ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

TRẦN THỊ DƯƠNG

VỀ CÁC BÀI TOÁN NP-C VÀ MỘT SỐ PHƯƠNG PHÁP GIẢI

Chuyên ngành : Khoa học máy tính
Mã số : 60480101

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

Thái Nguyên - 2014

MỞ ĐẦU

Ngày nay, cùng với sự phát triển mạnh mẽ của khoa học và công nghệ, đặc biệt là máy tính, người ta có khả năng giải quyết được nhiều bài toán rất phức tạp.

Tuy nhiên, còn những vấn đề là "không giải được" cho dù kỹ thuật máy tính có phát triển và cũng có những vấn đề được xem là "quá phức tạp", vượt mọi khả năng tính toán thực tế vì mất quá nhiều thời gian. Việc nghiên cứu về độ phức tạp của thuật toán đã cho phép chúng ta phân loại được các lớp bài toán theo từng mức độ phức tạp khác nhau, và chỉ ra ranh giới giữa các lớp bài toán giải được và những lớp bài toán không thể giải được trong thời gian đa thức.

Trong lý thuyết độ phức tạp tính toán, lớp NP - C (NP - đầy đủ) là một lớp các bài toán quyết định. Một bài toán L là NP - C nếu nó nằm trong lớp NP (lời giải cho L có thể được kiểm chứng trong thời gian đa thức) và là NP - Hard (mọi bài toán trong NP đều có thể quy về L trong thời gian đa thức).

Mặc dù bất kì lời giải nào cho mỗi bài toán đều có thể được kiểm chứng nhanh chóng, hiện chưa có cách nào tìm ra được lời giải đó một cách hiệu quả. Thời gian thực thi của tất cả các thuật toán hiện tại cho những bài toán này đều tăng rất nhanh theo kích thước bài toán. Vì vậy ngay cả những trường hợp có kích thước tương đối lớn đã đòi hỏi thời gian hàng tỷ năm để giải.

Các bài toán lớp NP - C là tập hợp các bài toán NP - Hard trong NP. Các bài toán lớp NP - C được quan tâm nghiên cứu bởi khả năng kiểm chứng nhanh chóng lời giải (NP) dường như có liên hệ với khả năng tìm kiếm nhanh chóng lời giải (P). Hiện vẫn chưa biết được nếu mọi bài toán trong NP đều có thể được giải quyết nhanh chóng hay không (đây chính là bài toán P so với

NP). Tuy nhiên, nếu bất kì một bài toán nào trong NP - C có thể được giải quyết nhanh chóng, thì theo định nghĩa của NP - C, mọi bài toán trong NP đều có thể được giải quyết nhanh chóng.

Các bài toán NP- C xuất hiện thường xuyên trong thực tế nên, mặc dù chưa có giải thuật trong thời gian đa thức cho chúng, các nhà nghiên cứu vẫn tìm cách giải quyết chúng thông qua các phương pháp khác như thuật toán xấp xỉ, thuật toán gần đúng nhân tử hóa, v.v... Việc tìm hiểu và nghiên cứu các phương pháp giải bài toán lớp NP- C là một trong những bài toán mở của khoa học máy tính hiện nay. Vì vậy em đã chọn đề tài: ***Về các bài toán NP-C và một số phương pháp giải*** để làm luận văn tốt nghiệp.

Cấu trúc của luận văn gồm: Phần mở đầu, phần kết luận và 3 chương nội dung, cụ thể:

Chương 1: Khái niệm các lớp bài toán P, NP, NP-C.

Trong chương này em giới thiệu chung về các lớp bài toán P, NP, NP – C, minh họa bằng các ví dụ cụ thể và đưa ra mối quan hệ giữa lớp P, NP và NP-C

Chương 2: Phương pháp tham và phương pháp nhánh cận giải một số bài toán NP-C

Trong chương này em trình bày phương pháp tham giải bài toán về đồ thị và phương pháp nhánh cận giải bài toán Ba lô, bài toán tìm đường đi ngắn nhất.

