Mật mã và độ phức tạp thuật toán (Complexity and Cryptography) Chủ đề 2: Độ phức tạp tính toán

PGS.TS. Nguyễn Đình Hân
(Mobile: 0915.046.320; Email: han.nguyendinh@hust.edu.vn)

Viện Toán ứng dụng và Tin học
Trường Đại học Bách khoa Hà Nội

Độ phức tạp tính toán

1. Độ phức tạp thời gian
   - Lớp phức tạp thời gian
   - Các lớp P và NP
   - Bài toán "P vs NP"
   - Phép dẫn thời gian đa thức và lớp NPC
   - Các bài toán NP đầy đủ
2. Độ phức tạp không gian
3. Những mối quan hệ cơ bản về độ phức tạp
   - Quan hệ giữa thời gian và không gian
   - Quan hệ giữa tất định và không tất định

Độ phức tạp tính toán

An algorithm is practical if and only if it has polynomial running time.

JOHN TALBOT AND DOMINIC WELSH, *Oxford*

- Máy Turing được lựa chọn cho khái niệm thuật toán và do đó việc xây dựng thuật toán giải bài toán cho trước được đưa về việc xây dựng máy Turing khẳng định ngôn ngữ tương ứng với bài toán ấy.
- Việc khảo sát các bài toán thực tế không chỉ dừng lại ở chỗ kết luận bài toán là giải được một cách chung chung, mà phải "giải được một cách thực tế", tức là giải được trong khuôn khổ thời gian và không gian (bộ nhớ) hiện có.
- Vì vậy, việc nghiên cứu độ phức tạp tính toán, theo thời gian và không gian, của máy Turing là hết sức cần thiết.

Độ phức tạp thời gian

Xét một máy Turing tất định một băng dừng với bảng chữ vào Σ, trạng thái ban đầu q_0 và các trạng thái kết thúc y, n. Khi đó quá trình tính toán của M trên mỗi từ vào $\omega \in \Sigma^*$ là một quá trình chuyển đổi các cấu hình của máy, kể từ cấu hình ban đầu đến cấu hình kết thúc:

$$C_0 \vdash_M C_1 \vdash_M \cdots \vdash_M C_t \vdash_M C_{t+1} \vdash_M \cdots \vdash_M C_{k-1} \vdash_M C_k.$$

Trong đó $C_0 = q_0\omega$ là cấu hình ban đầu và C_k là cấu hình kết thúc của M, tức là $C_k = uyv$ hoặc $C_k = unv$.

- Việc máy chuyển đổi từ cấu hình C_t đến cấu hình kế tiếp $C_{t+1}(t = 0, 1, \ldots, k-1)$ được thực hiện nhờ một phép biến đổi cơ bản và theo qui ước, đòi hỏi đúng 1 đơn vị thời gian.
- Thời gian tính toán của máy M trên từ vào ω, được kí hiệu là $t_M(\omega)$, chính là số lần thực hiện các phép biến đổi cơ bản để máy chuyển đổi từ cấu hình ban đầu đến cấu hình kết thúc. Nghĩa là $t_M(\omega) = k$.

Độ phức tạp thời gian

- Xác định thời gian hoạt động của máy Turing, của thuật toán nói riêng có ý nghĩa đáng kể về lý thuyết và ứng dụng, nhằm làm sáng tỏ khả năng cũng như tính hữu hiệu của thuật toán.

- Thời gian hoạt động của máy Turing sẽ được xác định như một hàm chỉ phụ thuộc vào độ dài của từ/xâu vào (nghĩa là độ dài của xâu chữ được dùng để biểu diễn dữ kiện của bài toán).

