**Câu 1. Trình bày nội dung và mối quan hệ mật thiết của ba lĩnh vực nghiên cứu của Lý thuyết tính toán. (cuối trang 20 – đầu trang 22)**

\*) Nội dung của ba lĩnh vực nghiên cứu của Lý thuyết tính toán:

* Lý thuyết otomat: đề cập đến việc xây dựng các mô hình đoán học về tính toán
* Lý thuyết về khả năng tính toán: có mục tiêu là phân chia các bài toán thành lớp các bài toán giải được và lớp các bài toán không giải được
* Lý thuyết độ phức tạp tính toán: phân chia các bài toán giải được thành các lớp khác nhau theo mức độ khó khăn khi giải chúng

\*) Mối quan hệ mật thiết của ba lĩnh vực nghiên cứu của Lý thuyết tính toán:

Ba lĩnh vực nghiên cứu của lý thuyết tính toán tuy có nội dung nghiên cứu riêng rẽ, nhưng chúng có quan hệ mật thiết với nhau. Để hình dung được sự ảnh hưởng qua lại giữa ba lĩnh vực trong sự phát triển chung của lý thuyết tính toán, chúng ta hãy quan sát vai trò của mỗi lĩnh vực được thể hiện như thế nào trong quá trình giải quyết một bài toán:

* Về lý thuyết, quá trình giải bài toán bao gồm việc lập mô hình toán học cho bài toán khi cần thiết và dựa trên mô hình đó xây dựng phương pháp giải (thuật toán giải). Tuy nhiên để bài toán giải được một cách thực tế, việc lập mô hình toán học cũng như việc xây dựng thuật toán giải đều phải thích hợp với trang thiết bị tính toán hiện có. Những trang thiết bị này có được là nhờ các thành tựu của công nghệ mang lại,mà cơ sở lý luận dựa trên những kết quả nghiên cứu của lý thuyết otomat
* Đôi khi trong quá trình giải bài toán, chúng ta không thể tìm được một thuật toán giải nó do bản chất phức tạp của bài toán. Việc chứng tỏ không có thuật toán giải bài toán,là phận sự của lý thuyết về khả năng tính toán. Để làm được điều đó, cần thiết phải có một định nghĩa chính xác về thuật toán thay thế cho khái niệm thuật toán theo nghĩa trực giá mà ta vẫn thường dùng khi xây dựng thuật toán giải một bài toán nào đó.
* Mặt khác, ngay cả khi ta đã xây dựng được thuật toán giải bài toán, nhưng trên thực tế đôi khi để nhận được một lời giải thỏa đáng lại rất gian nan, dù ta được cung cấp đầy đủ thiết bị tiên tiến nhất. Lý giải sự khác biệt này là việc làm của lý thuyết độ phức tạp tính toán

## **Câu 2. Trình bày sự ra đời và phát triển của Lý thuyết độ phức tạp tính toán. (cuối trang 22 – đầu trang 24)**

\*) Sự ra đời và phát triển của Lý thuyết độ phức tạp tính toán:

- Những năm 40 và 50 của thế kỉ XX, với sự ra đời của máy tính thực tế tưởng rằng việc thực hiện thuật toán để giải tiếp bài toán là một việc đơn giản. Song,thực tế lại không lúc nào cũng như mong muốn. Chẳng hạn, khi thực hiện thuật toán trên máy tính, nhiều bài toán giải được một cách dễ dàng và bên cạnh đó cũng có không ít bài toán khó giải thậm chí không thể giải được, mặc dù về mặt lý thuyết chúng hoàn toàn giải được bằng nhiều thuật toán khá nhau.Ngoài ra,một bài toán cũng có thể giải được bằng nhiều thuật toán xấu tốt khác nhau và ngay với một thuật toán thì có thể trong trường hợp này nó cho ta kết quả nhanh còn trong trường hợp khác lại cho kết quả chậm.Nguyên nhân ấy phải chăng là do bản chất phức tạp của bài toán,hay do thuật toán mà ta xây dựng chưa thật hiệu quả,…. Liên quan đến vấn đề này, vào cuối thập niên 60 của thế kỉ XX một lĩnh vực nghiên cứu được hình thành, đó là lý thuyết độ phức tạp tính toán.

- Ra đời và phát triển mạnh mẽ khoảng bốn mươi năm qua, lý thuyết độ phức tạp tính toán tuy vẫn chưa có được một lý giải thỏa đáng cho những hiện tượng phổ biến nêu trên,nhưng nó đã có một bước tiến đáng kể với nhiều kết quả phong phú có ý nghĩa nhất định về lý thuyết cũng như ứng dụng.

## **Câu 3. Trình bày các điểm cơ bản trong cách tiếp cận bài toán, có ví dụ minh họa (trang 25 – gần hết trang 26)**

- Các bài toán có thể được phát biểu chính xác bằng ngôn ngữ toán học, nhưng đôi khi cũng được phát biểu theo một ngôn ngữ tự nhiên dân dã. Trong trường hợp khi bài toán chưa được phát biểu bằng ngôn ngữ toán học thì việc đầu tiên phải làm là dịch bài toán đó sang một ngữ cảnh toán học thích hợp

- Trong việc hoàn tất quá trình giải,khâu cơ bản nhất là dựa trên mô hình toán học của bài toán, ta cần đề xuất một phương pháp giải, hay lý tưởng hơn là xây dựng một thủ tục chặt chẽ xử lý một cách hiệu quả các thông tin liên quan đến mỗi dữ kiện bài toán, để từ đó thu được nghiệm tương ứng với từng dữ kiện. Cuối cùng, cần tiến hành phân tích và đánh giá hiệu quả của cách giải bài toán cũng như khả năng hiện thực hóa trên những trang thiết bị tính toán hiện có.

Ví dụ minh họa:

Cho một lô hàng hóa gồm các gói hàng, mỗi gói đều có khối lượng cùng với giá trị cụ thể, và cho một chiếc balo. Hãy chọn từ balo này một gói hàng nào đó và xếp đầy balo,nhưng không được quá sao cho thu được giá trị lớn nhất có thể.

Đây là một bài toán tối ưu tổ hợp quen thuộc, được ký hiệu là MAX-KNAPSACK và được phát biểu bằng ngôn ngữ toán học dưới dạng tổng quát như sau: MAX-KNAPSACK

Dữ kiện: Cho hai dãy số nguyên dương , , …, và , , …, .

Yêu cầu: Tìm một tập con *I* {1,2,...,n} sao cho

## **Câu 4. Trình bày cách xác lập tương ứng bài toán tìm kiếm và bài toán quyết định, có ví dụ minh họa (trang 27 –trang 28)** - Một bài toán có thể được phát biểu thành hai phần tách biệt: phần dữ kiện và phần yêu cầu. Đối với phần dữ kiện ta cần xác định rõ tập dữ kiện của bài toán bao gồm những dữ kiện cụ thể nào, còn phần yêu cầu thường có hai loại. Loại thứ 1 là một câu hỏi mà đối với mỗi dữ kiện bài toán chỉ cần trả lời đơn giản là đúng hoặc sai. Bài toán với câu hỏi như vậy được gọi là bài toán quyết định. Loại thứ hai là yêu cầu tìm nghiệm đối với dữ kiện bất kỳ cho trước. Bài toán yêu cầu như vậy là bài toán tìm kiếm. Trong lớp các bài toán tìm kiếm các bài toán tối ưu có một vị trí quan trọng. Bài toán cực đại hóa và bài toán cực tiểu hóa là các bài toán tối ưu với yêu cầu tìm kiếm nghiệm chấp nhận được với giá trị lớn nhất và nhỏ nhất tương ứng.

Bài toán quyết định tương ứng với bài toán tìm kiếm có thể được xác định đơn giản bằng cách thay yêu cầu: “tìm nghiệm đối với dữ kiện bất kỳ cho trước ” bằng câu hỏi “phải chăng tồn tại nghiệm đối với mỗi dữ kiện đã cho” trong trường hợp này tập dữ kiện không thay đổi.

- Ví dụ minh họa:

bài toán “hãy tìm chu trình hamilton trong mỗi đồ thị cho trước”

Được tương ứng với bài toán quyết định sau đây:

HAMILTONIAN CYCLE

Dữ kiện: cho đồ thị G

Câu hỏi: Phải chăng trong G có chu trình hamilton

Tuy nhiên, các bài toán tối ưu được tương ứng với các bài toán quyết định thích hợp hơn bài toán quyết định tương ứng với bài toán cực đại hay cực tiểu hóa được xác định như sau: tập dữ kiện của nó được xây dựng bằng cách thêm vào mỗi dữ kiện của bài toán tối ưu một ranh giới B tùy ý, thuộc cùng miền xác định giá trị của nghiệm. Tiếp theo câu hỏi được phát biểu rằng, đối với mỗi dữ kiện như vậy, phải chăng có một nghiệm chấp nhận được với giá trị không nhỏ hơn B(không lớn hơn B, tương ứng)?