Chương 3: Chương trình thử nghiệm

Chương này thể hiện chương trình cài đặt thuật toán nhánh cận giải bài toán Ba lô.

Chương 1: KHÁI NIỆM CÁC LỚP BÀI TOÁN P, NP, NP – C

1.1. Vài khái niệm cơ bản của lý thuyết độ phức tạp

1.1.1 Máy Turing tất định và không tất định

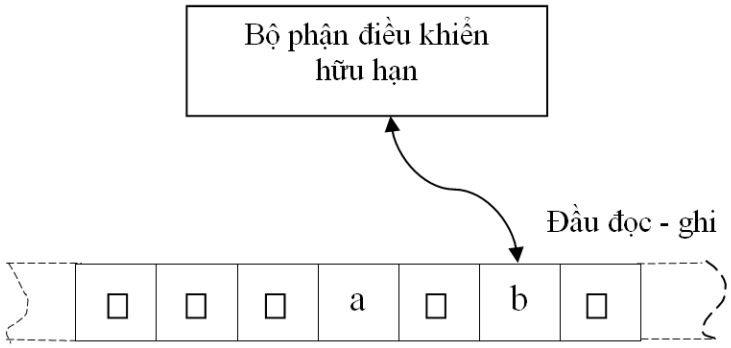
Máy Turing là một máy tính trừu tượng mô tả các quá trình tính toán trên máy tính.

Máy Turing có hai loại: Máy Turing tất định (Deterministic Turing Machine) và Máy Turing không tất định (Nondeterministic Turing Machine) được mô tả như sau:

Máy Turing tất định

Mô tả cách làm việc của máy:

Máy Turing tất định gồm một bộ điều khiển hữu hạn trạng thái, một đầu đọc và ghi, một băng vô hạn được chia thành từng ô vuông, mỗi ô có thể lưu giữ một ký hiệu thuộc tập hữu hạn các ký hiệu.



Hình 1.1. Mô tả máy tính Turing tất định

Khởi đầu, một xâu Input được đặt trên một băng, đó là chuỗi ký hiệu có chiều dài hữu hạn được chọn từ một bộ chữ cái. Những ô còn lại của băng vô hạn theo cả hai bên phải và trái, chứa một ký hiệu đặc biệt là ký hiệu trống (diễn tả trạng thái ô không có ký hiệu nào).

Có một đầu đọc – ghi luôn chỉ vào một trong các ô của băng. Ta nói rằng máy Turing đang đọc – ghi ô đó. Lúc khởi hoạt, đầu đọc – ghi nằm

tận bên trái của xâu Input.

Một bước hoạt động của máy Turing được quy định bởi một hàm phụ thuộc trạng thái của bộ điều khiển và ký hiệu đang được đọc chuyển vị. Trong một bước hoạt động, máy Turing sẽ:

- Thay đổi trạng thái. Trạng thái tiếp theo cũng chính là trạng thái hiện tại.

- Ghi một ký hiệu băng vào ô đang được quét. Ký hiệu băng này thay thế ký hiệu băng đang có ở ô vuông đó. Ký hiệu được ghi cũng có thể chính là ký hiệu hiện đang ở đó.

- Di chuyển đầu đọc – ghi sang trái hoặc sang phải.

Một cách không hình thức, ta có thể định nghĩa như sau:

Định nghĩa 1.1 Một máy Turing M tất định là một bộ M = (Q, Σ, Γ, δ, $q_{0,}$ B, F)

Trong đó các thành phần của M có ý nghĩa như sau:

Q: là tập hữu hạn các trạng thái của bộ điều khiển hữu hạn.

Σ: Tập hữu hạn các chữ cái.

Γ: Tập đầy đủ các kí hiệu băng; Σ luôn là tập con của Γ

δ: Hàm chuyển vị, δ: Γ ×Q x {1, 0, 1}.Đối của δ(q, X) là một trạng thái q và một kí hiệu băng X. Giá trị của (q, X) nếu được định nghĩa sẽ là một bộ ba (p, Y, D) trong đó:

p: trạng thái tiếp theo

Y: Một ký hiệu thuộc Γ và sẽ được ghi vào ô đang được quét, thay thế cho ký hiệu đang ở trong ô đó.

