \neg D$, trong đó A_i (i=1,..., m) và B_j (j=1,..., n) là các literal, còn C và D là các câu phân tử có thể hợp nhất được bởi phép thế θ, $C_\theta\theta$=D_θ. Khi đó ta có luật:

$$\frac{A_1 \vee ... \vee A_m \vee C,\ B_1 \vee ... \vee B_n \vee \neg D}{A_1' \vee ... \vee A_m' \vee B_1' \vee ... \vee B_n'}$$

Trong đó A_i'=$A_i\theta$ (i=1,..., m) và B_j'=$B_j\theta$ (j=1,..., n)

Trong luật phân giải này, hai câu ở tử số (giả thiết) của luật được gọi là hai câu phân giải được, còn hai câu ở mẫu số (kết luận) của luật được gọi là phân giải thức của hai câu ở tử số. Ta sẽ ký hiệu của hai câu A và B là Res(A,B).

Ví dụ: Giả sử ta có hai câu A=hear(X, "Music") $\vee$ play(x, "Tennis") và B=$\neg$play("An",Y) $\vee$ study("An"). Hai câu play(X,"Tennis") và play("An", Y) hợp nhất được bởi phép thế θ=[X/An, Y/Tennis]. Do đó từ hai câu đã cho, ta suy ra câu hear("An", "Music") $\vee$ study("An"). Trong ví dụ này, hai câu A= hear(X, "Music") $\vee$ play(X, "Tennis") và B=$\neg$ play("An", Y) $\vee$ study("An") là phân giải được và phân giải thức của chúng là hear("An", "Music") $\vee$ study("An").

- ***Luật phân giải trên các câu Horn***

Câu Horn (luật If- then) là câu có dạng:

$$P_1 \wedge ... \wedge P_m \Rightarrow Q$$

Trong đó P_i (i=1,..., m; m$\geq$ 0) và Q là các câu phân tử.

Giả sử, ta có hai câu Horn $P_1 \wedge ... \wedge P_m \wedge S \Rightarrow Q$ và $R_1 \wedge ... \wedge R_n \Rightarrow T$, trong đó hai câu S và T hợp nhất được bởi phép thế θ, $S_\theta=T_\theta$. khi đó ta có luật:

$$\frac{P_1 \wedge ... \wedge P_m \wedge S \Rightarrow Q, \quad R_1 \wedge ... \wedge R_n \Rightarrow T}{P_1' \wedge ... \wedge P_m' \wedge R_1' \wedge ... \wedge R_n' \Rightarrow Q'}$$

Trong đó, $P_i'=P_i\theta$ (i=1,..., m), $R_j'=R_j\theta$ (j=1,..., n), $Q'=Q\theta$

Trong thực tế, chúng ta thường sử dụng trường hợp riêng sau đây. Giả sử S và T là hai câu phân tử, hợp nhất được bởi phép thế θ. Khi đó ta có luật:

$$\frac{P_1 \wedge ... \wedge P_m \wedge S \Rightarrow Q, \quad T}{P_1' \wedge ... \wedge P_m' \Rightarrow Q'}$$

Trong đó $P_i'=P_i\theta$ (i=1,..., m) và $Q'=Q\theta$

Ví dụ: Xét hai câu student(X) $\wedge$ male(X) $\Rightarrow$ play(X, "Football") và male("Ba"). Hai câu male("Ba") và male(X) hợp nhất được với phép thế [X/Ba], do đó từ hai câu trên ta suy ra student("Ba") $\Rightarrow$ play("Ba", "Football").

1.3.4. Thuật toán hợp nhất

Về mặt cú pháp, hạng thức và công thức phân tử có cấu trúc giống nhau, do đó ta gọi các hạng thức và các công thức phân tử là các biểu thức đơn .

Trong mục này, chúng ta sẽ trình bày thuật toán xác định hai biểu thức đơn cho trước có hợp nhất được hay không và nếu được thuật toán sẽ cho ra hợp nhất tử tổng quát nhất.

Như ta đã biết, một phép thế θ là một danh sách $[X_1/t_1\ X_2/t_2 \ldots\ X_n/t_n]$, trong đó các X_i là các biến khác nhau, t_i là các hạng thức không chứa X_i (i=1,...,n). Kết quả áp dụng phép thế θ vào biểu thức E là biểu thức Eθ nhận được từ E bằng cách thay mỗi xuất hiện của biến X_i bởi t_i. Hai biểu thức được xem là hợp nhất được nếu tồn tại phép thế θ để chúng trở thành đồng nhất và khi đó θ được gọi là hợp nhất tử của chúng. Chẳng hạn, xét hai biểu thức sau:

know("An", X)

know(Y, husband(Z))

Với phép thế θ = [Y/An, X/ husband(Z)], cả biểu thức trên trở thành

know("An", husband(Z))

Tuy nhiên, nếu hai biểu thức hợp nhất được thì nói chúng sẽ có vô số hợp nhất tử. Chẳng hạn, ngoài hợp nhất tử đã nêu, hai câu know("An", X) và know(Y, husband(Z)) còn có các hợp nhất tử sau:

[Y/An, X/ husband("Hoa"), Z/ Hoa]

[Y/An, X/ husband("Lan"), Z/ Lan]

...

Chúng ta sẽ gọi hợp thành của hai phép thế θ và η là phép thế θη sao cho với mọi biểu thức E ta có E(θη)=E(θ)η. Nói cụ thể hơn, hợp thành của phép thế

$\theta = [X_1/t_1\ \ X_2/t_2 .... \ X_n/t_m]$ và $\eta = [Y_1/s_1\ \ Y_2/s_2 .... \ Y_n/s_n]$ là phép thế $\theta\eta = [X_1/t_1\eta\ X_2/t_2\eta .... \ X_n/t_m\eta,\ Y_1/s_1\ Y_2/s_2 .... \ Y_n/s_n]$ trong đó ta cần loại bỏ các cặp Y_i/s_i mà Y_i trùng với một X_k nào đó.

Ví dụ Xét hai phép thế:

$$\theta = [X/a, V/g(Y,Z)]$$

$$\eta = [X/b, Y/c, Z/f(X), V/h(a)]$$

khi đó hợp thành của chúng là phép thế: $\theta\eta = [X/a, V/g(c,f(X)), Y/c, Z/f(X)]$

Quan sát ví dụ trên ta thấy rằng phép thế $\theta\eta$ áp đặt nhiều hạn chế cho các biến hn là θ. Do đó ta sẽ nói rằng phép thế θ tổng quát hơn phép thế $\theta\eta$.

Chẳng hạn, phép thế đã đưa ra ở trên [Y/An, X/ husband(Z)] tổng quát hơn phép thế [Y/An, X/ husband("Hoa"), Z/ "Hoa"].

Giả sử E và F là hai biểu thức đơn hợp nhất được. Ta gọi hợp nhất tử tổng quát nhất của E và F là hợp nhất tử θ sao cho θ tổng quát hơn bất kỳ hợp nhất tử δ nào của E và F. Nói một cách khác, θ là hợp nhất tử tổng quát nhất của E và F nếu với mọi hợp nhất tử δ của E và F đều tồn tại phép thế η sao cho $\delta=\theta\eta$. Chẳng hạn với E và F là các câu know(" An" , X) và know(Y, husband(Z)) thì hợp nhất tử tổng quát nhất là: $\theta = [Y/An, X/ husband(Z)]$

Sau đây chúng ta sẽ trình bày thuật toán hợp nhất, đó là thuật toán có ba tham biến: hai biểu thức đơn E, F và hợp nhất tử tổng quát nhất là θ. Thuật toán sẽ dừng và cho ra hợp nhất tử tổng quát nhất θ nếu E và F hợp nhất được. Nếu E và F không hợp nhất được, thuật toán cũng sẽ dừng và cho thông báo về điều đó.

Ta sẽ giả sử rằng E và F chứa các biến có tên khác nhau, nếu không ta chỉ cần đặt tên lại các biến.

Trong thuật toán ta cần tìm sự khác biệt giữa hai biểu thức. Sự khác biệt đực xác định như sau: Đọc hai biểu thức đồng thời từ trái sang phải cho tới khi gặp hai ký hiệu khác nhau trong biểu thức. Trích ra hai biểu thức con bắt đầu từ các ký hiệu khác nhau đó. Cặp biểu thức con đó tạo thành sự khác biệt giữa hai biểu

thức đã cho. Ví dụ: Sự khác biệt giữa hai câu like(X,f(a,g(Z)) và like(X,Y) là cặp (f(a,g(Z),Y), còn sự biệt giữa hai câu know(X,f(a,U)) và know(X,f(a,g(b))) là cặp (U,g(b)).