Thí dụ, bài toán cực đại hóa MAx- KNAPSACK được tương ứng với bài toán quyết định sau đây:

KNAPSACK

Dữ kiện: Cho hai dãy số nguyên dương , , …, và , , …, ,B.

Câu hỏi: Phải chăng có một tập con *I* {1,2,...,n} sao cho

## **Câu 5. Trình bày về ngôn ngữ biểu diễn bài toán quyết định, ví dụ dùng xâu liên kết để biểu diễn cho đồ thị có hướng nào đó (trang 29 – đầu trang 31)**

\***Tính xúc tích**: Theo lẽ tự nhiên, một bài toán quyết định cũng thường được diễn tả bằng nhiều ngôn ngữ hình thức khác nhau theo những phép mã hóa khác nhau đối với các dữ kiện bài toán. Song, để đảm bảo yêu cầu tối thiểu về tính súc tích đối với ngôn ngữ, việc mã hóa cần thỏa mãn các tiêu chuẩn sau đây:

* Từ mã (xâu biểu diễn) của mỗi dữ kiện bài toán phải ngắn gọn và không được “độn thêm" những thông tin không cần thiết
* Các số tham gia dữ kiện bài toán cần được biểu diễn dưới dạng nhị phân hoặc theo một cơ số nào đó lớn hơn 1 (bởi vì dạng biểu diễn theo cơ số 1 dài hơn cỡ hàm mũ so với bất kỳ dạng biểu diễn nào theo cơ số k với k 2

\***Sơ đồ mã hóa chuẩn.** Đối với từng bài toán cụ thể, ta dễ dàng xây dựng được một phép mã hóa thích hợp. Tuy nhiên, việc đưa ra một định nghĩa hình thức về phép mã hóa là không hề đơn giản. Song, về mặt nguyên tắc, các phép mã hóa thích hợp có thể được xây dựng dựa trên một ***sơ đồ mã hóa chuẩn,*** mà theo đó các dữ kiện của bài toán được biểu diễn bởi các “xâu liên kết", tức “xâu có cấu trúc", trên bảng chữ chứa **ψ** = {0, 1, -, [, ], (, ), ,} với ký tự cuối cùng là dấu phẩy “,”. ***Xâu liên kết*** được định nghĩa một cách đề quy như sau:

* Biểu diễn nhị phân của một số nguyên k (một xâu bao gồm các ký tự 0 và 1, có ký tự “-” ở phía trước nếu k là số âm) là một xâu liên kết biểu thị số nguyên k.
* Nếu x là một xâu liên kết biểu thị số nguyên k, thì [x] là một xâu liên kết biểu thị “tên" của đối tượng mang số hiệu k.
* Nếu , , …, là các xâu liên kết biểu thị các đối tượng , , …, , thì (, , …, ) là một xâu liên kết biểu thị dãy (, , …, ).

Dựa theo sơ đồ mã hoá này, ta có thể biểu diễn các dữ kiện bài toán bởi các xâu liên kết trên một bảng chữ nào đó chứa các ký tự thuộc bảng chữ **ψ kể trên.** Dĩ nhiên, đây là một sơ đồ nên nó chỉ mang tính nguyên tắc. Để đáp ứng yêu cầu về tính súc tích của ngôn ngữ, trong từng trường hợp cụ thể ta có thể bổ sung những quy tắc thích hợp cần thiết nhưng vẫn phải đảm bảo tính chuẩn xác của việc biểu diễn.

Dưới đây là những biểu diễn của một vài đối tượng quen thuốc như: tập hợp, hàm, số hữu tỉ, và đồ thị. Ta sẽ dùng ký hiệu để chỉ ***biểu diễn nhị phân*** của số nguyên dương k

Để biểu diễn một tập hợp hữu hạn X, đầu tiên ta sắp xếp các phần tử của nó theo trật tự như một dãy X={, , …, ), trong trường hợp chúng chưa có một trật tự như vậy. Khi đó tập X được biểu diễn bởi một xâu liên kết (, , …, ), trong đó xi là xâu liên kết biểu diễn phần tử tương ứng của dãy, 1 i n.

*Một hàm* f: X ->Y được biểu diễn bởi một xâu liên kết ((, )(, )...(, )), trong đó xi là xâu liên kết biểu diễn phần tử ∈X và là xâu liên kết biểu diễn phần tử f() ∈Y, i i n.

*Một số hữu tỉ q = ,*  trong đó m và n là các số tự nhiên nguyên tố cùng nhau, được biểu diễn bởi xâu liên kết (,), trong đó và là biểu diễn nhị phân của các số m, n tương ứng.

*một đồ thị có hướng G = (V, A)* được biểu diễn bởi một xâu liên kết (x, y), ta ký hiệu là <G> = (x, y), trong đó x là xâu liên kết biểu diễn tập đỉnh V và y là xâu liên kết biểu diễn tập cung A của đồ thị (mỗi phần tử của A là một cặp các đỉnh thuộc V). Cụ thể, với V = {, , …, } và A = {, , …, ), trong đó cùng = (, ), 1 j m, theo sơ đồ mã hóa chuẩn nêu trên ta có thể biểu diễn đồ thị G như sau:

<G> =

Tuy nhiên, để đáp ứng yêu cầu về tính súc tích của ngôn ngữ, đồ thị G có thể được biểu diễn bởi một xâu đơn giản hơn

<G> =

trên bảng chữ gồm sáu ký tự {0, 1, (, ), #, ,}, trong đó xâu trước ký tự # biểu diễn tập đỉnh và sâu sau # biểu diễn các cung của đồ thị. Thí dụ, đồ thị hình ngôi sao năng cánh G1 với tập đỉnh {, , } được biểu diễn bởi xâu liên kết:

<> =

## **Câu 6. Trình bày Ngôn ngữ đặc trưng của bài toán quyết định và Mô hình tính toán đoán nhận ngôn ngữ (trang 31 – đầu trang 32).**

**Ngôn ngữ đặc trưng của bài toán quyết định**

Giả sử II là một bài toán quyết định với tập dữ kiện và câu hỏi trên mỗi dữ kiện của bài toán. Khi đó xác định một tính chất đối với mọi dữ kiện của bài toán, tức một hàm từ tập đến tập các giá trị chân lý {đúng, sai} như sau: đối với mỗi dữ kiện d ∈ ta có = “Đúng” khi câu hỏi trên dữ liệu d được trả lời là “đúng” trong trường hợp ngược lại = Sai

Giả sử e là một phép mã hóa thích hợp nào đó đối với bài toán II, mà theo đó mỗi dữ kiện bài toán được biểu diễn bởi một xâu liên kết trên bảng chữ Σ. như vậy, e ánh xạ các dữ kiện bài toán thành các xâu thuộcĐể đơn giản khi không cần lưu ý đến phép mã hóa e ta ký hiệu <d> = e(d) với mỗi d ∈

Ta định nghĩa ngôn ngữ sau:

Rõ ràng ngôn ngữ L(II) diễn đạt nội dung của bài toán II, nên được gọi là ngôn ngữ đặc trưng của II hay nôn ngữ tương đương với II, và được ký hiệu ngăng gọn bởi chữ in nghiêng *II* ta có

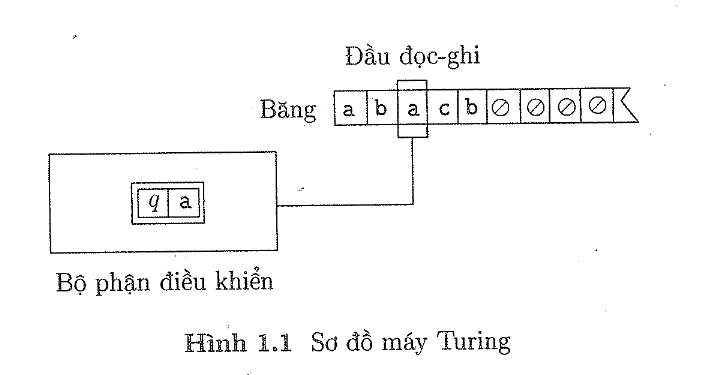
**\* Mô hình tính toán đoán nhận ngôn ngữ**

Cho II là một bài toán quyết định, L() là ngôn ngữ trên bảng chữ Σ biểu diễn dữ kiện của II và *II* là ngôn ngữ tương với bài toán II. giả sử tồn tại một mô hình tính toán hình thức mà khi xử lý trên mỗi xâu thuộc L(), hay thuộc nói chung, nó có thể phân biệt được xâu nào thuộc *II*  và xâu nào không, nghĩa là mô hình tính toán ấy có khả năng “nhận biết” hay đoán nhận ngôn ngữ *II*  khi đó ta dễ dàng thu được lời giải của bài toán quyết định II.