D: Một trong hai ký hiệu L (sang trái) hoặc R (sang phải) để chỉ ra hướng di chuyển của đầu đọc – ghi.

$q_{0:}$ Trạng thái bắt đầu, một phần tử Q là trạng thái khởi đầu của bộ điều khiển.

B: ký tự trắng (blank). B ∈ Γ

F: Tập kiểm hợp các trạng thái kết thúc, một tập con của Q.

- ***Các chức năng và đặc trưng cơ bản của máy Turing tất định:***

Ngôn ngữ xác định bởi máy Turing và ngôn ngữ đoán nhận được

Cho M = (Q, Σ, Γ , δ, q_0, B, F) là một máy tính Turing.

Một hình trạng của máy tính Turing M là một từ có dạng aqb, trong đó a ∈ Γ^*, b ∈ Γ^*, biểu thị nội dung: trên băng có từ ab, đầu đọc - ghi nhìn kí tự đầu b và máy ở trạng thái q. Hàm chuyển δ dho ta quy tắc chuyển đổi các hình trạng. Nếu máy tính Turing M bắt đầu làm việc với hình trạng q_0w (trong đó w ∈ Z^*) và chuyển đổi liên tiếp sau một số hữu hạn bước đến hình trạng kết thúc aFb ở trạng thái chấp nhận F, thì ta nói rằng máy tính Turing M chấp nhận từ vào w.

Ta ký hiệu L_M = {w\ w ∈ Z^*, M chấp nhận w).

Định nghĩa 1.2. Ngôn ngữ L_M gọi là ngôn ngữ xác định bởi máy Turing hay còn gọi là ngôn ngữ tương ứng với của máy tính Turing M.

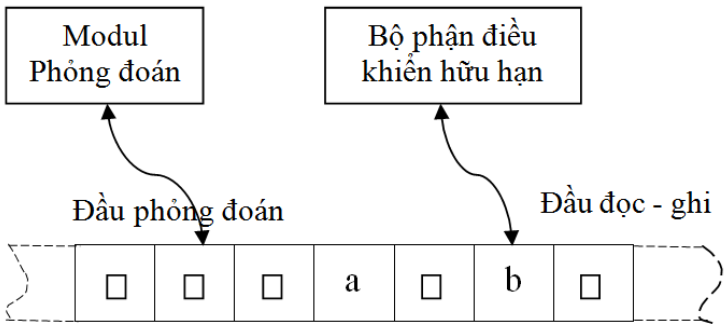
Định nghĩa 1.3. Cho L là một ngôn ngữ trên bảng chữ cái Σ

Ngôn ngữ L gọi là đoán nhận được bởi máy tính Turing nếu tồn tại M sao cho L_M = L (tức là tồn tại một máy tính Turing sao cho ngôn ngữ tương ứng của nó trùng với một ngôn ngữ cho trước L).

Ta nói máy tính Turing đoán nhận ngôn ngữ L.

Máy tính Turing không tất định

Máy tính Turing tất định là một dạng đặc biệt của máy tính Turing không tất định.



Hình 1.2. Mô tả máy tính Turing không tất định

Nhưng máy tính Turing không tất định có một hàm chuyển vị δ sao cho mỗi trạng thái q và ký hiệu đọc được δ(q, X) là một bộ ba $\{(q_1, Y_1, D_1), (q_2, Y_2, D_2), \ldots (q_k, Y_k, D_k)\}$, trong đó k là một số nguyên hữu hạn nào đó. Tại mỗi bước máy tính Turing không tất định có thể chọn một trong các bộ ba để thực hiện bước chuyển tiếp theo. Tuy nhiên nó không thể lấy một trạng thái từ một trong các bộ ba này, một ký hiệu băng từ bộ ba khác và một hướng lại từ bộ ba khác nữa.