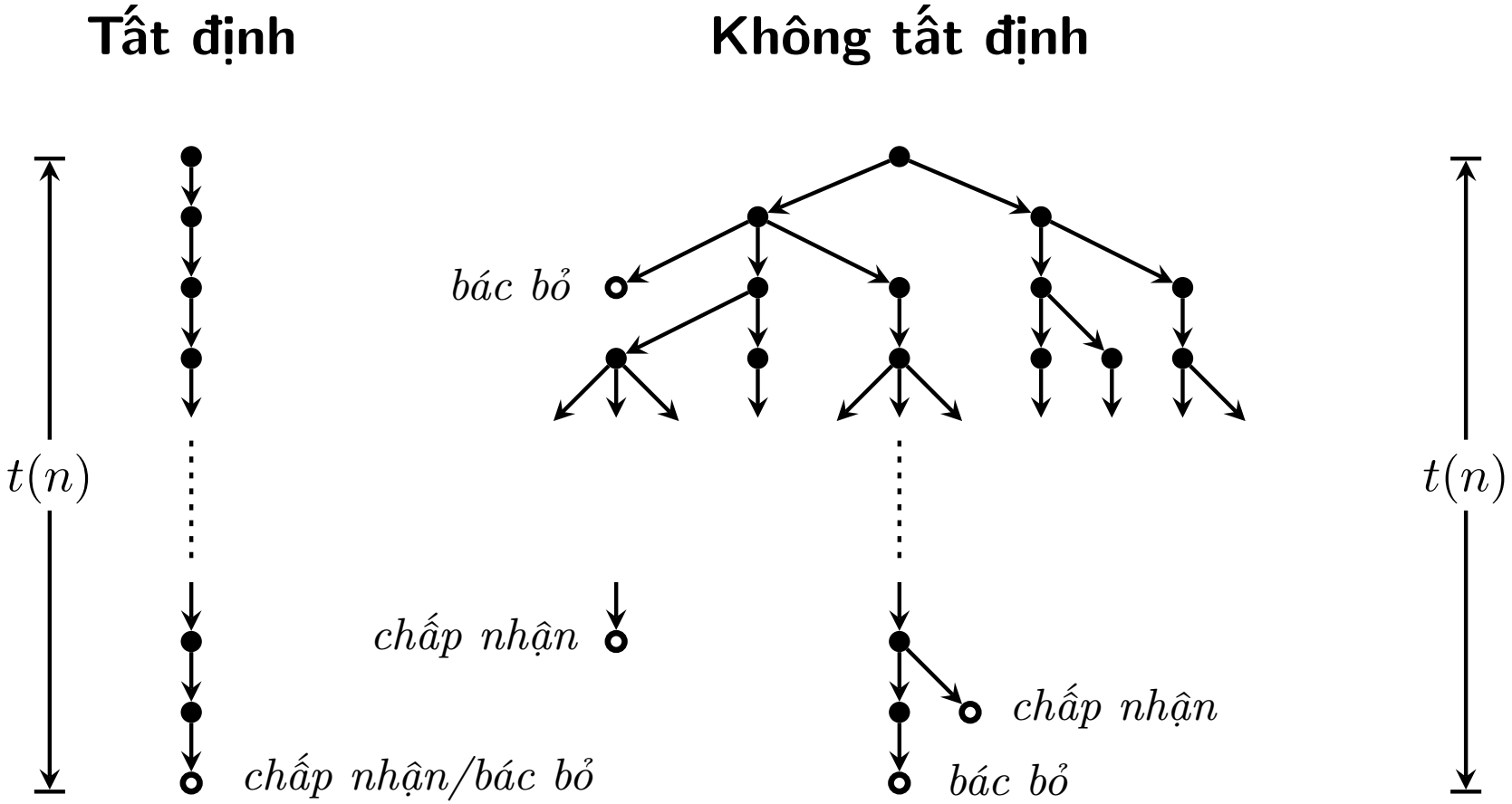
- Thời gian tính toán của thuật toán cũng chỉ được khảo sát theo một vài khía cạnh, tùy thuộc nhu cầu. Chẳng hạn, theo **thời gian trung bình** (nhằm mục đích làm sáng tỏ ý nghĩa thực tiễn của thuật toán), **trường hợp xấu nhất** (theo quan điểm chung là quan trọng để biết được thời gian tính toán tối đa trên những đầu vào cùng "kích cỡ").

Độ phức tạp thời gian

Định nghĩa 2.1 Cho M là một máy Turing tất định dừng (một băng hoặc nhiều băng). **Độ phức tạp thời gian** (time complexity) hay **thời gian hoạt động** (running time) của M là hàm $t : \mathbb{N} \longrightarrow \mathbb{N}$, mà giá trị $t(n)$ là số lần tối đa các phép biến đổi cơ bản được sử dụng trong quá trình tính toán của M trên bất cứ từ vào nào 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 gian** $t(n)$. Ta thường dùng n để chỉ độ dài của từ vào.