Như vậy thay cho việc xây dựng thuật toán giải bài toán quyết định cho trước ta có thể tìm kiếm một mô hình tính toán hình thức nào đó để đoán nhận ngôn ngữ tương ứng với bài toán ấy. Một trong những mô hình như vậy được Alan turing đề xuất , và vì thế nó được mang tên ôn đó là máy Turing.

## **Câu 7. Mô tả cấu tạo và nguyên tắc hoạt động của máy Turing (trang 36 – đầu trang 38)**

**Cấu tạo. *Máy turing*** *(Turing machine), hay* ***máy Turing tất định một băng đơn*** *(single-tape deterministic Turing machine)*, là một mô hình xử lý tin tự động, mà mỗi dữ liệu đầu vào được biểu diễn dưới dạng một từ trên bảng chữ Σ nào đó. Quá trình hoạt động của máy diễn ra trong thời gian rời rạc t = 0, 1, 2, ... Máy bao gồm một bộ phận điều khiển, một bằng vô hạn về phía phải, và một đầu đọc-ghi kết nối bộ phận điều khiển với băng.



\* *Băng* (hay chính xác là *băng đơn*) được chia thành các ô bắt đầu từ trái qua phải, mỗi ô chứa một ký tự đặc biệt ∅ được dùng để biểu thị *ô trống (blank)*, và có thể chứa một vài ký tự cần thiết khác. Tại thời điểm ban đầu t = 0, dữ liệu đầu vào (tức một từ thuộc Σ\*) được ghi trên băng bắt đầu từ ô đầu tiên, còn lại là các ô trống.

\**Đầu đọc-ghi* tại mỗi thời điểm hoạt động của máy soi xét một ô trên băng, có nhiệm vụ đọc ký tự ở ô đó rồi thông báo cho bộ phận điều khiển để được chỉ dẫn thay ký tự đó bằng một ký tự khác, và sau đó có thể chuyển sang ô bên cạnh ở bên trái hoặc bên phải (tương đương với việc dịch chuyển băng sang phải hoặc sang trái một ô). Tại thời điểm t = 0, đầu đọc ghi soi xét ô đầu tiên.

\**Bộ phận điều khiển* có một tập hữu hạn các trạng thái Q, và tại mỗi thời điểm hoạt động nso chỉ ở một trạng thái. Tập Q chứa 3 trạng thái đặc việt , và , trong đó là trạng thái của bộ phạn điều khiển tại thời điểm t = 0 nên được gọi là***Trạng thái ban đầu*** *(start state), là* ***Trạng thái chấp nhận (accept state) và*** *là* ***trạng thái bác bỏ( reject state).*** Đây là hai trạng thái khi máy dừng hoạt động nên còn được gọi là ***Trạng thái kết thúc*** *(final state).* Có thể nói rằng, bộ phận điều khiển có nhiệm vụ điều hành mọi hoạt động của máy theo bảng lệnh được cài đặt sẵn.

*Bảng lệnh*  của máy bao gồm các lệnh dạng qxHry, trong đó q và r là các trạng thái thuộc Q, x và y là các ký tự thuộc Γ, còn H ∈ {L, R, S}. Lệnh qxHry có nội dung như sau: Bộ phận điều khiển chuyển từ trạng thái hiện thời q sang trạng thái r và đầu đọc-ghi thay ký tự x ở ô nó đang nhìn bằng ký tự y rồi di chuyển sang trái một ô, sang phải một ô hay đứng yên tùy thuốc H = L, R, hay S. Bảng lệnh cần được xác định một cách chuẩn xác và đảm bảo mục đích tính toán của máy.

\***Nguyên tắc hoạt động**

Vào thời điểm ban đầu, trạng thái của bộ phận điều khiển, việc nạp dữ liệu đầu vào và vị trí của đầu đọc-ghi được xác định theo quy ước đã nêu trên. Quá trình xử lý tin trên mỗi dữ liệu đầu vào được diễn ra theo các lệnh của máy. Tại mỗi thời điểm hoạt động, khi bộ phận điều khiển ở trạng thái q và đầu đọc-ghi đọc được ký tự x trên băng, nếu máy có lệnh với khúc đầu qx, thì máy thực hiện theo lệnh đó với nội dung nói trên. Trong trường hợp không có lệnh nào như vậy, máy dừng hoạt động. Việc thực hiện một lệnh của máy được diễn ra trong khoảng thời gian giữa hai thời điểm kế tiếp nhau, và được gọi là ***phép biến đổi*** *(step),* hay còn được gọi là ***phép biến đổi cơ bản***, của máy. Rõ ràng, đây là một phép biến đổi vào loại đơn giản nhất. Quá trình được tiếp diễn cho đến khi máy có thể đi đến những quyết định cuối cùng đối với dữ liệu đầu vào. Dữ liệu đầu vào được ***chấp nhận*** hay ***bác bỏ*** tùy thuộc vào sự thể máy dừng ở trạng thái hay ở trạng thái khi quá trình xử lý kết thúc, quá trình tính toán của máy trên đầu vào sẽ không bao giờ dừng.

## 

## **Câu 8. Trình bày các định nghĩa về: Máy Turing chấp nhận đầu vào; Ngôn ngữ của máy Turing; Ngôn ngữ đoán nhận được; Ngôn ngữ khẳng định được; Thuật toán. Luận đề Church-Turing. ( Trang 50, 52, 80,81)**

**• Máy turing chấp nhận đầu vào:** cho máy turing M vs bảng chữ vào ∑. ta nói rằng máy turing M chấp nhận đầu vào w, nếu w uv, nghĩa là khi tính toán trên từ vào w, máy M chuyển từ hình thái ban đầu đến hình thái chấp nhận.

**•** **Ngôn ngữ của máy Turing:** Tập các từ vào mà máy M chấp nhận tạo thành ngôn ngữ chấp nhận đc của M, được gọi là ngôn ngữ của máy Turing M và được ký hiệu bởi LM:

• **Ngôn ngữ đoán nhận được:** Ngôn ngữ L được gọi là đoán nhận đc theo turing, hay đơn giản là đoán nhận đc, nếu nó là ngôn ngữ chấp nhận đc của 1 máy turing nào đó, nghĩa là tồn tại 1 máy turing M sao cho L= LM. khi đó ta nói rằng “máy turing M đoán nhận ngôn ngữ L” hay “ngôn ngữ L được đoán nhận bởi máy turing M”.

**•** **Ngôn ngữ khẳng định được:** Ngôn ngữ L được gọi là khẳng định đc hay cụ thể hơn là khẳng định đc theo turing, nếu nó được đoán nhận bởi máy quyết định nào đó. Trong trường hợp đó, ta nói rằng “máy turing dừng M khẳng định ngôn ngữ L” hay “ngôn ngữ L được khẳng định bởi máy turing dừng M”; ngược lại, ngôn ngữ L được gọi là không khẳng định đc.

**•** **Thuật toán** là máy turing dừng. Thuật toán được định nghĩa bởi mô hình máy Turing còn được gọi là thuật toán máy Turing. Thuật toán máy Turing không chỉ là một khái niệm chính xác về mặt toán học được thừa nhận rộng rãi nhất như đã nêu trên, mà mô hình toán học này còn được sử dụng như một công cụ khá tiện lợi trong việc nghiên cứu các quá trình tính toán nói chung, và đặc biệt trong việc phân tích độ phức tạp tính toán. Như vậy, tùy thuộc vào mục đích sử dụng, hiện tại về thuật toán có hai cách nhận thức khác nhau và cũng đều khá phổ biến. Đó là nhận thức theo nghĩa trực giác và nhận thức theo định nghĩa hình thức - thuật toán máy Turing. Vậy thì, một câu hỏi được đặt ra khá tự nhiên là: Thuật toán theo nghĩa trực giác và thuật toán máy Turing có tương đương nhau, theo nghĩa giải cùng một lớp bài toán, hay không? Dĩ nhiên, sự tương đương này là không thể chứng minh được một cách chặt chẽ về mặt toán học, bởi vì thuật toán theo nghĩa trực giác không phải là một khái niệm chính xác.