Thuật toán hợp nhất được mô tả bởi thủ tục đệ quy sau:

Procedure Unify(E, F, θ);

Begin

1. Xác định sự khác biệt giữa E và F;
2. If không có sự khác biệt then {thông báo thành công; stop}
3. If sự khác biệt là cặp (X, t), trong đó X là biến, t là hạng thức không chứa X then

begin

3.1. E ← E[X/t]; F ←F[X/t];

{tức là áp dụng phép thế [X/t] vào các biểu thức E và F}

3.2. θ ← θ[X/t];

{hợp thành của phép thế ? và phép thế [X/t] }

3.3. Unify(E, F, θ);

end

else

{thông báo thất bại; stop}

End;

Thuật toán hợp nhất trên luôn luôn dừng sau một số bước hữu hạn bước vì cứ mỗi lần bước 3 được thực hiện thì số biến còn lại trong hai biểu thức sẽ bớt đi một mà số biến trong hai biểu thức là hữu hạn. Để biết hai biểu thức P và Q có hợp nhất được hay không ta chỉ cần gọi thủ tục Unify(P,Q, θ), trong đó θ là phép thế rỗng.

P(a, X, f(a,g(X,Y)))

P(U, h(a), f(U,V)) (1)

Sự khác biệt giữa hai biểu thức này là (a,U). Thế U bởi a ta nhận được hai biểu thức sau:

P(a, X, f(a,g(X,Y)))

P(a, h(a), f(a,V)) (2)

Và phép thế θ =[U/a]

Sự khác biệt giữa hai biểu thức (2) là (X,h(a)). Thế x bởi h(a), ta có hai biểu thức sau:

P(a, h(a), f(a,g(h(a),Y)))

P(a, h(a), f(a,V)) (3)

Và phép thế θ =[U/a, X/h(a)]

Sự khác biệt giữa hai biểu thức (3) là (g(h(a),Y), V). Thế V bởi g(h(a),Y), hai biểu thức (3) trở thành đồng nhất:

P(a, h(a), f(a,g(h(a),Y))) (4)

Như vậy hai câu (1) hợp nhất được vi hợp nhất tử tổng quát nhất là:

θ =[U/a, X/h(a), V/g(h(a),Y)]

2. Một số thuật giải chứng minh.

Một trong những vấn đề khá quan trọng của logic mệnh đề là chứng minh tính đúng đắn của phép suy diễn (a □ b). Đây cũng chính là bài toán chứng minh thường gặp trong toán học.

Rõ ràng rằng với hai phép suy luận cơ bản của logic mệnh đề (Modus Ponens, Modus Tollens) cộng với các phép biến đổi hình thức, ta cũng có thể chứng minh được phép suy diễn. Tuy nhiên, thao tác biến đối hình thức là rất khó cài đặt được trên máy tính. Thậm chí điều này còn khó khăn với cả con người!

Với công cụ máy tính, bạn có thể cho rằng ta sẽ dễ dàng chứng minh được mọi bài toán bằng một phương pháp "thô bạo" là lập bảng chân trị . Tuy về lý thuyết,

phương pháp lập bảng chân trị luôn cho được kết quả cuối cùng nhưng độ phức tạp của phương pháp này là quá lớn, $O(2^n)$ với n là số biến mệnh đề.

Sau đây chúng ta sẽ nghiên cứu hai phương pháp chứng minh mệnh đề (ở mục 2.1 v à 2.2) với độ phức tạp chỉ có O(n).

Bài toán Cho tập các giả thiết dưới dạng các biểu thức logic mệnh đề $GT=\{GT_1, GT_2, ...GT_n\}$. Hãy chứng minh tập kết luận $KL=\{KL_1, KL_2,..., KL_m\}$.

2.1. Thuật giải Vương Hạo

Cơ sở lý luận Cho các giả thiết $GT_1, GT_2, ...,GT_n$. Để chứng minh tập kết luận $KL_1, KL_2,...,KL_m$, ta chứng minh $GT_1, GT_2,...,GT_n \rightarrow KL_1, KL_2,...,KL_m$: True

Thuật giải bao gồm các bước sau:

B1: Phát biểu lại giả thiết và kết luận của vấn đề theo **dạng chuẩn** sau :

$GT_1, GT_2, ..., GTn$ □ $KL_1, KL_2, ..., KLm$

Trong đó các GTi và KLi là các biểu thứclogic dạng chuẩn (chỉ chứa 3 phép toán cơ bản : □ , □ , □)

B2: Nếu GTi có phép □ thì tách thành hai dòng con.

Nếu ở Kli có phép □ thì tách thành hai dòng con.

Ví dụ: $p \wedge (□\ p\ □\ q)\ □\ q$ thì tách thành 2 dòng:

$p \wedge □$

$p\ □\ q$

và $p \wedge q\ □\ q$

Hoặc nếu có$p \wedge q \rightarrow q \vee r$ thì tách thành 2 dòng:

P ∧ q → q

và

p ∧ q → r

B3: Một dòng được chứng minh nếu:

1) Tồn tại chung một mệnh đề ở cả hai phía.

Ví dụ :

p ∧ **q** □ **q** ∨ r: True

2) **Tồn tại 2 mệnh đề phủ định lẫn nhau** (p và ¬p)

Ví dụ: p ∧ □ p □ q: True

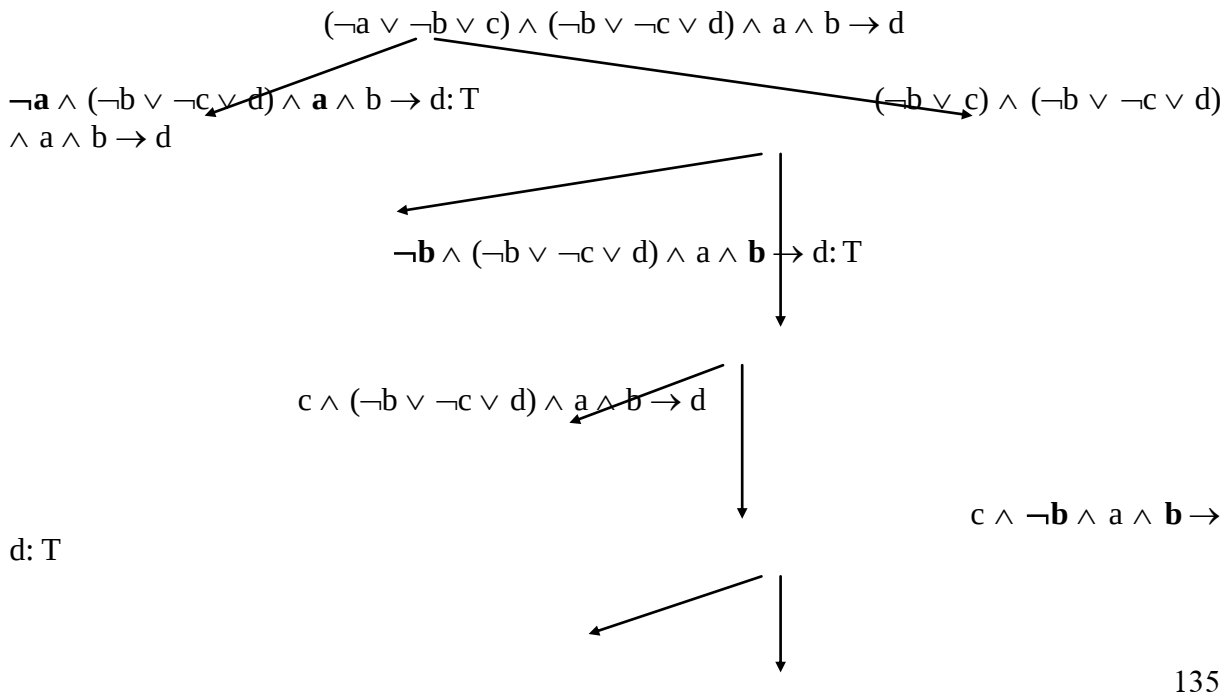
B4 :

a) Nếu một dòng không còn phép nối □ hoặc □ ở cả hai vế và ở 2 vế không có chung một biến mệnh đề thì dòng đó không được chứng minh.

b) Một vấn đề được chứng minh nếu tất cả dòng dẫn xuất từ dạng chuẩn ban đầu đều được chứng minh.