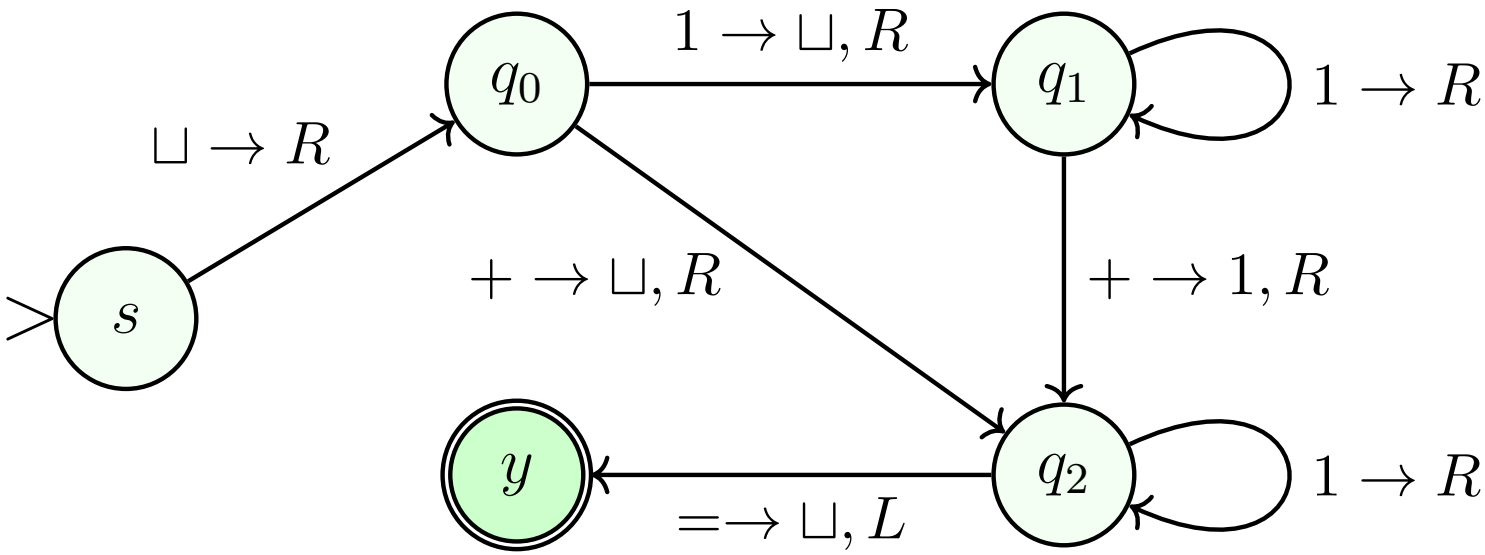
Định nghĩa 2.2 Cho một máy Turing không tất định dừng N. **Độ phức tạp thời gian của máy Turing không tất định** N là hàm $t : \mathbb{N} \longrightarrow \mathbb{N}$, mà giá trị $t(n)$ là số lần tối đa các phép biến đổi cơ bản được sử dụng trong quá trình tính toán của N trên mỗi từ vào độ dài n và theo từng nhánh trong cây tính toán của máy.

Độ phức tạp thời gian



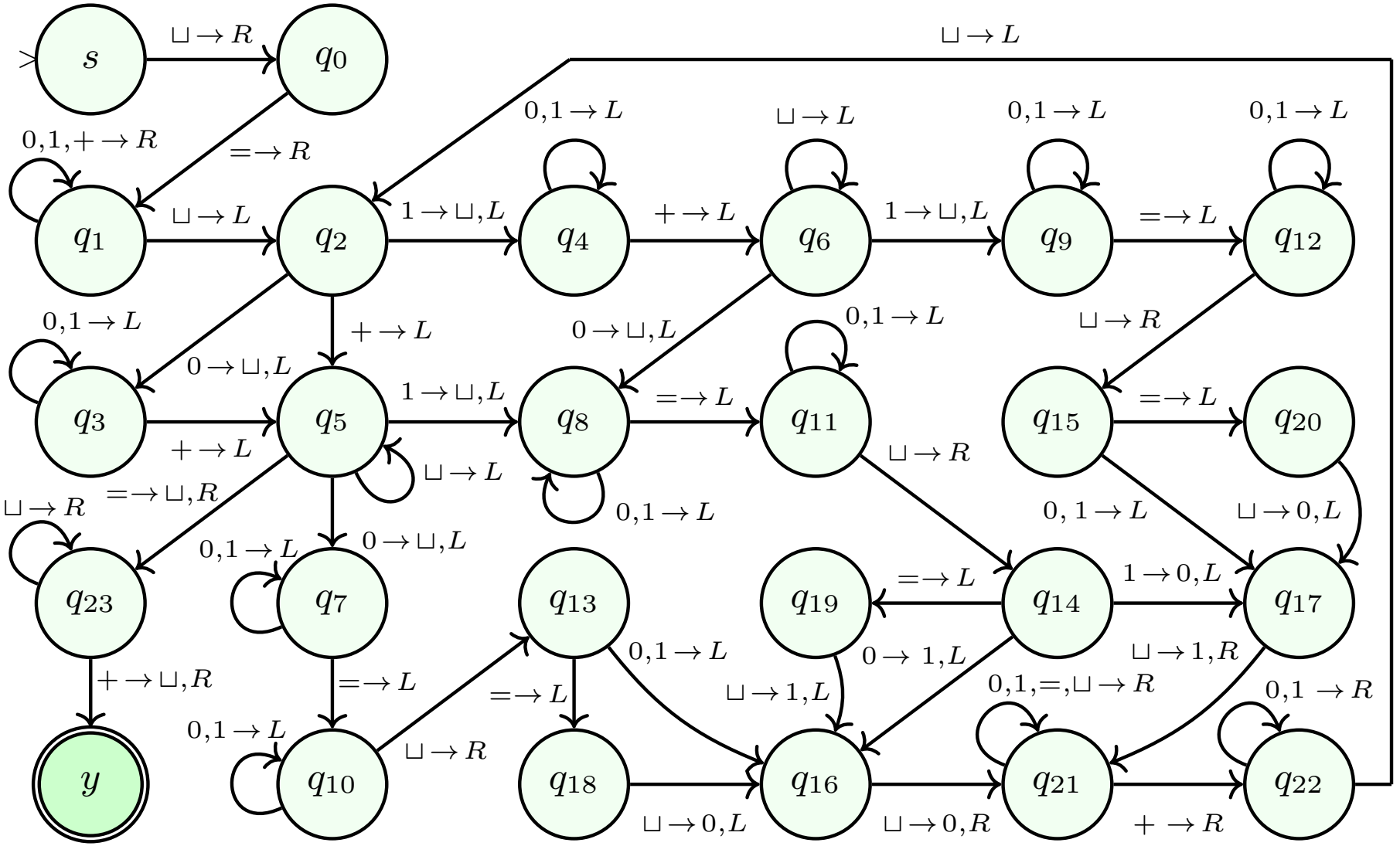
Hình 2.1 Qui chuẩn về thời gian tất định và không tất định