**• Luận đề Church-Turing (Church-Turing thesis):** Thuật toán theo nghĩa trực giác đồng nhất với thuật toán máy Turing. Mặc dù luận đề Church-Turring không thể được chứng minh, nhưng trải qua hơn 70 năm kể từ khi luận đề này được đưa ra, thực tiễn ngày càng củng cố thêm niềm tin vào sự đúng đắn của nó. Ngày nay luận đề này được gần như mọi người thừa nhận. 1.3 Luận đề Church- Turing và Thuật toán 81 Luận đề Church-Turing cùng với Định nghĩa về thuật toán nêu trên cho ta một định nghĩa tổng quát về thuật toán theo nghĩa rộng nhất, tạo cơ sở thiết yếu để tiến hành việc chứng minh “không có thuật toán” giải một bài toán nào đó.

## **Câu 9. Trình bày bài toán chấp nhận (cuối trang 85- đầu trang 86)\**

Đầu tiên ta xem xét bài toán kiểm tra xem liệu máy turing có chấp nhận một từ vào bất kỳ cho trước. Bài toán này được gọi là bài toán chấp nhận đối với máy turing được ký hiệu là . Để chứng minh tính không giải được của bài toán ta sẽ chứng tỏ rằng ngôn ngữ tương ứng của nó là không khẳng định được, tức không tồn tại máy turing dừng đoán nhận ngôn ngữ ấy.

Ngôn ngữ tương ứng với bài toán này được ký hiệu là và được xác định như sau:

= {<M,w> | M là máy Turing và M chấp nhận w}.

Định lý 1.4.1: ngôn ngữ là không khẳng định được

Trước khi bắt đầu chứng minh định lý ta lưu ý rằng là ngôn ngữ đoán nhận được. Như vậy lớp các ngôn ngữ đoán nhận rộng hơn hẳn lớp các ngôn ngữ khẳng định được. Việc đòi hỏi máy turing phải dừng trên mọi từ vào phần nào hạn chế các ngôn ngữ mà máy có thể đoán nhận. Máy turing U sau đây đoán nhận :

U =” trên đầu vào <M, w> trong đó M là một máy Turing và w là một từ:

1. Mô tả việc tính toán của M trên từ w
2. Nếu đầu vào bất cứ lúc nào máy M ở trạng thái chấp nhận nếu bao giờ M ở trạng thái bác bỏ

Nhận thấy rằng máy U không dừng trên <M,w> nếu như M không dừng trên w và do nó không thể khẳng định . Nếu như có thuật toán để kiểm định được rằng M không dừng trên w, nó cần phải bác bỏ <M, w>. Nói cách khác, việc kiểm tra xem liệu M có dừng không trên w là yếu tố quyết định sự tồn tại thuật toán. Ta sẽ chứng tỏ rằng không tồn tại thuật toán để kiểm tra yếu tố quan trọng này.

## **Câu 10. Trình bày Quy chuẩn về thời gian hoạt động của máy Turing (mục 2.1.1, không cần ví dụ)**

**Quy chuẩn về thời gian hoạt động của máy**

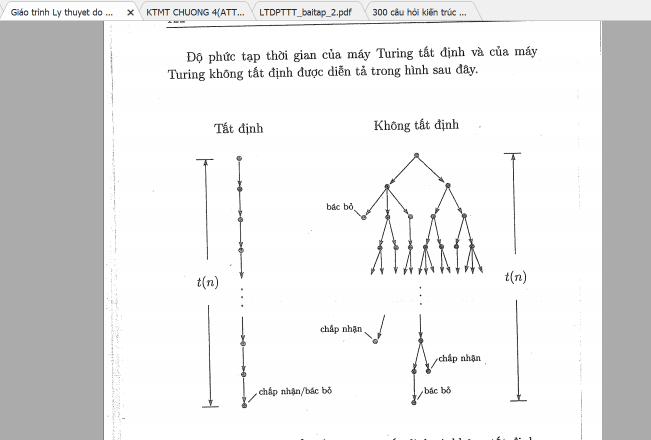
Ta hãy bắt đầu bằng việc xem xét quá trình tính toán của một máy Turing dừng, tức máy quyết định hay thuật toán máy Turing, và từng bước xác lập một quy chuẩn cho thời gian hoạt động của máy Turing nói chung. Trước tiên, qua việc nghiên cứu những máy Turing tất định trong Chương 1, và cụ thể là qua việc quan sát quá trình tính toán của máy trên mỗi từ vào, ta đi đến những nhận xét sau đây. Giả sử M là một máy Turing tất định một băng dừng với bảng chữ vào ∑, trạng thái ban đầu 4% và các trạng thái kết thúc ,. Khi đó quá trình tính toán của máy M trên mỗi từ vào w ∈ Σ\* là một quá trình chuyển đổi các hình thái của máy, kể từ hình thái ban đầu đến hình thái kết thúc: ... .... → trong đó = w là hình thái ban đầu và là hình thái kết thúc của máy M, tức = u với {,}. Hơn thế nữa, việc máy M chuyển đổi từ hình thái đến hình thái kế tiếp (t = 0, 1, ..., k − 1) được thực hiện nhờ một phép biến đổi cơ bản và, theo quy ước, nó được diễn ra giữa hai thời điểm kế tiếp nhau, nên đòi hỏi đúng 1 đơn vị thời gian. Như vậy, thời gian tính toán của máy M trên từ vào u, được ký hiệu là (w), chính là số lần thực hiện các phép biến đổi cơ bản để máy chuyển đổi từ hình thái ban đầu đến hình thái kết thúc; nghĩa là (w) = k

Dĩ nhiên, đối với máy Turing M, điều mà ta cần không chỉ là thời gian tính toán của máy trên mỗi từ vào, mà phải là một tiêu chuẩn thích hợp để biết được thời gian hoạt động của máy khi nó thực hiện một công việc cụ thể, chẳng hạn như đoán nhận một ngôn ngữ hoặc tính một hàm cho trước. Muốn vậy, ta hãy khảo sát quá trình tính toán của một máy Turing dừng nào đó khẳng định một ngôn ngữ cho trước

## **Câu 11. Trình bày định nghĩa độ phức tạp thời gian của máy Turing tất định dừng và máy Turing không tất định dừng, có vẽ hình ảnh minh họa (cuối trang 121- đầu trang 122).**

- Cho M là một máy Turing tất định dừng. Độ phức tạp thời gian hay thời gian hoạt động của M là hàm t : N->N, mà giá trị t(n) là số lần tối đa các phép biến đổi cơ bản được sử dụng trong quá trình tính toán cảu M trên bất cứ từ vào nào của độ dài n. Nếu t(n) là độ phức tạp thời gian của M thì ta nói rằng M hoạt động trong thời gian t(n) hoặc M là máy Turing thời gain t(n). Ta thường dùng n để chỉ độ dài của từ vào

- Cho một máy Turing không tất định dừng N. Độ phức tạp thời gian cảu máy Turing không tất định N là hàn t : N->N, mà giá trị t(n)là số lần tối đã các phép biến đổi cở bản được sử dụng trong quá trình tính toán của máy N trên mỗi từ vào độ dài n là theo từng nhánh trong cây tính toán của máy.



## **Câu 12. Trình bày nội dung phân tích thuật toán (mục 2.1.2, không cần ví dụ)**

Để giải chọn vẹn một bài toán cho trước, ta không chỉ dừng ở việc xây dựng thuật toán mà còn tiến hành phân tích thuật toán, nhằm xác định độ phức tạp thời gian của thuật toán và qua đố cho thấy tính hữu hiệu của nó. Việc phân tích thuật toán giúp ta phát hiện “ những trường hợp xấu nhất” đối với thuật toán , những trường hợp mà khi tính toán cần một lượng thời gian lớn nhất, và tiến hành xác định độ phức tạp thời gian của thuật toán

Việc phân tích thuật toán đối với máy Turing giúp ta hiểu được cặn kẽ về bản chất của các quá trình tính toán, qua đó giúp ta tìm được cận trên tiệm cận , xác định chính xác độ phức tạp thời gian của máy Turing , gợi mở cho ta cải tiến cách tính toán hoặc lựa chọn mô hình tính toán hiệu quả hơn.