Ví dụ

1) Cần chứng minh rằng từ a ∧ b → c và b ∧ c → d và a và b, suy ra d



$$c \wedge (\neg c \vee d) \wedge a \wedge b \rightarrow d$$

$$\mathbf{c} \wedge \neg\mathbf{c} \wedge a \wedge b \rightarrow d: T$$

$$c \wedge \mathbf{d} \wedge a \wedge b \rightarrow \mathbf{d}: T$$

Cây suy diễn trong giải thuật Vương Hạo

2) Xét các câu đúng sau:

"Nếu trời mưa thì Lan mang theo dù"

"Nếu Lan mang theo dù thì Lan không bị ướt"

"Nếu trời không mưa thì Lan không bị ướt"

a) Xây dựng các câu trên bằng các biểu thức logic mệnh đề

b) Hãy chứng minh rằng "Lan không bị ướt" bằng phương pháp Vương Hạo

Lời giải

a) r: "Trời mưa"

u: "Lan mang theo dù"

w: "Lan bị ướt"

Lúc đó, ta có các biểu thức logic đúng sau:

$r \rightarrow u$

$u \rightarrow \neg w$

$\neg r \rightarrow \neg w$

Ta phải chứng minh $(r \rightarrow u) \wedge (u \rightarrow \neg w) \wedge (\neg r \rightarrow \neg w) \rightarrow \neg w$: True

2.2. Thuật giải Robinson

Thuật giải này do Robinson đề xuất và hoạt động dựa trên *phương pháp chứng minh phản chứng*.

Phương pháp chứng minh phản chứng:

Bài toán Chứng minh phép suy luận (a □ b) là đúng (với a là giả thiết, b là kết luận).

Phản chứng: giả sử b sai suy ra □ b là đúng.

Bài toán được chứng minh nếu a đúng và □ b đúng sinh ra một mâu thuẫn.

B1 : Phát biểu lại giả thiết và kết luận của vấn đề dưới dạng chuẩn như sau: $GT_1, GT_2, ...,GTn \square KL_1, KL_2, .., KLm$

Trong đó : GTi và KLj là các biểu thức logic dạng chuẩn (chỉ chứa các phép toán : $\square$, $\square$, $\square$)

B2 : Giả sử $\neg KL_1, \neg KL_2,...\neg KL_m$ là đúng, lúc đó ta có các biểu thức logic đúng sau:

$\{ GT_1, GT_2, ..., GTn , \square KL_1, \square KL_2, ..., \square KLm \}$

B3 : Sau đó đưa về dạng chuẩn hội (tích các tổng)

Ví dụ:

$$p \rightarrow q \wedge t \equiv \neg p \vee (q \wedge t) \equiv (\neg p \vee q) \wedge (\neg p \vee t)$$

B4 : Xây dựng một mệnh đề mới bằng cách áp dụng đồng nhất đúng:

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

Nghĩa là: nếu $(\neg p \vee q) \wedge (\neg p \vee r)$: True thì có thêm biểu thức $r \vee t$: True và đưa vào tập giả thiết.

Lặp lại quá trình trên cho đến khi sinh ra 2 mệnh đề có chân trị đối nhau (có sự mâu thuẫn) và bài toán lúc đó kết luận là được chứng minh, hoặc không tạo thêm mệnh đề mới nào gây mâu thuẫn và lúc này kết luận không chứng minh được.

Ví dụ

1) Xét bài toán:

Sau khi đưa về dạng chuẩn hội, để đơn giản, ta viết các biểu thức (chỉ chứa phép $\vee$) trên từng dòng. Ta có:

1. $\neg a \vee \neg b \vee c$
2. $\neg b \vee \neg c \vee d$

3. a

4. b

Giả sử ¬d đúng, ta có thêm các biểu thúc đúng sau:

5. ¬d

6. ¬b ∨ c (Res 1A, 3)

7. ¬a ∨ c (Res 1B, 4)

8. ¬c ∨ d (Res 2A, 4)

9. ¬b ∨ ¬c (Res 2C, 5)

10. c (Res 3, 7A)

11. ¬c (Res 4, 9A)

Mâu thuẫn giữa 10 và 11. Chứng minh xong.

2.3. Chứng minh bằng luật phân giải trên logic vị từ

Trong phần logic mệnh đề, ta đã đưa ra các luật suy diễn quan trọng như: luật Modus Ponens, luật Modus Tolens, luật bắc cầu,..., luật phân giải. Chúng ta đã chỉ ra rằng luật phân giải là luật đầy đủ cho bác bỏ. Điều đó có nghĩa là bằng phương pháp chứng minh bác bỏ, chỉ sử dụng luật phân giải ta có thể chứng minh được một công thức có là hệ quả logic của một tập các công thức cho trước hay không. Kết quả quan trọng này sẽ được mở rộng sang logic vị từ.

Giả sử ta có một cơ sở tri thức (CSTT) gồm các câu trong logic vị từ. Chúng ta luôn luôn xem CSTT đều đúng. Chẳng hạn minh hoạ đó là thế giới thực của vấn đề mà chúng ta đang quan tâm và CSTT gồm các câu mô tả sự hiểu biết của chúng ta về thế giới hiện thực đó.

Không mất tính tổng quát, ta có thể xem các câu trong CSTT là các câu tuyển. Chúng ta có thể sử dụng luật phân giải để suy ra các câu mới là hệ quả logic của CSTT.

Ví dụ: Giả sử CSTT gồm các câu tuyển sau:

$$\neg P(W) \vee Q(W) \qquad (1)$$

$$P(X) \vee R(X) \qquad (2)$$

$$\neg Q(Y) \vee S(Y)` \qquad (3)$$

$$\neg R(z) \vee S(z) \qquad (4)$$

Sau đây chúng ta sẽ đưa ra một chứng minh của câu S(a) từ CSTT trên bằng luật phân giải. Áp dụng luật phân giải cho câu (2) và (4) với phép thế [X/a, Z/a], ta suy ra câu sau:

$$P(a) \vee S(a) \qquad (5)$$

Áp dụng luật phân giải cho câu (1) và (3) với phép thế [w/a, y/a] ta nhận được câu:

$$\neg P(a) \vee S(a) \qquad (6)$$

Áp dụng luật phân giải cho câu (5) và (6), ta suy ra câu $S(a) \vee S(a)$. Câu này tương đương với câu S(a).

Chứng minh bằng cách áp dụng các luật suy diễn để dẫn tới điều cần phải chứng minh (như chứng minh trên) được gọi là chứng minh diễn dịch (deduction proof). Nhưng cần biết rằng luật phân giải không phải là luật đầy dủ cho diễn dịch, tức là từ một tập tiên đề, chỉ sử dụng luật phân giải chúng ta không thể sinh ra tất cả các câu là hệ quả logic của tiên đề đã cho.

Tuy nhiên định lý phân giải (trong mục logic mệnh đề) vẫn còn đúng trong logic vị từ cấp một là thoã được hay không thoã được. Nếu một tập câu là không thax được thì qua một số bước áp dụng luật phân giải sẽ sinh ra một câu rỗng (tức là dẫn tới mâu thuẫn).

Để chứng minh câu H là hệ quả logic của tập các câu $\{G_1, G_2,..., G_n\}$ (các tiên đề), ta có thể áp dụng phương pháp chứng minh bác bỏ, tức là chứng minh tập câu $\{G_1, G_2,..., G_n, \neg H\}$ không thoã được. Mặt khác, ở trên ta đã chỉ ra rằng

luật phân giải cho phép ta xác định được một tập câu là thoã được hay không thoã được. Vì vậy luật phân giải được xem là luật đầy đủ cho bác bỏ.

Ví dụ

Từ CSTT gồm các câu (1) đến (4) trong ví dụ trên, ta có thể chứng minh được câu S(a) bằng phưng pháp bác bỏ như sau. Thêm vào CSTT câu:

$$\neg S(a) \qquad (7)$$

(lấy phủ định của câu cần chứng minh), áp dụng luật phân gii cho câu (3) và (7) với phép thế [y/a], ta suy ra câu:

$$\neg Q(a) \qquad (8)$$

Từ câu (1) và (8) với phép thế [w/a], ta nhận được câu:

$$\neg P(a) \qquad (9)$$

Từ câu (4) và (7) với phép thế [z/a], ta suy ra câu:

$$\neg R(a) \qquad (10)$$

Từ câu (2) và (10) với phép thế [x/a], ta nhận được câu:

$$P(a) \qquad (11)$$

Áp dụng luật phân gii cho câu (9) và (11) ta nhận được câu rỗng (mâu thuẫn: $\neg P(a)$ và $P(a)$).