Ví dụ 2.1 Máy Turing M_1 cộng hai số nguyên dương được biểu diễn trong hệ đơn phân (Ví dụ, $5+2$ được mã hóa bởi xâu '$11111+11=$') là $M_1 = (\mathcal{K}, \Sigma, \delta, s, \{y\})$, với $\mathcal{K} = \{s, q_0, q_1, q_2, y\}, \Sigma = \{1, +, =, \sqcup, \triangleright\}$ và $\delta : (\mathcal{K} - \{y\}) \times \Sigma \longrightarrow \mathcal{K} \times \Sigma \times \{L, R\}$ được cho như sau:



Hình 2.2 Máy Turing M_1 cộng hai số nguyên dương với cấu hình ban đầu $(s, \sqcup 1^n + 1^m =), n, m \geq 0$

Độ phức tạp thời gian



Hình 2.3 Máy Turing M_2 cộng hai số nguyên dương trong hệ nhị phân

Độ phức tạp thời gian

Xâu vào a, b	**Số bước của** M_1	**Số bước của** M_2
10	23	<200
1000	2000	<900
10^6	2×10^6	<3000
2^{100}	2.5×10^{30}	<65000

Hình 2.4 So sánh số bước tính toán của các máy Turing M_1 và M_2

Độ phức tạp thời gian

Phân tích thuật toán:

- 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.

- Thông thường, ta chủ yếu muốn biết được lượng thời gian khi thuật toán tính toán trên những đầu vào cỡ lớn. Do vậy, ta sẽ sử dụng cách ước lượng gọi là **phân tích tiệm cận**. Khi đó, độ phức tạp thời gian của thuật toán có thể được ước lượng tương đối bởi các khái niệm O-lớn $(\leq)$, o-nhỏ $(<)$, $\Theta\,(=)$, $\Omega\,(\geq)$ và $\omega\,(>)$.

Định nghĩa 2.3 Cho f và g là các hàm số: $f, g : \mathbb{N} \longrightarrow \mathbb{R}^+$. Ta nói $f(n) = O(g(n))$ nếu tồn tại các số nguyên dương c và n_0 sao cho với mọi số nguyên $n \geq n_0$

$$f(n) \leq cg(n).$$

Khi $f(n) = O(g(n))$, ta nói rằng $g(n)$ là **cận trên tiệm cận** đối với $f(n)$ để nhấn mạnh việc ta bỏ qua thành phần hằng số sai khác.

Định nghĩa 2.4 Cho f và g là các hàm số: $f, g : \mathbb{N} \longrightarrow \mathbb{R}^+$. Ta nói $f(n) = o(g(n))$ nếu