Do lượng thời gian thuật toán cần đến trong trường hợp xấu nhất thường được diễn đạt bởi một biểu thức phức tạp, nên thông thường ta tìm cách ước lượng nó bởi biểu thức đơn giản. Mặt khác, theo yêu cầu, ta cũng chỉ cần biết được lượng thời gian khi tính toán trên những đầu vào cỡ lớn. Để đáp ứng những yêu cầu thích hợp này, khái niệm O-lớn thường được sử dụng và tỏ ra khá tiện lợi trong việc diễn tả độ phức tạp thời gian của thuật toán. Như vậy, khi không cần thiết phải xác định chính xác, độ phức tạp thời gian của thuật toán được thể hiện một cách tương đối bởi cận trên tiệm cận của nó.

Như đã thấy, máy Turing có khả năng diễn tả một cách tỉ mỉ các quá trình tính toán và được thừa nhận là một mô hình toán học của thuật toán. Bởi vậy, việc phân tích thuật toán đối với máy Turing sẽ cho ta hiểu được một cách cặn kẽ về bản chất của các quá trình tính toán, và qua đó không chỉ giúp ta tìm được cận trên tiệm cận hoặc xác định chính xác độ phức tạp thời gian của máy Turing mà còn gợi mở cho ta cải tiến cách tính toán hoặc lựa chọn mô hình tính toán khác hiệu quả hơn

## **Câu 13. Trình bày nội dung các định lý về quan hệ thời gian giữa các loại máy (mục 2.1.3, không cần chứng minh)**

**Định lý 1:** Giả sử t: N -> N là một hàm mà t(n) ≥ n. Khi đó mỗi máy Turing tất định nhiều băng thời gian t(n) đều có máy Turing tất định một băng thời gian O[t2(n)] tương đương

**Định lý 2:** Giả sử t: N -> N là một hàm mà t(n) ≥ n. Khi đó mỗi máy Turing không tất định thời gian t(n) đều có máy Turing tất định một băng thòi gian 2O[t(n)] tương đương

## **Câu 14. Trình bày về tg đa thức.**

Về mặt lý thuyết, sự khác nhau về tg tính toán ở mức độ đa thức đc coi là nhỏ, trong khi đó sự khác nhau ở mức độ hàm mũ là rất lớn. vậy tại sao ta lại chọn sự khác biệt giữa hàm đa thức và hàm mũ hơn là giữa các hàm khác.

Một mặt là do có sự khác nhau rất lớn giữa tốc độ tăng trưởng của đa thức và của hàm mũ, mặt khác tốc độ tăng trưởng của đa thức cũng đủ lớn nhưng vẫn trong phạm vi quản lý được, trong khi đó hàm mũ tăng với một tốc độ to lớn đến mức k thể kiểm soát nổi. Ví dụ như đa thức và hàm mũ , vs 1000,…. Vì vậy, những thuật toán thời gian đa thức có thể đc coi là đủ nhanh và chấp nhận đc, còn những thuật toán thời gian hàm mũ họa hoằn lắm mới sd đến.

Các mô hình máy Turing tất định 1 băng và nhiều băng là tương đương nhau. Ta có thể nói rằng tất cả các mô hình tính toán tất định thích hợp vs các máy tính thực tế đều là tương đương đa thức. Nghĩa là, bất cứ một mô hình nào như vậy cũng mô phỏng được mô hình khác chỉ với một lượng thời gian tăng thêm cỡ đa thức.

Từ đây ta tập trung xem xét các khía cạnh của lý thuyết độ phức tạp tính toán mà chúng k bị ảnh hưởng bởi những thay đổi về tg ở mức độ đa thức. những thay đổi hoặc những khác nhau như vậy đc coi là k đáng kể và đc bỏ qua. Cách tiếp cận này cho phép chúng ta phát triển lý thuyết theo hướng k phụ thuộc vào sự tuyển chọn mô hình tính toán riêng biệt, để khám phá ra những tính chất của tính toán cơ bản nói chung.

Bỏ qua những khác nhau ở mức độ đa thức k có nghĩa là ta coi những khác nhau như vậy là k quan trọng. ngược lại, sự khác nhau như vậy dù nhỏ cũng có những ý nghĩa nhất định về mặt ứng dụng thực tiễn. không những thế, sự khác nhau đó giúp chúng ta có cái nhìn thống nhất đối với vấn đề phức tạp, đồng thời nó còn cho ta những hiểu biết tổng quát về độ phức tạp tính toán và một hướng đi đúng đắn khi giải quyết các bài toán khó mà ta gặp phải.

## **Câu 15: trình bày định nghĩa lớp phức tạp TIME, lớp P, vì sao lớp P đóng vai trò trung tâm trong ltdpttt và là một lớp rất quan trọng.**

- định nghĩa lớp phức tạp TIME: cho một hàm t:N R+ . Ta định nghĩa lớp phức tạp TIME (t(n)) là lớp tất cả các ngôn ngữ đc khẳng định bởi máy Turing tất định thời gian O[t(n)].

Rõ ràng: Time(n) Time( Time() ...

- Định nghĩa lớp P: P

Nói cách khác, P là lớp tất cả các ngôn ngữ đc khẳng định bởi máy TR tất định thời gian đa thức.

Lớp P đóng vai trò trugn tâm trong lý thuyết độ phức tạp tính toán và là lớp rất quan trọng bởi vì:

1: P không thay đổi đối vs tất cả các mô hình tính toán tương đương đa thức vs máy Turing tất định một băng.

2: P gần như tương ứng với các bài toán giải đc 1 cách thực tế, tức giải đc trên máy tính điện tử.

Lý do thứ nhất cho thấy rằng P là một lớp mạnh về mặt toán học, còn lý do thứ hai cho thấy rằng P thích đáng theo quan điểm thực tế.

## **Câu 16: trình bày định nghĩa lớp phức tạp NTIME, Lớp NP, trình bày một bài toán cụ thể thuộc lớp NP là bài toán đường đi Hamilton …..**

- Định nghĩa lớp phức tạp NTIME: cho một hàm t:N R+ . Ta định nghĩa lớp phức tạp NTIME (t(n)) là lớp tất cả các ngôn ngữ đc khẳng định bởi máy turing không tất định thời gian O[t(n)]

- Định nghĩa lớp NP: NP

Nói cách khác, NP là lớp tất cả các ngôn ngữ đc khẳng định bởi máy Turing k tất định thời gian đa thức

Khi diễn tả và phân tích các thuật toán k tất định thời gian đa thức, ta thường theo sát những thông lệ như đối với thuật toán tất định thời gian đa thức. việc phân tích thuật toán k tất định nhằm chứng tỏ rằng mỗi nhánh tính toán của thuật toán sử dụng k quá một số đa thức các phép biến đổi cơ bản.

VD về bài toán thuộc lớp NP, bài toán tìm đường đi hamilton.

Dữ kiện: cho một đồ thị có hướng G và 2 đỉnh u,v thuộc G.

Câu hỏi: phải chăng trong G tồn tại đường đi Hamilton từ u đến v?

Ngôn ngữ ứng vs bài toán này đc xác định bởi

DIHAMILTON = {<G,u,v> | G là một đồ thị có hướng chứa đường đi Hamilton từ u đến v }.

DIHAMILTON NP

CM: Giả sử G là một đồ thị có hướng với m đỉnh đc gán tên bởi các số tự nhiên từ 1 đến m. khi đó dãy (,….,),,bao gồm các số tự nhiên từ 1 đến m, sẽ tạo thành một đg đi hamilton trong G có hướng từ đỉnh u đến đỉnh v, nếu các điều kiện sau thỏa mãn: Trước tiên, tất cả các ki ấy phải khác nhau, tiếp đến, u= và v=. cuối cùng, đối vs mỗi 1 i m-1 trong đồ thị G có cung (,). Ta dễ dàng thu đc thuật toán tg hàm mũ cho bài toán, bằng cách kiểm tra lần lượt toàn bộ các dãy nêu trên xem dãy nào tạo thành đg đi hamilton.

Để xây dựng máy Turing ko tất định tg đa thức khẳng định DIHAMPATH, thay vì duyệt toàn bộ ta lựa chọn một cách không tất định một dãy nào đó và tiến hành kiểm tra xem liệu dãy ấy có thỏa mãn các điều kiện nêu trên háy không. Bằng cách này ta thu đc máy Turing k tất định N cho bài toán.

Máy N đc mô tả như sau:

N= “trên từ vào <G,u,v>, trong đó G là một đồ thị có hướng vs m đỉnh chứa hai đỉnh u,v”:

1. Đề xuất một cách k tất định một dãy gồm m số tự nhiên khác nhau (,…,), đc chọn k tất định trong tập {1,2,….,m} sao cho =u và =v.
2. Đối vs mỗi I từ 1 đến m-1, lần lượt kiểm tra xem liệu G có chứa cung (,) hay k. nếu G k chứa dù bất cứ cung nào, thì bác bỏ. ngược lại mọi việc kiểm tra đều trôi chảy, thì chấp nhận.