Thủ tục tổng quát sử dụng luật phân gii để chứng minh một công thức H có là hệ qu logic của một tập công thức $G=[G_1, G_2,...,G_n]$

Procedure Proof_by_Resolution;

Input: tập $G=[G_1, G_2,...,G_n]$ các công thức {các tiên đề}, công thức cần chứng minh H

Begin

1. Biến đổi các công thức G_i (i=1,....,n) vµ $\neg$ H về dạng chuẩn hội;
2. Từ các dạng chuẩn hội nhận được ở bước 1, thành lập tập các câu tuyển C;
3. Repeat

3.1. Chọn ra hai câu A và B trong C;

3.2. If A và B phân gii được then

Tính phân gii thức Res(A,B);

3.3. If Res(A,B) là câu mới then

Thêm Res(A,B) vào tập C;

Until nhận được câu rỗng hoặc không có câu mới nào được sinh ra;

4. If câu rỗng được sinh ra then

Thông báo H đúng

Else

Thông báo H sai;

End;

3. Ví dụ và bài tập.

Để trình bày gọn, chúng ta quy ước

- Phép và (hội hay tích)

Ví dụ a và b được ký hiệu $a \wedge b$ hoặc ab

- Phép hoặc (tuyển hay tổng)

Ví dụ a hoặc b được ký hiệu $a \vee b$ hoặc a+b

- Phép phủ định

Ví dụ phủ định của a được ký hiệu $\neg a$

- Phép kéo theo (suy ra)

Ví dụ a kéo theo b được ký hiệu $a \rightarrow b$

Ví dụ 1 Cho các khẳng định đúng với các ý nghĩa như sau:

A MỘT NGƯỜI CÓ NHÓM MÁU A

b một người có nhóm máu B

c một người có nhóm máu AB

o một người có nhóm máu O

t mẫu máu của một người có xét nghiệm T dương tính

s mẫu máu của một người có xét nghiệm S dương tính

Hãy viết các biểu thức logic diễn tả những ý tưởng sau:

a. Nếu xét nghiệm T dương tính thì người đó có nhóm máu A hoặc AB
b. Nếu xét nghiệm S dương tính thì người đó có nhóm máu B hoặc AB
c. Nếu một người có nhóm máu B thì xét nghiệm S sẽ dương tính
d. Một người có nhóm máu A, B, AB hoặc O

Lời giải

a. $t \rightarrow a+c$

b. $s \rightarrow b+c$

c. $b \rightarrow s$

d. a + b+ c + o

VÍ DỤ 2. HÃY BIỂU DIỄN CÁC TRI THỨC SAU BẰNG LOGIC MỆNH ĐỀ

a. Nếu n là số nguyên chẵn và n là số nguyên tố thì n=2

b. Số n là chính phương thì n tận cùng bằng 1, 4, 5, 6 hoặc 9

Lời giải

a. Gọi a : "là một số nguyên chẵn",
b : "là một số nguyên tố",
c : "số nguyên 2".

Lúc đó, tri thức sẽ được biểu diễn như sau: $ab \rightarrow c$

b. Gọi a : "là một số chính phương",
b : "số có chữ số tận cùng bằng 1"
c : "số có chữ số tận cùng bằng 4"
d : "số có chữ số tận cùng bằng 5"
e : "số có chữ số tận cùng bằng 5"
f : "số có chữ số tận cùng bằng 9".

Lúc đó, tri thức sẽ được biểu diễn như sau: a→ b+c+d+e+f

Ví dụ 3 Cho các biểu thức logic mệnh đề đúng sau

1. $a \rightarrow f$
2. $a \rightarrow (f \rightarrow p)$
3. $p+q \rightarrow d$
4. a
5. $ad \rightarrow g$

Hãy dùng phương pháp Robinson và Vương Hạo để chứng minh hoặc bác bỏ g≡1

Lời giải

- **Chuyển về dạng chuẩn**

$a \rightarrow f \equiv \neg a + f$

$a \rightarrow (f \rightarrow p) \equiv \neg a + (f \rightarrow p) \equiv \neg a + (\neg f + p) \equiv \neg a + \neg f + p$

$p+q \rightarrow d \equiv \neg(p+q)+d \equiv (\neg p \neg q)+d \equiv (\neg p + d)(\neg q + d)$

$ad \rightarrow g \equiv \neg(ad)+g \equiv \neg a+\neg d+g$

- **Dùng phương pháp phân giải Robinson**

GIẢ SỬ ¬G≡1, TA CÓ CÁC BIỂU THỨC ĐÚNG SAU

1. $\neg a + f$
2. $\neg a + \neg f + p$
3. $\neg p + d$
4. $\neg q + d$
5. a
6. $\neg a+\neg d+g$
7. $\neg g$
8. $\neg a+\neg d$ $res(6_3,7)$

9. ¬d res(5, 8_1)

10. ¬a +¬f +d res(2_3,3_1)

11. ¬a + d res(1_2,10_2)

12. d res(5,11_1)

Ta thấy 9) kết hợp 12) sẽ cho ra câu rỗng (tức là có sự mâu thuẫn). Vì vậy g≡1.

- **Dùng phương pháp Vương Hạo**

Ta cần chứng minh:

(a)(¬a + f)(¬a + ¬f + p)(¬p + d)(¬q + d)(¬a+¬d+g)→g: true(I)

(1) (2) (3) (4) (5)

Để chứng minh (I) tách (1), biểu thức (I) trở thành:

I.1) (a)(¬a)(¬a + ¬f + p)(¬p + d)(¬q + d)(¬a+¬d+g) → g : true

I.2) af (¬a + ¬f + p)(¬p + d)(¬q + d)(¬a+¬d+g)→g

Chứng minh I.2) tách 2), I.2) trở thành

I.2.1) af (¬a)(¬p + d)(¬q + d)(¬a+¬d+g) → g: true

I.2.2) af (¬f)(¬p + d)(¬q + d)(¬a+¬d+g) → g: true

I.2.3) afp(¬p + d)(¬q + d)(¬a+¬d+g) → g

Chứng minh I.2.3) tách 3), I.2.3) trở thành

I.2.3.1) afp(¬p)(¬q + d)(¬a+¬d+g) → g: true

I.2.3.2) afpd(¬q + d)(¬a+¬d+g) → g

Chứng minh I.2.3.2) tách 5), I.2.3.2) trở thành

I.2.3.2.1) afpd(¬q + d)(¬a) → g : true

I.2.3.2.2) afpd(¬q + d)(¬d) → g : true

I.2.3.2.3) afpd(¬q + d)g → g : true

(I) được chứng minh, vì vậy g≡1

Bài tập1. Biểu diễn các tri thức sau dưới dạng logic mệnh đề

a. Trong tam giác vuông, tổng bình phương chiều dài hai cạnh góc vuông bằng bình phương chiều dài cạnh huyền
b. Một số nguyên dương có chữ số hàng đơn vị bằng 5 thì số đó chia hết cho 5
c. Nếu x là số lẻ và bình phương của x tận cùng bằng 1 thì x tận cùng bằng 1 hoặc bằng 9
d. Trong tam giác vuông, chiều dài của đườn trung tuyến xuất phát từ góc vuông bằng nữa chiều dài của cạnh huyền

Bài tập 2. Cho các biểu thức logic mệnh đề đúng sau

1. $n+c+d \rightarrow p$
2. $qp \rightarrow c$
3. $qc \rightarrow f + \neg h$
4. $\neg n + \neg p + h$
5. nq

Hãy dùng phương pháp Robinson và Vương Hạo để chứng minh hoặc bác bỏ $f \equiv 1$

Bài tập 3. Cho các biểu thức logic mệnh đề đúng sau

1. $abc \rightarrow c$
2. $abc \rightarrow p$
3. $as \rightarrow h$
4. $abcp \rightarrow s$
5. abd

Hãy dùng phương pháp Robinson và Vương Hạo để chứng minh hoặc bác bỏ $sh \equiv 1$

Bài tập 4. Ta có cơ sở tri thức của hệ chuyên gia về ***bệnh cảm cúm*** như sau:

1) "*Nếu* bệnh nhân rát họng và viêm nhiễm *thì* viêm họng và đi chữa họng"
2) "*Nếu* thân nhiệt >37^0 *thì* sốt"
3) " *Nếu* ốm trên 7 ngày và sốt *thì* viêm nhiễm"
4) "*Nếu* sốt và ho và kèm theo khó thở hoặc kèm theo tếng ran thì viêm phổi"

a) Hãy biểu diễn các tri thức trên dưới dạng logic mệnh đề.

b) Có một bệnh nhân khai : "Thân nhiệt > 37^0 " và "ốm trên 7 ngày". Dùng phương pháp chứng minh Robinson và Vương Hạo để kết luận bệnh nhân này bị "viêm nhiễm".