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

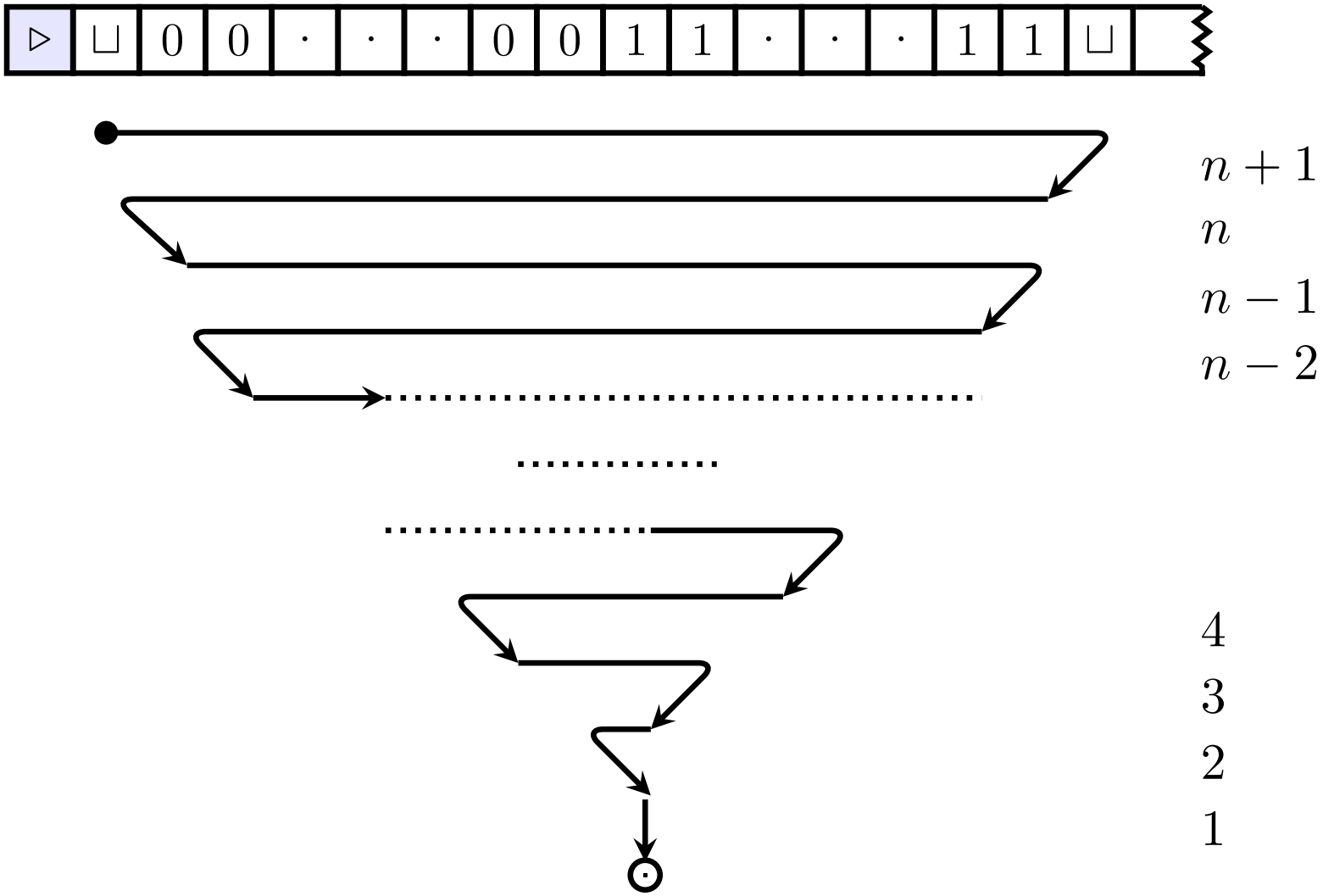
Biểu thức $f(n) = o(g(n))$ nghĩa là, với một số thực $c > 0$ bất kỳ, tồn tại một số n_0 mà $f(n) < cg(n)$ với mọi $n \geq n_0$.

Ví dụ 2.2 Trong các phần trước, ta đã chứng tỏ rằng có tồn tại máy Turing dừng M đoán nhận ngôn ngữ $L = \{0^i 1^i \mid i = 1, 2, \ldots\}$. Ý tưởng cài đặt M được cho như sau:

Với xâu vào $\omega \in L$ bất kì đặt trên băng vào, máy sẽ thực hiện tính toán theo một qui trình lặp mà nội dung mỗi phép lặp bao gồm:

- ⋆ Di chuyển zigzac đầu đọc-ghi từ kí tự khác $\sqcup$ đầu tiên đến kí tự khác $\sqcup$ cuối cùng trên băng và xóa các kí tự ấy nếu chúng tương ứng là 0 và 1; ngược lại, *bác bỏ*.
- ⋆ Khi thực hiện bước lặp đầu tiên, máy đồng thời loại bỏ từ rỗng cũng như những từ đầu vào mà bắt đầu bởi kí tự 1 và những từ vào mà sau kí tự 1 còn xuất hiện kí tự 0.
- ⋆ Khi bắt đầu một phép lặp tiếp theo (máy ở trạng thái q_0'), không kể phép lặp đầu tiên, nếu bắt gặp $\sqcup$, tức băng còn toàn ô trống, thì *chấp nhận*.