## **Câu 17. Trình bày các định nghĩa: độ phức tạp kgian của máy TR tất định 1 băng dừng, máy TR tất định nhiều băng dừng, máy TR k tất định dừng. Định lý về mối quan hệ giữa tg và kg.**

- Cho M là một máy Turing tất định một băng dừng. độ phức tạp ko gian của máy Turing M là một hàm s: NN, trong đó giá trị s(n) là số tối đa các ô trên băng mà máy M sử dụng tại mỗi thời điểm trong quá trình tính toán của máy trên bất cứ từ vào nào với cùng độ dài n. Khi máy Turing M có độ phức tạp ko gian s(n) ta nói rằng M hoạt động trong ko gian s(n) hay M là máy Turing không gian s(n).

- Cho M là một máy Turing tất định nhiều băng dừng. độ phức tạp kg của máy Turing M là một hàm s: N N, trong đó giá trị s(n) là số tối đa các ô trên băng mà máy M sử dụng tại mỗi thời điểm trong quá trình tính toán của máy trên bất cứ từ vào nào với cùng độ dài n.

- Cho N là một máy Turing k tất định dừng. độ phức tạp ko gian của máy Turing ko tất định N là một hàm s: NN, trong đó giá trị s(n) là số tối đa các ô trên băng mà máy N sử dụng tại mỗi thời điểm trong quá trình tính toán theo mọi nhánh thuộc cây tính toán của máy N trên bất cứ từ vào nào với cùng độ dài n

- Định lý về mối quan hệ giữa thời gian và ko gian: Giả sử M là 1 máy Turing dừng khi đó:

1: nếu M có độ phức tạp thời gian f(n), f(n)n, thì độ phức tạp kg của M k vượt quá f(n)+1.

2: nếu M có độ phức tạp kg là f(n), f(n)log n, thì độ phức tạp thời gian của M k vượt quá

# **Câu 1**. **Xây dựng máy Turing 1 băng đoán nhận ngôn ngữ { 0i1j với điều kiện nào đó của i, j, ví dụ i>j, i<j, i=j+1, j=i+1, i=j-1, i=j+2}**

Máy Turing 1 băng đoán nhận ngôn ngữ {0i1j với ...} được xác định như sau: M = {Q, {0,1}, {0,1,Ø}, 𝛿, q0, qy, qn }  
Trong đó: Q = {q0, q0’, q1, q2, q3, q4, ..., qy, qn}  
Hàm chuyển được xác định như sau:

|  |  |
| --- | --- |
| **Điều kiện: i<j**  Hàm chuyển:  𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (qn, 0, S) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 0) = (q1, Ø, R) 𝛿(q0’, Ø) = (qn, Ø, S) 𝛿(q0’, 1) = (qy, 1, S) | **Điều kiện i>j**  Hàm chuyển: 𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (qy, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, 0) = (qn, 0, S) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 0) = (q1, Ø, R) 𝛿(q0’, Ø) = (qn, Ø, S) |
| **Điều kiện: i = j+1**  Hàm chuyển: 𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (qn, 0, S) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 1) = (qn, 1, S) 𝛿(q0’, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (q5, Ø, L) 𝛿(q5, 0) = (q6, 0, L) 𝛿(q6, 0) = (qn, 0, S) 𝛿(q6, Ø) = (qy, Ø, R) | **Điều kiện j = i+1**  Hàm chuyển: 𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (qn, 0, S) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, S) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 0) = (q1, Ø, R) 𝛿(q0’, Ø) = (qn, Ø, S) 𝛿(q0’, 1) = (q5, 1, R) 𝛿(q5, 1) = (qn, 1, S) 𝛿(q5, Ø) = (qy, Ø, L) |
| **Điều kiện: i = j+2**  Hàm chuyển: 𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, S) 𝛿(q2, 0) = (qn, 0, R) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 1) = (qn, 1, S) 𝛿(q0’, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (q5, Ø, L) 𝛿(q5, 0) = (q6, 0, L) 𝛿(q6, Ø) = (qn, Ø, S) 𝛿(q6, 0) = (q7, 0, L)  𝛿(q7, 0) = (qn, 0, S)  𝛿(q7, Ø) = (qy, Ø, S) | **Điều kiện: j = i+2**  Hàm chuyển:  𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, Ø, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (qn, 0, S) 𝛿(q2, 1) = (q2, 1, R) 𝛿(q2, Ø) = (q3, Ø, L) 𝛿(q3, 1) = (q4, Ø, L) 𝛿(q4, 1) = (q4, 1, L) 𝛿(q4, 0) = (q4, 0, S) 𝛿(q4, Ø) = (q0’, Ø, R) 𝛿(q0’, 0) = (q1, Ø, R)  𝛿(q0’, Ø) = (qn, Ø, S) 𝛿(q0’, 1) = (q5, 1, R) 𝛿(q5, Ø) = (qn, Ø, S) 𝛿(q5, 1) = (q6, 1, R) 𝛿(q6, 1) = (qn, 1, S)  𝛿(q6, Ø) = (qy, Ø, S) |

# **Câu 2. Xây dựng máy Turing 1 băng tính hàm f(n,m) = n\*m-n, n\*m-m, n\*m+n, n\*m+m, n\*m-m+1, n\*m-n+1, n\*m+m-1, n\*m+n-1 (m,n>0)**

Máy Turing 1 băng tính hàm f(n,m) = ... được định nghĩa như sau: M = {Q, {0,1}, {0,1,Ø, x, #, Ω}, 𝛿, q0, qy, qn}

Trong đó: Q = {q0, q0’, q1, q2, q3,q4, q5, q6, q7, q8, q9, q10, q11, q12, q13, q14, q15,... qy, qn}  
Hàm chuyển được xác định như sau:

|  |  |
| --- | --- |
| **Hàm n\*m**  𝛿( q0, Ø) = (qn, Ø, S)  𝛿( q0, 1) = (qn, 1, S)  𝛿( q0, 0) = (q1, Ø, R)  𝛿( q1, Ø) = (qn, Ø, S)  𝛿( q1, 0) =(q1, 0, R)  𝛿( q1, 1) =(q2, 1, R)  𝛿( q2, Ø) =(qn, Ø, S)  𝛿( q2, 1) =(qn, 1, S)  𝛿( q2, 0) =(q3, 0, R)  𝛿( q3, 0) =(q3, 0, R)  𝛿( q3, 1) =(qn, 1, S)  𝛿( q3, Ø) =(q4, #, L)  𝛿( q4, 0) =(q4, 0, L)  𝛿( q4, 1) =(q5, 1, R)  𝛿( q5, 0) =(q6, x, R)  𝛿( q6, 0) =(q6, 0, R) | 𝛿( q6, #)=(q6, #, R)  𝛿( q6, Ø)=(q4, 0, L)  𝛿( q4, #)=(q4, #, L)  𝛿( q4, x)=(q5, x, R)  𝛿( q5, #)=(q7, #, L)  𝛿( q7, x)=(q7, 0, L)  𝛿( q7, 0)=(q7, 0, L)  𝛿( q7, 1)=(q7, 1, L)  𝛿( q7, Ø)=(q0’, Ø, R)  𝛿( q0’,0)=(q8, Ø, R)  𝛿( q8,0)=(q8, 0,R)  𝛿( q8,1)=(q5, 1,R)  𝛿( q0’,1)=(q9, Ø,R) // kết thúc  𝛿( q9,0)=(q9, Ø,R) //  𝛿( q9,#)=(qy, Ø,R) // |