Bài tập 5. Ta có cơ sở tri thức mô tả mối quan hệ của các thành phần trong một tam giác như sau:

- Nếu biết 3 cạnh của 1 tam giác ta có thể biết nủa chu vi của tam giác đó
- Nếu biết 2 cạnh và nữa chu vi của một tam giác thì ta có thể biết được cạnh còn lại của tam giác đó
- Nếu biết được diện tích và một cạnh của một tam giác thì ta có thể biết được chiều cao tương ứng với cạnh đó
- Nếu biết 2 cạnh và một góc kẹp giữa 2 cạnh đó của một tam giác thì ta có thể biết được cạnh còn lại của tam giác đó.
- Nếu biết 2 cạnh và một góc kẹp giữa 2 cạnh đó của một tam giác thì ta có thể biết được diện tích của tam giác đó
- Nếu biết ba cạnh và nữa chu vi của một tam giác thì ta biết được diện tích của tam giác đó
- Nếu biết diện tích và đường cao của một tam giác thì ta biết được cạnh tương ứng với đường cao của tam giác đó

Giả sử biết được 2 cạnh và và góc kẹp giữ hai cạnh đó. Bằng phương pháp Robinson, hãy chứng minh rằng ta có thể suy ra được đường cao tương ứng với cạnh còn lại

Hướng dẫn

Ký hiệu a: cạnh a của tam giác **k: đường cao tương ứng với cạnh a**

b: cạnh b của tam giác **l: đường cao tương ứng với cạnh b**

c: cạnh c của tam giác **m: đường cao tương ứng với cạnh c**

A: góc tương ứng với cạnh a **S: diện tích của tam giác**

B: góc tương ứng với cạnh b **p: nữa chu vi của tam giác**

C: góc tương ứng với cạnh c

- Các tri thức đó được biểu diễn dưới dạng logic mệnh đề như sau:

10) abc $\rightarrow$ p

11) bpc $\rightarrow$ a

12) apc $\rightarrow$ b

13) abp $\rightarrow$ c

14) Sa $\rightarrow$ k

15) Sb $\rightarrow$ l

16) Sc $\rightarrow$ m

17) abC $\rightarrow$ c

18) acB $\rightarrow$ b

19) bcA $\rightarrow$ a

20) abC $\rightarrow$ S

21) acB $\rightarrow$ S

22) bcA $\rightarrow$ S

23) abcp $\rightarrow$ S

24) Sk $\rightarrow$ a

25) Sl $\rightarrow$ b

26) Sm $\rightarrow$ c

Sau đó dùng phương pháp Robinson (GT={a, b}, KL={m})

Bài tập 6. ***Biểu diễn các tri thức sau dưới dạng logic vị từ***

a. Bất kỳ người nào cũng có cha mẹ

b. Mọi số nguyên tố lớn hơn 2 đều là số lẻ

c. Chuồn chuồn bay thấp thì mưa

Lời giải

a. Ký hiệu NGUOI(X): nghĩa là X là người

CHAME(X, Y): X là cha mẹ của Y

$\forall$X (NGUOI(X) $\rightarrow$ $\exists$Y: CHAME (Y, X))

b. Ký hiệu P(X): X là số nguyên tố lớn hơn 2

Q(X): X là số lẽ

$\forall$X (P(X) $\rightarrow$ Q(X))

c. Ký hiệu BAY(X,Y): con vật X bay với độ cao Y

TROIMUA: trời mưa

BAY(“chuồn chuồn”, “thấp”) $\rightarrow$ TROIMUA

Bài tập 7.

Giả sử chúng ta biết các thông tin sau đây:

1) **Ông Ba nuôi một con chó**
2) **Hoặc ông Ba hoặc ông Am đã giết con mèo Bibi**
3) **Mọi người nuôi chó đều yêu quý động vật**
4) **Ai yêu quý động vật cũng không giết động vật**
5) **Chó mèo đều là động vật**

Ai đã giết Bibi?

Lời giải

- Biểu các thông tin trên dưới dạng logic vị từ cấp một như sau

Để biểu diễn các tri thức trên trong logic vị từ cấp một, chúng ta cần sử dụng các hằng D, Ba, An, Bibi, các vị từ Dog(x) (x là chó), Cat(y) (y là mèo), Rear(u,v) (u nuôi v), AnimalLover(u) (u là người yêu quý động vật), Kill(u,v) (u giết v), Animal(x) (x là động vật).

Sử dụng các hằng và các vị từ trên, chúng ta có thể chuyển các trên thành các câu trong logic vị từ cấp một như sau:

1) Dog(“D”) Rear(“Ba”, “D”)

2) Cat(“Bibi”) (Kill(“Ba”, “Bibi”) + Kill(“Am”, “Bibi”))

3) ∀X (∀Y(Dog(Y) Rear(X,Y)) → AnimalLover(X)))

4) ∀U (AnimalLover(U) → (∀V AnimalLover(V) →¬ Kill(U,V)))

5) ∀Z (Dog(Z) → Animal(Z)) ∀W (Cat(W) →Animal(W))

- Chuyển về dạng chuẩn và dùng phương pháp phân giải Robinson

1) Dog(“D”)

2) Rear(“Ba”, “D”)

3) Cat(“Bibi”)

4) Kill(“Ba”, “Bibi”) + Kill(“Am”, “Bibi”)

5) ¬Dog(Y) + ¬Rear(X,Y) + AnimalLover(X)

6) ¬AnimalLover(U) + ¬Animal(V) + ¬Kill(U,V)

7) ¬Dog(Z) + Animal(Z)

8) ¬Cat(W) + Animal(W)

Giả sử ¬Kill(T, “Bibi”) đúng

9) ¬Kill(T, “Bibi”)

Từ câu (4) và câu (9) với phép thế [t/Am], ta nhận được câu:

10) Kill(“Ba”, “Bibi”)

Từ câu (6) và câu (10) với phép thế [u/Ba, v/Bibi], ta nhận được câu:

11) ¬AnimalLover(“Ba”) + ¬Animal(“Bibi”)

Từ câu (3) và câu (8) với phép thế [w/Bibi], ta nhận được câu:

12) Animal(“Bibi”)

Từ câu (11) và câu (12), ta nhận được câu:

13) ¬AnimalLover(“Ba”)

Từ câu (1) và câu (5), với phép thế [y/D] ta nhận được câu:

14) ¬ Rear(X, “D”) + AnimalLover(X)

Từ câu (2) và câu (14), với phép thế [x/Ba] ta nhận được câu:

15) AnimalLover(“Ba”)

Từ câu (13) và câu (15) ta suy ra câu rỗng (có sự mâu thuẫn). Như vậy ông Am đã giết con mèo Bibi.

Bài tập 8. Giả sử chúng ta biết các thông tin sau đây:

a. Mọi người đếu chết

b. Mọi phụ nữ đều chết

c. Thần thánh không chết

d. Tất cả cả những người bệnh phải được điều trị

e. Beatrice là phụ nữ

f. Christel là phụ nữ

g. Marta là phụ nữ

h. Socrate là người

i. Zeus là thần thánh

k. Socrate bị bệnh

Dùng phương pháp phân giải Robinson để có thể suy ra được Socrate có được điều trị hay không?

. . .

Chương 5

TRI THỨC VÀ CÁC PHƯƠNG PHÁP SUY DIỄN

Như ta đã biết con người sống trong môi trường có thể nhận được thế giới nhờ các giác quan và sử dụng tri thức tích luỹ được và nhờ khả năng lập luận, suy diễn, con người có thể đưa ra các hành động hợp lý cho công việc mà con người đang làm. Trong khi đó mục tiêu của trí tuệ nhân tạo ứng dụng là thiết kế các **tác nhân thông minh (intelligent agent)** cũng có khả năng đó như con người. (Tác nhân thông minh là bất cứ cái gì có thể nhận thức được môi trường thông qua các bộ cảm nhận (sensors) và đưa ra hành động hợp lý đáp ứng lại môi trường thông qua bộ phận hành động (effectors). **Ví dụ:** robots, softrobot (software robot), các hệ chuyên gia,...là các tác nhân thông minh).

1. Tri thức và dữ liệu

- Tri thức là sự hiểu biết về một miền chủ đề (lĩnh vực) nào đó.