So sánh hai định nghĩa trên ta thấy máy tính Turing không tất định được định nghĩa một cách hình thức giống máy tính Turing tất định có thêm môđun phỏng đoán, nhằm để có thể chọn bước thực thi kế tiếp tùy ý trong một tập cho trước các lệnh và có khả năng xử lý song song các phỏng đoán.

1.1.2 Bài toán quyết định và ngôn ngữ tương ứng

Định nghĩa 1.4 Cho một tập các dữ kiện (instance) và câu hỏi (question) trên các dữ kiện thuộc tập đó. Bài toán quyết định là bài toán mà câu trả lời của nó là "Yes" hay "No" (tương ứng với True/1 hay False/0)

Sau đây là vài ví dụ minh họa cho bài toán quyết định:

Ví dụ 1.1*: *Bài toán kiểm tra số nguyên tố

Instance: một số nguyên n > 2

Question: n có phải số nguyên tố hay không?

Ví dụ 1.2: *Bài toán HC (Hamilton Cycle)*

Instance: đồ thị vô hướng G = (V, E)

Question: đồ thị vô hướng G = (V, E) có chu trình Hamilton hay không?

Về nguyên tắc mọi bài toán đều có thể biểu diễn lại dưới bài toán quyết định tương ứng.

Ngôn ngữ tương ứng với bài toán quyết định:

Giả sử cho một bài toán quyết định π với tập các dữ kiện I được biểu diễn bởi các xâu trên bảng chữ cái Σ nào đó, và với question Q trên tập

I. Ký hiệu L(π) là tập các xâu (thuộc Σ^*) biểu diễn các dữ kiện mà câu hỏi Q có trả lời "đúng". Khi đó ta nói ngôn ngữ L(π) là ngôn ngữ tương ứng với bài toán π.

2.3.1 Thời gian tính của một máy tính Turing

Cho trước một bài toán quyết định π với tập các dữ kiện I được biểu diễn bởi các xâu trên bảng chữ cái Σ nào đó, với câu hỏi Q trên tập I. Ký hiệu L(π) là tập các xâu (thuộc Z^*) biểu diễn các dữ kiện của một instance cụ thể của bài toán π. Ta biết rằng ngôn ngữ L(π) là một ngôn ngữ tương ứng với bài toán π. Với mỗi instance I cụ thể, ta sẽ có một input biểu diễn nó, mà ta ký hiệu là x(*I*). Bây giờ ta có thể biểu diễn thời gian tính của bài toán π đối với một máy tính Turing cho trước. Với một input x(*I*) $\in$ L(π), máy tính sẽ chạy cho đến lúc dừng tại trạng thái "Yes/No". Thời gian tính x(*I*) sẽ là số bước đoán nhận xâu x*(I)* của máy cho tới khi máy dừng lại. Thông thường số bước chạy máy này phụ thuộc vào *I*, và tất nhiên là một hàm số của độ dài biểu diễn *I*, tức là độ dài của xâu x(*I*).

Bằng cách đó ta định nghĩa: $T_M(n)$:= max{m: tồn tại một xâu x $\in$ Z^* với |x| = n mà thời gian đoán nhận xâu là m}

Một máy tính Turing (hay một chương trình tính toán trên cơ sở máy tính Turing) được nói là có thời gian tính toán đa thức (gọi tắt là thời gian đa thức) nếu như tồn tại một đa thức p(*n*) sao cho mọi số tự nhiên n ta có $T_M(n) \leq p(n)$. Khi đó ta cũng nói rằng chương trình máy tính Turing có độ phức tạp tính toán không vượt quá p(*n*).

1.1.4. Lớp P, NP và mối quan hệ giữa lớp P và lớp NP

Định nghĩa l.5 (Lớp P - Polynomial time)

Ta gọi lớp P là lớp những bài toán quyết định giải được bằng máy tính Turing tất định trong thời gian đa thức.