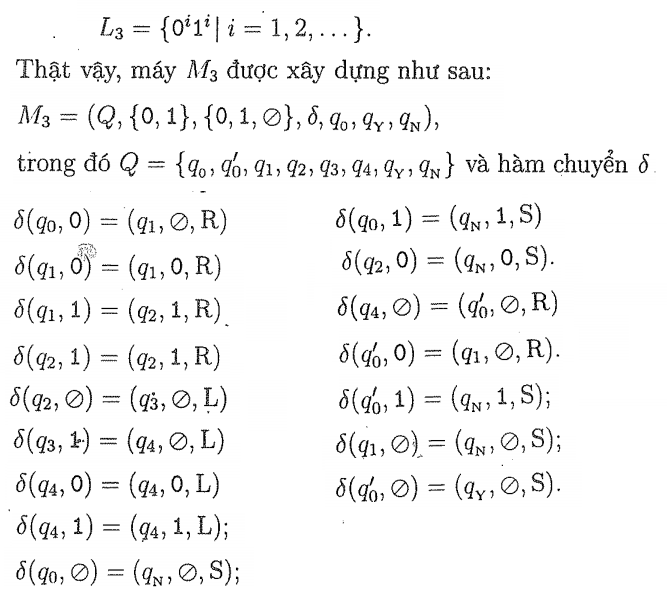
|  |  |
| --- | --- |
| **f(n,m)=n\*m-n**  𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, x, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (q3, 0, R) 𝛿(q2, 1) = (qn, 1, S) 𝛿(q2, Ø) = (qn, Ø, S) 𝛿(q3, 0) = (q3, 0, R) 𝛿(q3, 1) = (qn, 1, S) 𝛿(q3, Ø) = (q4, #, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, 1) = (q5, 1, R) 𝛿(q5, 0) = (q6, x, R) 𝛿(q6, 0) = (q6, 0, R) 𝛿(q6, #) = (q6, #, R) 𝛿(q6, Ø) = (q4, 0, L)  𝛿(q4, #) = (q4, #, L)  𝛿(q4, x) = (q5, x, R) 𝛿(q5, #) = (q7, #, L)  𝛿(q7, x) = (q7, 0, L) 𝛿(q7, 1) = (q8, 1, L) 𝛿(q8, 0) = (q8, 0, L)  𝛿(q8, x) = (q0’, x, R)  𝛿(q0’, 0) = (q9, x, R) | 𝛿(q9, 0) = (q9, 0, R)  𝛿(q9, 1) = (q5, 1, R) 𝛿(q0’, 1) = (q10, 1, L)  𝛿(q10, x) = (q10, 0, L) 𝛿(q10,Ø) = (q11, Ø, R) 𝛿(q11, 0) = (q12, Ø, R)  𝛿(q12, 0) = (q12, 0, R) 𝛿(q12, 1) = (q12, 1, R) 𝛿(q12, #) = (q12, #, R) ***𝛿(q12, Ø) = (q13, Ø, L) 𝛿(q13, 0) = (q14, Ø, L) 𝛿(q14, 0) = (q14, 0, L) 𝛿(q14, #) = (q14, #, L) 𝛿(q14, 1) = (q14, 1, L) 𝛿(q14, Ø) = (q11, Ø, R)***𝛿(q11, 1) = (q15, Ø, R) 𝛿(q15, 0) = (q15, Ø, R) 𝛿(q15, #) = (qy, #, R)  *Dấu x khoanh tròn bên ngoài: x*  **F(n,m) = n\*m+n**  **𝛿(q12, Ø) = (q13, 0, L) 𝛿(q13, 0) = (q13, 0, L) 𝛿(q13, #) = (q13, #, L) 𝛿(q13, 1) = (q13, 1, L) 𝛿(q13, Ø) = (q11, Ø, R)** |
| **F(n,m) = n\*m+n+1**  𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, x, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (q3, 0, R) 𝛿(q2, 1) = (qn, 1, S) 𝛿(q2, Ø) = (qn, Ø, S) 𝛿(q3, 0) = (q3, 0, R) 𝛿(q3, 1) = (qn, 1, S) 𝛿(q3, Ø) = (q4, #, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, 1) = (q5, 1, R) 𝛿(q5, 0) = (q6, x, R) 𝛿(q6, 0) = (q6, 0, R) 𝛿(q6, #) = (q6, #, R) 𝛿(q6, Ø) = (q4, 0, L)  𝛿(q4, #) = (q4, #, L)  𝛿(q4, x) = (q5, x, R) 𝛿(q5, #) = (q7, #, L)  𝛿(q7, x) = (q7, 0, L) 𝛿(q7, 1) = (q8, 1, L) 𝛿(q8, 0) = (q8, 0, L)  𝛿(q8, x) = (q0’, x, R)  𝛿(q0’, 0) = (q9, x, R) | 𝛿(q9, 0) = (q9, 0, R) 𝛿(q9, 1) = (q5, 1, R) 𝛿(q0’, 1) = (q10, 1, L)  𝛿(q10, x) = (q10, 0, L) 𝛿(q10,Ø) = (q11, Ø, R) 𝛿(q11, 0) = (q12, Ø, R)  𝛿(q12, 0) = (q12, 0, R) 𝛿(q12, 1) = (q12, 1, R) 𝛿(q12, #) = (q12, #, R) 𝛿(q12, Ø) = (q13, 0, L) 𝛿(q13, 0) = (q13, 0, L) 𝛿(q13, #) = (q13, #, L) 𝛿(q13, 1) = (q13, 1, L) 𝛿(q13, Ø) = (q11, Ø, R)  𝛿(q11, 1) = (q15, Ø, R) 𝛿(q15, 0) = (q15, Ø, R) **𝛿(q15, #) = (qy, 0, R)**  **F(n,m) = n\*m+n-1**  **𝛿(q15, #)=(q16, Ø, R)**  **𝛿(q16, 0)=(qy, Ø, R)** |
| **F(n,m) = n\*m-n+1**  𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, x, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (q3, 0, R) 𝛿(q2, 1) = (qn, 1, S) 𝛿(q2, Ø) = (qn, Ø, S) 𝛿(q3, 0) = (q3, 0, R) 𝛿(q3, 1) = (qn, 1, S) 𝛿(q3, Ø) = (q4, #, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, 1) = (q5, 1, R) 𝛿(q5, 0) = (q6, x, R) 𝛿(q6, 0) = (q6, 0, R) 𝛿(q6, #) = (q6, #, R) 𝛿(q6, Ø) = (q4, 0, L)  𝛿(q4, #) = (q4, #, L)  𝛿(q4, x) = (q5, x, R) 𝛿(q5, #) = (q7, #, L)  𝛿(q7, x) = (q7, 0, L) 𝛿(q7, 1) = (q8, 1, L) 𝛿(q8, 0) = (q8, 0, L)  𝛿(q8, x) = (q0’, x, R)  𝛿(q0’, 0) = (q9, x, R) | 𝛿(q9, 0) = (q9, 0, R)  𝛿(q9, 1) = (q5, 1, R) 𝛿(q0’, 1) = (q10, 1, L)  𝛿(q10, x) = (q10, 0, L) 𝛿(q10,Ø) = (q11, Ø, R) 𝛿(q11, 0) = (q12, Ø, R)  𝛿(q12, 0) = (q12, 0, R) 𝛿(q12, 1) = (q12, 1, R) 𝛿(q12, #) = (q12, #, R)  𝛿(q12, Ø) = (q13, Ø, L) 𝛿(q13, 0) = (q14, Ø, L) 𝛿(q14, 0) = (q14, 0, L) 𝛿(q14, #) = (q14, #, L) 𝛿(q14, 1) = (q14, 1, L) 𝛿(q14, Ø) = (q11, Ø, R)𝛿(q11, 1) = (q15, Ø, R) 𝛿(q15, 0) = (q15, Ø, R) ***𝛿(q15, #) = (qy, 0, R)***  **Hàm F (n,m)=n\*m-n-1**  **𝛿(q15, #)=(q16, Ø, R)**  **𝛿(q16, 0)=(qy, Ø, R)** |
| **F(n,m) = n\*m-m** 𝛿(q0, Ø) = (qn, Ø, S) 𝛿(q0, 1) = (qn, 1, S) 𝛿(q0, 0) = (q1, x, R) 𝛿(q1, Ø) = (qn, Ø, S) 𝛿(q1, 0) = (q1, 0, R) 𝛿(q1, 1) = (q2, 1, R) 𝛿(q2, 0) = (q3, 0, R) 𝛿(q2, 1) = (qn, 1, S) 𝛿(q2, Ø) = (qn, Ø, S) 𝛿(q3, 0) = (q3, 0, R) 𝛿(q3, 1) = (qn, 1, S) 𝛿(q3, Ø) = (q4, #, L) 𝛿(q4, 0) = (q4, 0, L) 𝛿(q4, 1) = (q5, 1, R) 𝛿(q5, 0) = (q6, x, R) 𝛿(q6, 0) = (q6, 0, R) 𝛿(q6, #) = (q6, #, R) 𝛿(q6, Ø) = (q4, 0, L)  𝛿(q4, #) = (q4, #, L)  𝛿(q4, x) = (q5, x, R) 𝛿(q5, #) = (q7, #, L)  𝛿(q7, x) = (q7, 0, L) 𝛿(q7, 1) = (q8, 1, L) 𝛿(q8, 0) = (q8, 0, L)  𝛿(q8, x) = (q0’, x, R)  𝛿(q0’, 0) = (q9, x, R) | 𝛿(q9, 0) = (q9, 0, R)  𝛿(q9, 1) = (q5, 1, R)  𝛿(q0’, 1) = (q10, Ø, R) **𝛿(q10, 0) = (q11, Ø, R) 𝛿(q11, 0) = (q11, 0, R) 𝛿(q11, #) = (q11, #, R) 𝛿(q11, Ø) = (q12, Ø, L) 𝛿(q12, 0) = (q13, Ø, L)**  **𝛿(q13, 0)=(q13, 0, L) 𝛿(q13, #) = (q13, #, L) 𝛿(q13, Ø) = (q10, Ø, R)** 𝛿(q10, #) = (q14, Ø, R)  𝛿(q14, 0) =(q14, 0,R)  *𝛿(q14, Ø)=(qy, Ø, S)*  *𝛿(q14, Ø)=(qy, 0, S) [+1]*  *𝛿(q14, Ø)=(q15, Ø, L) [-1]*  *𝛿(q15, 0)=(q16, Ø, R)*  *𝛿(q16, Ø)=(qy, Ø, R)*  **F(n,m) = n\*m+m  𝛿(q11, Ø) = (q12, 0, L) 𝛿(q12, 0) = (q12, 0, L)**  **𝛿(q12, #)=(q12, #, L) 𝛿(q12, Ø) = (q10, Ø, R)**  **(Thay q14,15,16 🡪q13,14,15)** |