VÍ DỤ - HIỂU BIẾT VỀ Y HỌC, VĂN HỌC,.... LÀ TRI THỨC

- Thu thập thông tin ta được dữ liệu và căn cứ vào tri thức ta có được những quyết dịnh phán đoán.

Đối với quả cam ta xét các dữ liệu như vỏ, cuống, màu sắc,...của nó như thế nào? và dựa vào hiểu biết của ta mà xác định xem quả cam đó là ngon hay không ngon, ngon vừa,...

Như vậy, tri thức là dạng dữ liệu bậc cao. Khó phân biệt giữa tri thức và dữ liệu (không có ranh giới rõ ràng giữa chúng). Tuy nhiên ta có thể phân biệt theo bảng sau:

Dữ liệu	**Tri thức**
- Định lượng - Có cấu trúc đơn giản - Ở dạng đơn giản	- Định tính - Không có cấu trúc hoặc có cấu trúc phức hợp - Ở dạng phức hợp

2. Các dạng mô tả tri thức (các phương pháp biểu diễn tri thức)

(Để máy tính có thể sử dụng được tri thức, có thể xử lý được tri thức, chúng ta cần phải biểu diễn tri thức dưới dạng thuận tiện cho máy tính. Đó là mục tiêu của biểu diễn tri thức). Sau nhiều cố gắng, các nhà TTNT đã phát triển một số cách biểu diễn (thể hiện) tri thức có hiệu quả trong máy.

2.1. Biểu diễn tri thức bằng logic

Như ta đã nghiên cứu ở phần trước, ta có thể biểu diễn bài toán bằng các biểu thức logic (logic mệnh đề, logic vị từ)

2.2. Biểu diễn tri thức bằng mạng ngữ nghĩa

Phương pháp biểu diễn tri thức bằng cách dùng một đồ thị G = (V, E) gồm tập đỉnh V và tập cung E. Trong đó các đỉnh ứng với các đối tượng, khái niệm hay sự kiện cụ thể, các cung thể hiện quan hệ giữa các đối tượng. Có một cung nối giữa hai đối tượng a và đối tượng b, ký hiệu a ⟶ b nếu có một quan hệ nào đó giữa hai đối tượng a, b.

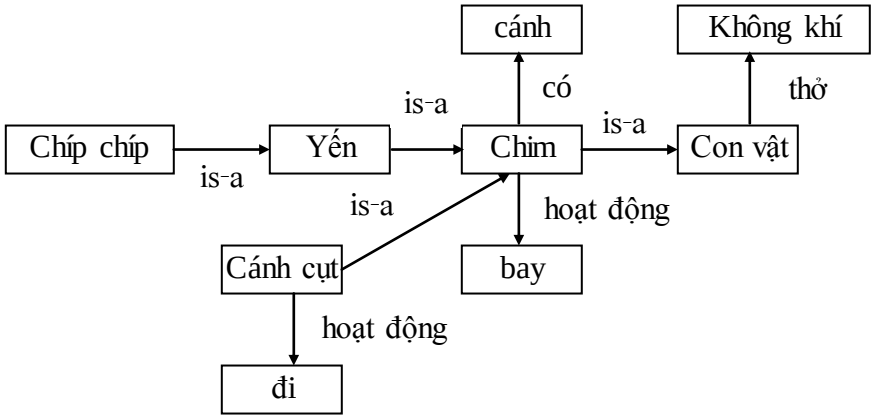
Có 2 loại quan hệ đặc biệt

- "a là b" nghĩa là đối tượng a thuộc vào tập đối tượng được biểu diễn bởi khái niệm b hoặc tập các đối tượng biểu diễn bởi khái niệm a là tập con của tập đối tượng biểu diễn khái niệm b. (quan hệ is-a)

Ví dụ Yến ⟶ chim

- Ngược lại với quan hệ "là" là quan hệ "bao gồm". Khi có " a là b" (hoặc "b bao gồm a"), các thông tin cơ bản về các đối tượng được cho bởi b sẽ truyền lại cho a (nghĩa là a được thừa hưởng những gì b có).

VÍ DỤ



Ưu điểm:

- Cho phép biểu diễn một cách trực quan các sự kiện và các mối liên hệ giữa chúng.
- Tính mô đun cao theo nghĩa các tri thức mới được thêm vào hoàn toàn độc lập với các tri thức cũ.
- Có thể áp dụng một số cơ chế suy diễn trên mạng: cơ chế truyền và thừa hưởng thông tin giữa các đối tượng, cơ chế "cháy" trên mạng

Nhược điểm:

- Không có một phương pháp suy diễn chung nào cho mọi loại mạng ngữ nghĩa
- Khó kiểm soát quá trình cập nhật tri thức để dẫn đến mâu thuẫn trong cơ sở tri thức.

2.3. Biểu diễn tri thức bằng khung (Frame)

Khung thực chất là sự tổng quát hoá của cấu trúc bản ghi trong Pascal và tương tự như cấu trúc đối tượng trong C^{++}

Một khung được mô tả bởi cấu trúc:

- Tên khung: Định danh đối tượng mô tả
- Các khe (slot): trên mỗi khe lưu trữ các thông tin, n\miền giá trị, thuộc tính và chiều mũi tên chỉ đến các khung khác

VÍ DỤ xÉT KHUNG (FRAME) MÔ TẢ TẬP HỌC SINH

HOCSINH

Frame **HOCSINH**

IS-A:

PART-OF: NGUOI-DI-HOC

A KIND OF: (HOCSINHCOSO, HOCSINHTRUNGHOC)

Cân nặng: 10-60kg

Chiều cao: 80-170cm

Cấu trúc frame này cho ta một "khung dữ liệu" để khoanh vùng các đối tượng là học sinh. Trường hợp gặp một người cao 175cm, nặng 45kg thì ta có thể khẳng định rằng đó không phải là học sinh vì không thoã mãn các ràng buộc đã có.

Ngoài ra, một trong những đặc trưng quan trọng của frame là khả năng thừa kế các thông tin của các khe có cùng tên ở đối tượng bậc trên.

Ví dụ Trong frame HOCSINHCOSO, HOCSINHTRUNGHOC có khe chiều cao với giá trị mô tả miền, thì sau khi thừa kế thông tin ở mức trên Frame HOCSINH, khe này cần phải lấy các giá trị trong khoảng 80-170cm.

2.4. Biểu diễn tri thức bằng các luật sản xuất

Phương pháp biểu diễn tri thức nhờ logic (logic mệnh đề và logic vị từ) khá trực quan song chỉ phù hợp khi không có quá nhiều luật suy diễn.

Một tri thức được thể hiện bằng một **câu Horn dạng chuẩn:**

$$p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q$$

(Các câu Horn dạng này còn được gọi là luật if- then và được biểu diễn như sau: ***if P_1 and....and P_m then Q***)

Một câu Horn dạng tổng quát:

$$p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q_1 \vee q_2 \vee \ldots \vee q_m$$

Lưu ý:

Nếu có luật dạng: $p_1 \wedge p_2 \wedge ....\wedge p_n \Rightarrow q_1 \vee q_2 \vee ....\vee q_m$ thì tương đương với m luật sau:

$$p_1 \wedge p_2 \wedge ....\wedge p_n \wedge \neg q_2 \wedge ....\wedge \neg q_m \Rightarrow q_1$$

$$p_1 \wedge p_2 \wedge ....\wedge p_n \wedge \neg q_1 \wedge \neg q_3 ...\wedge \neg q_m \Rightarrow q_2$$

$$p_1 \wedge p_2 \wedge ....\wedge p_n \wedge \neg q_1 ....\wedge \neg q_{m-1} \Rightarrow q_m$$

Tuy nhiên ta chỉ xét câu Horn dạng chuẩn (m=1)

- Nếu n=0, m=1: câu Horn có dạng $\Rightarrow q$: gọi là sự kiện (fact) q.
- Nếu n>0, m=1: câu Horn có dạng: $p_1 \wedge p_2 \wedge ....\wedge p_n \Rightarrow q$: gọi là luật (rule).

Trong các hệ chuyên gia, cơ sở tri thức gồm 2 phần: tập các sự kiện (facts) và tập luật (rules).