Độ phức tạp thời gian



Hình 2.5 Sơ đồ di chuyển của đầu đọc ghi trên băng với từ $0^i1^i, i = n/2$

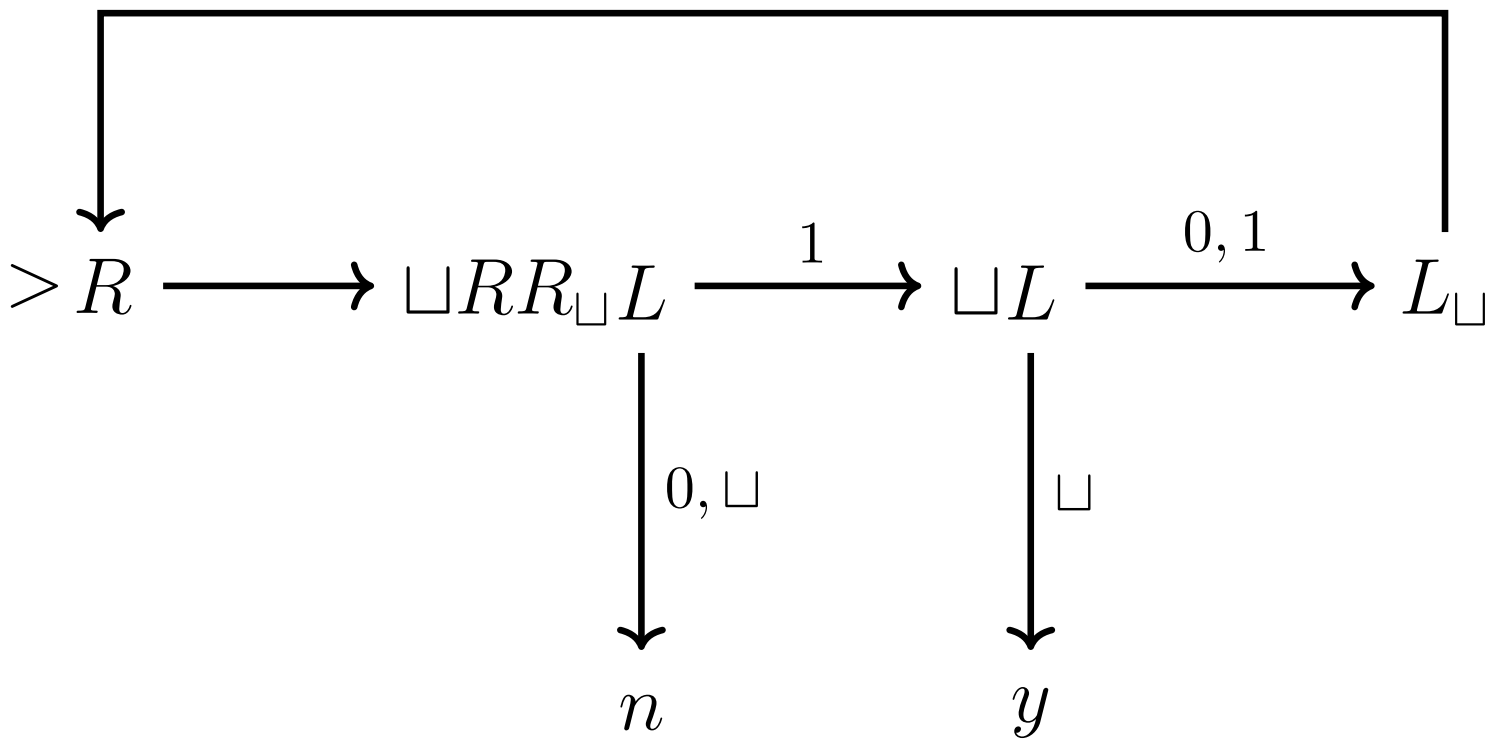
Độ phức tạp thời gian

Máy M đoán nhận $L = \{0^i 1^i \mid i = 1, 2, \ldots\}$ là $M = (\mathcal{K}, \Sigma, \delta, s, \mathcal{H})$, với $\mathcal{K} = \{s, q_0, q_1, q_2, q_3, q_4, q_5, y, n\}, \Sigma = \{0, 1, \sqcup, \triangleright\}, \mathcal{H} = \{y, n\}$ và δ được cho trong bảng sau:

$q,$	σ	$\delta(q, \sigma)$	$q,$	σ	$\delta(q, \sigma)$
s	$\sqcup$	$(q_0, \rightarrow)$	q_3	$\sqcup$	$(n, \sqcup)$
q_0	0	$(q_1, \sqcup)$	q_3	1	$(q_4, \sqcup)$
q_1	$\sqcup$	$(q_2, \rightarrow)$	q_4	$\sqcup$	$(q_5, \leftarrow)$
q_2	0	$(q_2, \rightarrow)$	q_5	$\sqcup$	$(y, \sqcup)$
q_2	1	$(q_2, \rightarrow)$	q_5	0	$(q_5, \leftarrow)$
q_2	$\sqcup$	$(q_3, \leftarrow)$	q_5	1	$(q_5, \leftarrow)$
q_3	0	$(n, 0)$	q_5	$\sqcup$	$(q_0, \rightarrow)$