# **Câu 3. Xây dựng máy Turing 2 băng đoán nhận ngôn ngữ {w#wR , w ∈ {0,1}+}, hoặc ngôn ngữ {w#w, w ∈ {0,1}+} , {w#0wR, w ∈ {0,1}+}**

Máy Turing 2 băng đoán nhận ngôn ngữ ... được xác định như sau: M(2) = {Q, {0,1}, {0,1,Ø,#,Ω}, 𝛿, q0, qy, qn}  
Trong đó: Q = {q0, q1, q2,... , qy, qn}  
Hàm chuyển được xác định như sau:

|  |  |
| --- | --- |
| **ngôn ngữ w#wR**  𝛿(q0, (Ø,Ø)) = (qn, (Ø,Ø), (S,S)) 𝛿(q0, (#,Ø)) = (qn, (#,Ø), (S,S)) 𝛿(q0, (0,Ø)) = (q1, (0,Ω), (S,R))  𝛿(q0, (1,Ø)) = (q1, (1,Ω), (S,R))  𝛿(q1, (Ø,Ø)) = (qn, (Ø,Ø), (S,S))  𝛿(q1, (1,Ø)) = (q1, (1,1), (R,R)) 𝛿(q1, (0,Ø)) = (q1, (0,0), (R,R)) 𝛿(q1, (#,Ø)) = (q2, (#,Ø), (R,L)) 𝛿(q2, (0,0)) = (q2 (0,0), (R,L)) 𝛿(q2, (1,1)) = (q2, (1,1), (R,L)) 𝛿(q2, (0,1)) = (qn, (0,1), (S,S)) 𝛿(q2, (1,0)) = (qn, (1,0), (S,S))  𝛿(q2, (Ø,0)) = (qn, (Ø,0), (S,S)) 𝛿(q2, (Ø,1)) = (qn, (Ø,1), (S,S))  𝛿(q2, (1,Ω)) = (qn, (1,Ω), (S,S)) 𝛿(q2, (0,Ω)) = (qn, (0,Ω), (S,S)) 𝛿(q2, (Ø,Ω)) = (qy, (Ø,Ω), (S,S)) | **ngôn ngữ w#w**  𝛿(q0, (Ø,Ø)) = (qn, (Ø,Ø), (S,S)) 𝛿(q0, (0,Ø)) = (q1, (0,Ω), (S,R))  𝛿(q0, (1,Ø)) = (q1, (1,Ω), (S,R))  𝛿(q0, (#,Ø)) = (qn, (#,Ø), (S,S))  𝛿(q1, (Ø,Ø)) = (qn, (Ø,Ø), (S,S))  𝛿(q1, (0,Ø)) = (q1, (0,0), (R,R))  𝛿(q1, (1,Ø)) = (q1, (1,1), (R,R)) 𝛿(q1, (#,Ø)) = (q2, (#,Ø), (R,S)) 𝛿(q2, (0,Ø)) = (q2 (0,Ø), (R,S)) 𝛿(q2, (1,Ø)) = (q2, (1,Ø), (R,S)) 𝛿(q2, (Ø,Ø)) = (q3, (Ø,Ø), (L,L)) 𝛿(q3, (0,1)) = (qn, (0,1), (S,S))  𝛿(q3, (1,0)) = (qn, (0,1), (S,S)) 𝛿(q3, (0,0)) = (q3, (0,0), (L,L))  𝛿(q3, (1,1)) = (q3, (1,1), (L,L))  𝛿(q3, (#,0)) = (qn, (#,0), (S,S)) 𝛿(q3, (#,1)) = (qn, (#,1), (S,S))  𝛿(q3, (1,Ω)) = (qn, (1,Ω), (S,S)) 𝛿(q3, (0,Ω)) = (qn, (0,Ω), (S,S)) 𝛿(q3, (#,Ω)) = (qy, (Ø,Ω), (S,S)) |
| **w#0wR**  Hàm chuyển: 𝛿(q0, (Ø,Ø)) = (qn, (Ø,Ø), (S,S)) 𝛿(q0, (#,Ø)) = (qn, (#,Ø), (S,S)) 𝛿(q0, (1,Ø)) = (q1, (1,Ω), (S,R))  𝛿(q0, (0,Ø)) = (q1, (0,Ω), (S,R))  𝛿(q1, (0,Ø)) = (q1, (0,0), (R,R)) 𝛿(q1, (1,Ø)) = (q1, (1,1), (R,R)) 𝛿(q1, (Ø,Ø)) = (q1, (Ø,Ø), (S,S)) 𝛿(q1, (#,Ø)) = (q2, (#,Ø), (R,L))  𝛿(q2, (Ø,0)) = (qn, (Ø,1), (S,S)) 𝛿(q2, (Ø,1)) = (qn, (Ø,0), (S,S))  𝛿(q2, (1,1)) = (qn, (1,1), (S,S))  𝛿(q2, (1,0)) = (qn, (1,0), (S,S))  𝛿(q2, (0,1)) = (q3, (0,1), (R,S))  𝛿(q2, (0,0)) = (q3, (0,0), (R,S))  𝛿(q3, (0,1)) = (qn, (0,1), (S,S))  𝛿(q3, (1,0)) = (qn, (1,0), (S,S))  𝛿(q3, (0,0)) = (q3, (0,0), (R,L))  𝛿(q3, (1,1)) = (q3, (1,1), (R,L))  𝛿(q3, (Ø,Ω)) = (qy (Ø,Ω), (S,S)) | **w#1wR**  𝛿(q0, (Ø,Ø)) = (qn, (Ø,Ø), (S,S)) 𝛿(q0, (#,Ø)) = (qn, (#,Ø), (S,S)) 𝛿(q0, (1,Ø)) = (q1, (1,Ω), (S,R))  𝛿(q0, (0,Ø)) = (q1, (0,Ω), (S,R))  𝛿(q1, (Ø,Ø)) = (q1, (Ø,Ø), (S,S))  𝛿(q1, (0,Ø)) = (q1, (0,0), (R,R))  𝛿(q1, (1,Ø)) = (q1, (1,1), (R,R)) 𝛿(q1, (#,Ø)) = (q2, (#,Ø), (R,L))  𝛿(q2, (0,0)) = (qn, (0,0), (S,S)) 𝛿(q2, (0,1)) = (qn, (0,1), (S,S))  𝛿(q2, (1,0)) = (q2, (1,0), (S,L))  𝛿(q2, (1,1)) = (q2, (1,1), (S,L))  𝛿(q2, (1, Ω)) = (q3, (1, Ω), (R,S))  𝛿(q2, (0,0)) = (q3, (0,0), (R,R))  𝛿(q3, (1,0)) = (qn, (1,0), (S,S))  𝛿(q3, (0,1)) = (qn, (0,1), (S,S)) 𝛿(q3, (1,1)) = (q3, (1,1), (R,R))  𝛿(q3, (0,0)) = (q3, (0,0), (R,R))  𝛿(q3, (Ø,Ø)) = (qy (Ø,Ø), (S,S)) |

**Bài tập tính độ phức tạp**

****