Một bài toán quyết định π là giải được trong thời gian đa thức, nếu ngôn ngữ L(π) tương ứng với nó thuộc lớp P, tức nó đoán nhận được trong thời gian đa thức.

Như vậy, lớp P gần như tương ứng với lớp các bài toán quyết định giải được trong thời gian đa thức, về mặt lý thuyết, có thể xem là lớp các bài toán dễ.

Ví dụ 1.3: *Bài toán kiểm tra số nguyên tố*

Instance: một số nguyên n > 2

Question: n có phải là số nguyên tố hay không?

Ví dụ 1.4: *Thuật toán Kruskal tìm cây khung bé nhất của một đồ thị có m nút và e cạnh.*

Instance: một đồ thị có *m* nút và *e* cạnh

Question: tìm cây khung bé nhất?

Định nghĩa l.6 (Lớp NP - Nondeterministic Polynomial)

Ta gọi lớp NP là lớp các bài toán quyết định có thể giải được bằng máy tính Turing không tất định trong khoảng thời gian đa thức.

Một cách không hình thức, chúng ta nói một ngôn ngữ L thuộc lớp NP nếu có một máy tính Turing không tất định và một độ phức tạp thời gian T(n) sao cho L = L_M và khi M được cho nguyên liệu có chiều dài n thì không có dãy bước chuyển nào của M vượt quá T(n) bước chuyển.

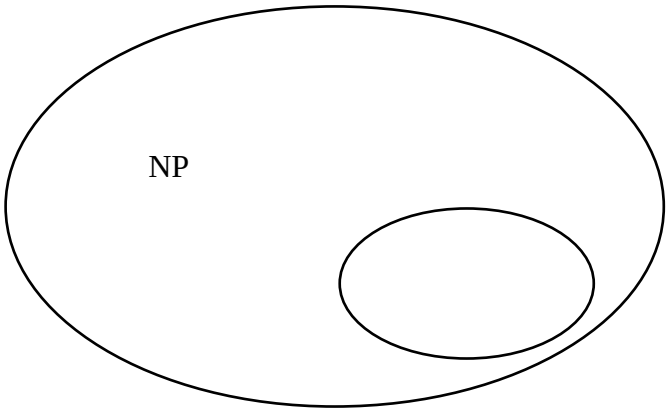
Ví dụ 1.5: *Bài toán chu trình Hamilton*

Instance: đồ thị vô hướng G = (V, E)

Question: đồ thị vô hướng G = (V, E) có chu trình Hamilton hay không?

Để nói về mối quan hệ giữa lớp P và lớp NP ta thấy do máy tính Turing tất định là trường hợp đặc biệt của máy tính Turing không tất định nên các bài toán thuộc lớp P sẽ thuộc lớp NP.

Tuy P $\in$ NP là rất hiển nhiên song ta vẫn chưa biết P = NP hay không, nhưng hầu hết các nhà nghiên cứu đều tin rằng P $\neq$ NP. Từ đó ta có mô hình mô phỏng sau:



Hình 1.3 Mối quan hệ giữa lớp P và NP

1.1.5 Phép dẫn với thời gian đa thức (Polynomial Time Reduction)

Giả sử ta muốn giải quyết bài toán π_1 (trong đó π_1: I_1 —> {Yes/ No}) mà ta có thuật toán cho bài toán π_2 (trong đó π_2: I_2 —> {Yes/ No}, giả sử ta có hàm f: I_1 —> I_2 mà dữ kiện x của π_1 sinh ra dữ kiện f(x) cho π_2 sao cho câu trả lời đúng cho π_1 trên x là "Yes" nếu và chỉ nếu câu trả lời đúng cho π_2 trên f(x) là "Yes" và ngược lại, thì ta có thể sử dụng thuật toán cho π_2 để giải quyết bài toán π_1.

Định nghĩa 1.7

Cho π_1 và π_2 là hai bài toán quyết định, π_i (Y) là lớp các Instance ứng với YES, π_i (N) là lớp các Instance ứng với No.