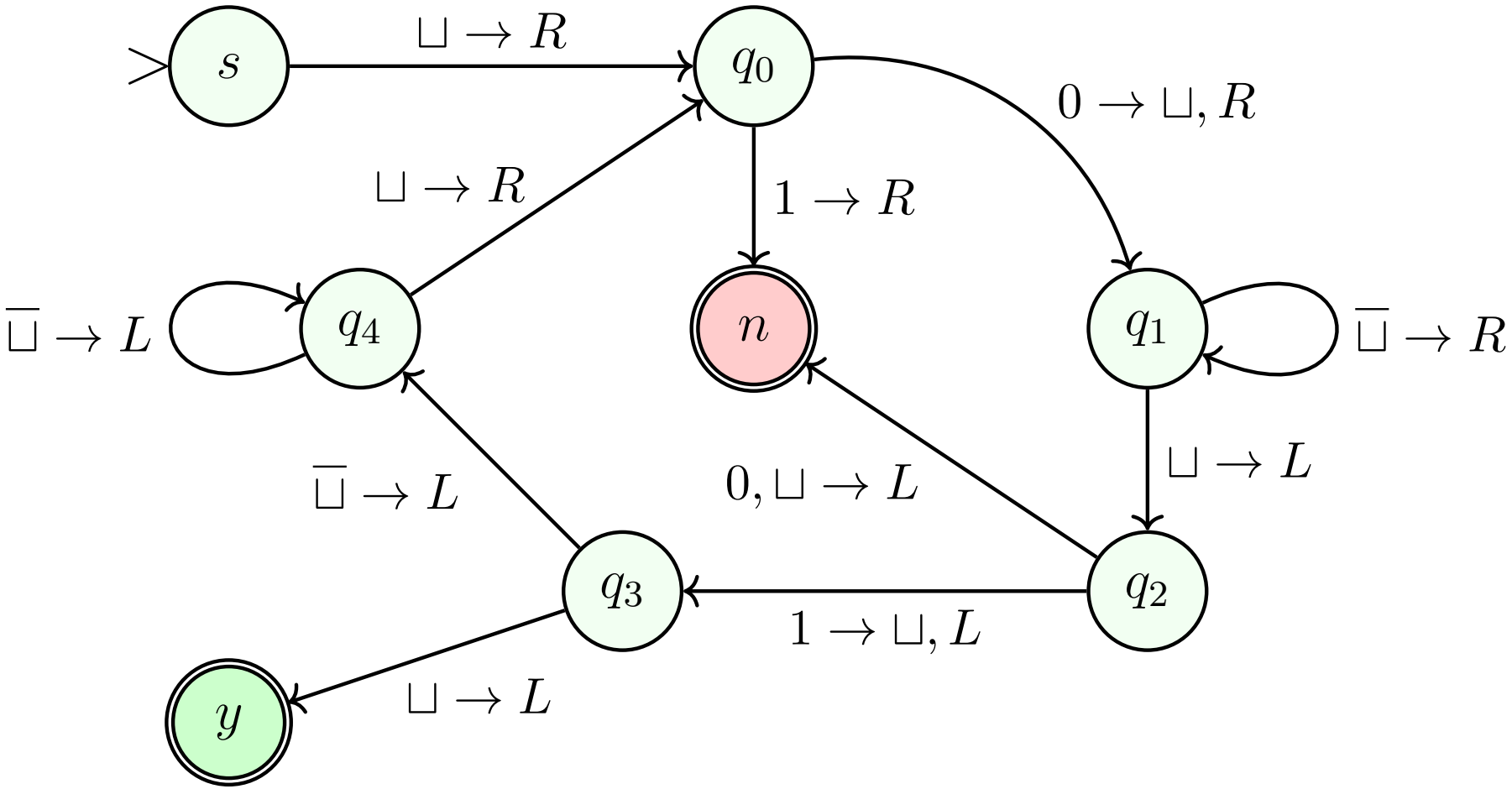
- Ta có

$$t_M(n) = \frac{(n+1)(n+2)}{2} \approx O(n^2)$$



Hình 2.6 Biểu diễn trực quan máy Turing M trong Ví dụ 2.2

Độ phức tạp thời gian



Hình 2.7 Máy Turing M trong Ví dụ 2.2 với hàm chuyển $\delta : (\mathcal{K} - \mathcal{H}) \times \Sigma \longrightarrow \mathcal{K} \times \Sigma \times \{L, R\}$

Độ phức tạp thời gian

Ví dụ 2.2 (tiếp) Giả sử máy Turing M đã cho ở Hình 2.7 khởi đầu với cấu hình $(s, \triangleright\underline{\sqcup}000111)$. Các bước tính toán của M như sau:

$$
\begin{array}{llll}
(s, \triangleright\underline{\sqcup}000111) & \vdash_M & (q_0, \triangleright \sqcup \underline{0}00111) & \vdash_M \quad (q_1, \triangleright \sqcup \sqcup \underline{0}0111) \\
& \vdash_M & (q_1, \triangleright \sqcup \sqcup 0\underline{0}111) & \vdash_M \quad (q_1, \triangleright \sqcup \sqcup 00\underline{1}11) \\
& \vdash_M & (q_1, \triangleright \sqcup \sqcup 001\underline{1}1) & \vdash_M \quad (q_1, \triangleright \sqcup \sqcup 0011\underline{1}) \\
& \vdash_M & (q_1, \triangleright \sqcup \sqcup 00111\underline{\sqcup}) & \vdash_M \quad (q_2, \triangleright \sqcup \sqcup 00111) \\
& \vdash_M & (q_3, \triangleright \sqcup \sqcup 0011)^{\dagger} & \vdash_M \quad (q_4, \triangleright \sqcup \sqcup 00\underline{1}1) \\
& \vdash_M & (q_4, \triangleright \sqcup \sqcup 0\underline{0}11) & \vdash_M^* \quad (q_4, \triangleright \sqcup \underline{\sqcup}0011) \\
& \vdash_M & (q_0, \triangleright \sqcup \sqcup \underline{0}011) & \vdash_M^* \quad (q_2, \triangleright \sqcup \sqcup \sqcup 011) \\
& \vdash_M^* & (q_3, \triangleright \sqcup \sqcup \sqcup \underline{\sqcup}) & \vdash_M \quad (y, \triangleright \sqcup \sqcup \underline{\sqcup})
\end{array}
$$

(†) Cấu hình không kết thúc với một dấu cách. Băng vào là $\triangleright \sqcup \sqcup 0011\underline{1}\sqcup$.