VÍ DỤ

1) Ta có các luật về kinh nghiệm dự báo thời tiết:

"Chuồn chuồn bay thấp thì mưa, bay cao thì nắng, bay vừa thì râm"

a: chuồn chuồn bay thấp, b: chuồn chuồn bay cao, c: chuồn chuồn bay vừa

d: trời mưa, e: trời nắng, f: trời râm

lúc đó ta có các luật sau:

$$a \Rightarrow d$$

$$b \Rightarrow e$$

$$c \Rightarrow f$$

2) Nhiều định lý trong toán học có thể biểu diễn bởi các luật, ví dụ:

Nếu tam giác có một góc bằng 60^0 và tam giác có hai cạnh bằng nhau *thì* tam giác đó là tam giác đều.

3. Suy diễn trên luật sản xuất

3.1. Khái niệm

Suy diễn là quá trình suy luận dựa vào các quy luật đã cho, thiết lập các thông tin mới từ các thông tin đã biết. Suy diễn sẽ sử dụng tập sự kiện làm tiên đề.

Các phương pháp suy diễn dần dần chuyển từ các giả thiết về các kết luận bằng cách thêm vào giả thiết những sự kiện đã được khẳng định đúng, dựa trên 2 phương thức:

- Modus ponens: $\frac{A,\ A \Rightarrow B}{B}$

nghĩa là nếu A đúng và A⇒B đúng thì B cũng đúng

- Modus tollens $\frac{\neg B,\ A \Rightarrow B}{\neg A}$

nghĩa là nếu B sai và biết rằng A⇒B đúng thì A cũng sai.

Trong quá trình suy diễn, ta cần quan tâm đến các vấn đề sau:

- Xây dựng tập luật, câu hỏi nào được chọn để người sử dụng trả lời
- Chọn quá trình tìm kiếm như thế nào
- Thông tin nhận được có ảnh hưởng đến quá trình tìm kiếm không

3.2. Bài toán

Cho tập sự kiện F= $\{f_1, f_2,...,f_n\}$ và tập luật R= $\{r_1, r_2,...,r_m\}$. Chứng minh tập kết luận G đúng.

3.3. Các phương pháp suy diễn

Quá trình suy diễn trong hệ luật sản xuất bao gồm 2 phương pháp cơ bản: suy diễn tiến và suy diễn lùi.

a) Suy diễn tiến (lập luận tiến - forward chaining hoặc forward reasoning)

(Tư tưởng cơ bản của suy diễn tiến là áp dụng luật suy diễn Modus Ponens tổng quát)

Là quá trình suy diễn bắt đầu từ tập sự kiện đã biết, rút ra những sự kiện mới và cứ như vậy cho đến khi có được sự kiện cần chứng minh hoặc không có luật nào sinh ra các sự kiện mới (tập sự kiện đúng là cực đại).

- **Phương pháp**

GỌi T là tập các sự kiện tại thời điểm đang xét (khởi tạo tập T=F: tập sự kiện đúng ban đầu).

Xét các luật r_i có dạng: $p_1 \wedge p_2 \wedge ....\wedge p_n \Rightarrow q$ và $p_j \in T$ $\forall_j = \overline{1,n}$ nghĩa là left (r_i) $\in$ T

thì T= T+ right (r_i)

quá trình lặp lại cho đến khi G$\subset$ T hoặc không có luật nào sinh ra thêm sự kiện mới.

- **Giải thuật**

Procedure suydientien;

```
Begin
    T:= F;
    S:= loc(R, T); { S: là tập luật có dạng p1 ∧ p2 ∧ ....∧ pn ⇒ q sao cho pj∈T ∀j = 1,n }
    While G ⊄ T and S<>ϕ do
        Begin
            r := get(S);
            T:= T + right(r);
            R:=R \ {r};
            S:= loc(R,T);
        End;
    If G ⊂ T then write ("thành công")
    Else write ("không thành công");
End;
```

Ví dụ

1) Cho trước tập sự kiện F={a,b}. Sử dụng các luật:

r_1: a $\Rightarrow$ c

r_2: b $\Rightarrow$ d

r_3: c $\Rightarrow$ e

r_4: a $\wedge$ d $\Rightarrow$ e

r_5: $b \wedge c \Rightarrow f$

r_6: $e \wedge f \Rightarrow g$

cần suy ra g

r	T	S	R
	a, b	r_1, r_2, r_3	r_1, r_2, r_3, r_4, r_5, r_6
r_1	a, b, c	r_2, r_3, r_5	$r_{2,...}r_6$
r_2	a, b, c, d	r_3, r_4, r_5	r_3,..., r_6
r_3	a, b, c, d, e	r_4, r_5	r_4, r_5, r_6
r_4	a, b, c, d, e	r_5	r_5, r_6
r_5	a, b, c, d, e, f	r_6	r_6
r_6	a. b, c, d, e, f, g		

$g \in T$ nên bài toán được chứng minh (g: true)

Chú ý

- Quá trình suy diễn tiến là quá trình xem xét các luật, với mỗi luật ta xét phần điều kiện (ở vế trái) tới phần kết luận (ở vế phải) và khi mà tất cả các điều kiện của luật đều thoã mãn thì ta suy ra sự kiện trong phần kết luận. Chính vì lẽ đó mà có tên là suy diễn tiến.
- Trong mỗi bước của thủ tục, người ta xét một luật trong tập luật. So sánh mỗi điều kiện (ở vế trái) của tập luật với các sự kiện trong cơ sở sự kiện, nếu tất cả các điều kiện của luật được thoã mãn thì sự kiện trong phần kết luận được xem là sự kiện được suy ra. nếu sự kiện này là sự kiện mới (không có trong bộ nhớ làm việc) thì nó được đưa vào bộ nhớ làm việc. Quá trình trên cứ lặp lại cho đến khi nào không có luật nào sinh ra sự kiện mới.

- Quá trình suy diễn tiến không định hướng tới giải quyết một vấn đề nào cả, không hướng tới tìm ra câu trả lời cho một câu hỏi nào cả. Suy diễn tiến chỉ là quá trình suy ra các sự kiện mới từ các sự kiện có trong bộ nhớ làm việc.

b) Suy diễn lùi (lập luận lùi - backward chaining hoặc backward reason)

Là quá trình xuất phát từ sự kiện cần chứng minh và thay vào đó là những sự kiện ở vế trái của 1 luật có vế phải là sự kiện cần chứng minh. Quá trình này được thực hiện cho đến khi đưa về các sự kiện là tập sự kiện con của tập sự kiện giả thiết.

(Nghĩa là: để đưa ra kết luận b, ta thử tìm tất cả các luật có dạng: $a_1 \wedge ....\wedge a_n \Rightarrow$ b, để có b, phải đưa ra các kết luận $a_1,...,a_n$. Quá trình xác định a_i cũng tương tự như đối với b, nếu đến một lúc nào đó phát hiện được rằng có một a_i nào đó không dẫn xuất được từ các giả thiết thì quay lui sang các luật sản xuất khác sinh ra b có dạng $b_1\wedge....\wedge b_m \Rightarrow$ b. Ngược lại, nếu mọi a_i đều dẫn xuất được giả thiết thì quá trình dẫn xuất ra b là đúng)

- **Giải thuật**

GỌi T là tập các sự kiện cần chứng minh tại thời điểm đang xét (khởi tạo T= G, G là tập kết luận).

$S(p) = \{r_i \in R \ / \ right(r_i) = p\}$ (là tập các luật trong R sao cho vế phải chứa p)

Procedure suydienlui (g);

Begin

T:= {g};

If $T \subset F$ then write (‘g đã được chứng minh ‘)

Else

Begin

p:=get(T);

```
If S(p) = {} then write ('g không chứng minh được ')
    Else
        For r_i ∈ S(p) do
            Begin
                T:= T \ right(r_i);
                T:= T + left(r_i);
                For l∈T \ F do          suydienlui(l);
            End;
End;
```

Ví dụ

1) Cho tập sự kiện F={p, r}, và tập luật R:

r_1) $p \Rightarrow q$

r_2) $q \wedge r \Rightarrow s$

Chứng minh s

p	r	T	S(p)
		s	
s	r_2	q, r	r_2
q	r_1	r, p	r

Nhận xét

- **Suy diễn tiến:**

  Ưu điểm:

i) Làm việc tốt khi bài toán có bản chất là đi thu thập thông tin rồi thấy điều cần suy diễn

ii) Cho ra khối lượng lớn các thông tin từ một số thông tin ban đầu. Nó sinh ra nhiều thông tin mới.

iii) Suy diễn tiến là tiếp cận lý tưởng đối với các loại bài toán cần giải quyết các nhiệm vụ như lập kế hoạch, điều hành, điều khiển và diễn dịch.

Nhược điểm:

i) Không cảm nhận được rằng chỉ cần một vài thông tin quan trọng. Hệ thống hỏi các câu hỏi có thể hỏi mà không biết rằng chỉ một ít câu đã đi đến kết luận được.

ii) Hệ thống có thể hỏi cả câu hỏi không liên quan. Có thể các câu trả lời cũng quan trọng nhưng làm người dùng lúng túng khi phải trả lời các câu chẳng dính đến chủ đề.

- **Suy diễn lùi:**

Ưu điểm:

i) Phù hợp với bài toán đưa ra giả thuyết và liệu giả thuyết đó có đúng hay không?

ii) Tập trung vào đích đã cho. Nó tạo ra một loạt câu hỏi chỉ liên quan đến vấn đề đang xét, thuận tiện đối với người dùng.

iii) Khi suy diễn một điều gì từ thông tin đã biết , nó chỉ tìm trên một phần của cơ sở tri thức thích đáng đối với bài toán đang xét.

iv) Suy diễn lùi được đánh giá cao trong các bài toán như là chẩn đoán, dự đoán và tìm lỗi.

Nhược điểm:

Nhược điểm cơ bản của loại suy diễn này là nó thường tiếp theo dòng suy diễn thay vì đúng ra phải dừng ở đó mà sang nhánh khác.

- **Như vậy,** dựa vào các ưu và nhược điểm của từng loại suy diễn mà ta nên chọn kỹ thuật suy diễn nào để áp dụng vào bài toán. Trước tiên, ta xem xét các chuyên gia giải nó như thế nào?. Nếu cần thu thập dữ liệu rồi mới quyết định suy diễn cái gì thì ta chọn suy diễn tiến. còn nếu đã có giử thuyết và cần chứng minh cái đích này thì ta dùng suy diễn lùi.

Ví dụ Một bác sĩ có thể hiểu hàng trăm vấn đề có thể xảy ra với một cá nhân, nhưng vẫn phải tìm hiểu hiện trạng của bệnh nhân, lúc đó cần suy diễn tiến. Ngược lại bác sĩ hầu như thấy được bệnh (ví dụ như viêm họng) thì ông ta dùng suy diễn lùi.

Bài tập 1. Cho các biểu thức logic mệnh đề đúng sau:

1) ac

2) $ab \rightarrow f$

3) $(d + b)f \rightarrow i$

4) $\neg h + \neg a + f$

5) $fgh \rightarrow i$

6) $(\neg a + d + \neg c)$

7) $ad \rightarrow gh$

Chứng minh hoặc bác bỏ mệnh đề i bằng phương pháp suy diễn tiến và suy diễn lùi

LỜI GIẢI

- Biểu diễn các biểu thức đúng đã cho bằng luật sản xuất (xác định tập luật, tập sự kiện ban đầu, tập sự kiện cần chứng minh)

Quá trình biến đổi

3) $(d+b)f \rightarrow i \equiv \neg((d+b)f)+i \equiv \neg(d+b)+\neg f+i \equiv (\neg d\neg b)+\neg f+i \equiv$

$(\neg d+\neg f+i)(\neg b+\neg f+i) \equiv (df \rightarrow i)(bf \rightarrow i)$

4) $\neg h + \neg a + f \equiv \neg(ha)+f \equiv ha \rightarrow f$

6) $(\neg a + d + \neg c) \equiv \neg(ac)+d \equiv ac \rightarrow d$

7) $ad \rightarrow gh) \equiv \neg(ad)+(gh)) \equiv (\neg(ad)+g)(\neg(ad)+h) \equiv (ad \rightarrow g)(ad \rightarrow h)$

TẬP SỰ KIỆN F={A, C}, TẬP SỰ KIỆN CẦN CHỨNG MINH G={I}

Tập luật R:

r_1) $ab \rightarrow f$

r_2) $(df \rightarrow i)$

r_3) $(bf \rightarrow i)$

r_5) $fgh \rightarrow i$

r_6) $ac \rightarrow d$

r_7) $ad \rightarrow g$

r_4) ha $\rightarrow$ f r_8) ad $\rightarrow$ h

- **Suy diễn tiến (tiến hành lập bảng sau)**

r	T	S	R
	a, c	r_6	r_1, r_2, r_3, r_4, r_5, r_6, r_7,
r_6	a, c, d	r_7, r_8	r_8
r_7	a, c, d, g	r_8	$r_{1,...}r_5$, r_7, r_8
r_8	a, c, d, g, h	r_4	$r_{1,...}r_5$, r_8
r_4	a, c, d, g, h, f	r_2, r_5	$r_{1,...}r_5$
r_2	a, c, d, g, h, f, i		r_1, r_2, r_3,r_5

(trong đó: r: là luật đang xét, T: tập sự kiện đúng tại thời điểm đang xét, S: tập các luật có dạng các mệnh đề ở vế trái thuộc T. R là tập luật tại thời điểm đang xét)

Vì i$\in$T (là tập sự kiện đúng). Vậy i được chứng minh

- **Suy diễn lùi (tiến hành lập bảng sau)**

p	r	T	S(p)
		i	
i	r_2	d, f	r_2, r_3, r_5
f	r_1	d, b	r_1, r_2
b		d	$\varnothing$
quay lui			
f	r_2	d, h	r_2
h	r_8	d	r_8
d	r_6	$\varnothing$	r_6

Vậy i được chứng minh

Bài tập 2. Cho cơ sở tri thức được biểu diễn bằng các biểu thức logic đúng sau

1) $pt \rightarrow a$
2) $qt \rightarrow s$
3) $pq \rightarrow b$
4) $\neg b \rightarrow st$
5) $p \rightarrow t$
6) $apq \rightarrow c$
7) $bc \rightarrow t$
8) pq

Biểu diễn tri thức đã cho dưới dạng luật sản xuất và dùng phương pháp suy diễn tiến và suy diễn lùi để chứng minh hoặc bác bỏ sự kiện $s \equiv 1$.

Bài tập 3. Cho cơ sở tri thức được biểu diễn bằng các biểu thức logic đúng sau

1) $(a+c)b \rightarrow f$
2) $\neg e + \neg f + a$
3) $gfh \rightarrow i$
4) $(e+ f)b \rightarrow gi$
5) $(\neg a+ e + \neg c)abc$

Dùng phương pháp suy diễn tiến và suy diễn lùi để chứng minh hoặc bác bỏ sự kiện $i \equiv 1$.

Bài tập 4.. Cho cơ sở tri thức được biểu diễn bằng các biểu thức logic đúng sau

1) efh
2) $\neg a + g + d$
3) $\neg h + c + d$
4) $af \rightarrow bg$
5) $ke \rightarrow d$
6) $(ef \rightarrow a)(\neg c+ \neg e + \neg f)$

- Biểu diễn tri thức đã cho dưới dạng luật sản xuất
- Dùng phương pháp suy diễn tiến để chứng minh sự kiện $d \equiv 1$ đúng. Cho biết các luật dư thừa trong vết suy diễn

Bài tập 5.. Trong một lớp học, có một nhóm học sinh gồm 10 bạn có tên lần lượt là: A, B, C, D, E, F, G, H, I và J. Giữa các bạn học sinh đó có mối quan hệ gọi là quan hệ ảnh hưởng. Ví dụ: nếu ta viết AB>C thì có nghĩa là hai bạn đồng thời cùng thuyết phục bạn C tham gia một hoạt động nào đó. Giả sử ban đầu có bốn bạn E, F, H, I tham gia dự thi sản phẩm phần mềm do nhà trường tổ chức và ta cũng biết được rằng:

1) ACH>B
2) BH>ACD
3) ABCI>BDI
4) ADEI>BCG
5) CGI>AJE
6) H>BC

Hãy dùng phương pháp suy diễn tiến để chứng minh rằng cả 10 bạn trong nhóm trên đều tham gia dự thi sản phẩm phần mềm.

TÀI LIỆU THAM KHẢO

1. Bạch Hưng Khang, Hoàng Kiếm
   Trí tuệ nhân tạo: Các phương pháp và ứng dụng. Nhà xuất bản Khoa học và Kỹ thuật, 1989.

2. Đinh Mạnh Tường
   Giáo trình Trí tuệ nhân tạo, Đại học Quốc gia Hà nội.

3. Nguyễn Thanh Thuỷ
   Trí tuệ nhân tạo: Các phương pháp giải quyết vấn đề và kỹ thuật xử lý tri thức. Nhà xuất bản Giáo dục, 1996.

4. N. Nilson
   Artificial Intelligence. Ed. McGrawhill, 1971

5. Patrick Henry Winston
   Artificial Intelligence. Ed. Addison Wesley, 1992.

.