Bài tập 2.1 Với các máy Turing M đã xét trong Ví dụ 2.2, thực hiện các yêu cầu sau:

$a)$ Lấy ví dụ minh họa hoạt động của các máy Turing M.

$b)$ Biểu diễn máy Turing trong Hình 2.7 dưới dạng bảng chuyển.

$c)$ Hiệu chỉnh thiết kế của máy Turing trong Hình 2.7 với hàm chuyển $\delta : (\mathcal{K} - \mathcal{H}) \times \Sigma \longrightarrow \mathcal{K} \times \Sigma \times \{L, R, S\}$.

$d)$ Phân tích và so sánh độ phức tạp thời gian của các máy Turing trong Hình 2.7 và Hình 2.6.

Ví dụ 2.3 Cải tiến máy Turing M của Ví dụ 2.2 ta được máy M^1 khẳng định L trong thời gian $O(n \log_2 n)$ như sau.

Trên mỗi từ vào $\omega \in \{0,1\}^*$:

1. Loại bỏ những từ không có dạng $0^i 1^i (i, j \geq 0)$ bằng cách đọc lướt qua băng và *bác bỏ*, nếu bắt gặp kí tự 0 sau kí tự 1.
2. Trên mỗi từ vào dạng $0^i 1^i$, thực hiện qui trình lặp mỗi khi trên băng vẫn còn cả hai kí tự 0 và 1:
   - $2a$. Đọc lướt qua băng, kiểm tra xem toàn bộ số những kí tự 0 và những kí tự 1 có ở trên băng là chẵn hay lẻ. Nếu đó là số lẻ, thì *bác bỏ*.
   - $2b$. "Xóa nhảy cóc" (đánh dấu một và để lại một) đối với các kí tự 0 cũng như đối với các kí tự 1, bắt đầu từ trái qua phải.
3. Khi bắt đầu tính toán hoặc khi qui trình lặp kết thúc, nếu trên băng không còn bất cứ kí tự 0 và 1 nào thì *chấp nhận*; ngược lại *bác bỏ*.

Lớp phức tạp thời gian

Định nghĩa 2.5 Cho một hàm số $t : \mathbb{N} \longrightarrow \mathbb{R}^+$. Ta định nghĩa lớp phức tạp (complexity class) $\mathbf{TIME}(t(n))$ là lớp tất cả các ngôn ngữ khẳng định được bởi một máy Turing thời gian $O(t(n))$.

Ta có

$$\mathbf{TIME}(n) \subseteq \mathbf{TIME}(n^2) \subseteq \mathbf{TIME}(n^3) \subseteq \cdots$$

Định nghĩa 2.6 Cho một hàm số $t : \mathbb{N} \longrightarrow \mathbb{R}^+$. Ta định nghĩa lớp phức tạp $\mathbf{NTIME}(t(n))$ là lớp tất cả các ngôn ngữ khẳng định được bởi máy Turing không tất định thời gian $O(t(n))$.

Thời gian đa thức và lớp P

Định nghĩa 2.7

$$\mathbf{P} = \bigcup_k \mathbf{TIME}(n^k)$$

Nói cách khác, $\mathbf{P}$ là lớp tất cả các ngôn ngữ được khẳng định bởi máy Turing tất định thời gian đa thức.

Lớp $\mathbf{P}$ đóng vai trò trung tâm trong lý thuyết độ phức tạp tính toán và là lớp rất quan trọng, bởi vì:

- $\mathbf{P}$ không thay đổi đối với tất cả các mô hình tính toán tương đương đa thức với máy Turing tất định một băng, và
- $\mathbf{P}$ gần như tương ứng với lớp các bài toán giải được một cách thực tế, tức là giải được trên máy tính điện tử.

Bài toán về đường đi trong đồ thị có hướng - DIPATH:

DIPATH

Dữ kiện: Cho một đồ thị có hướng G và hai đỉnh u, v thuộc G.
Câu hỏi: Phải chăng trong G tồn tại đường đi có hướng từ u đến v?

Ngôn ngữ tương ứng của bài toán này được xác định bởi

$DIPATH = \{\langle G, u, v\rangle |$ G là đồ thị có hướng chứa hai đỉnh u, v và trong G tồn tại đường đi từ u đến $v\}$.

Định lý 2.1

$$DIPATH \in \mathbf{P}.$$

Một vài bài toán thuộc lớp P

Ví dụ 2.3 Sử dụng phương pháp tìm kiếm theo chiều rộng trên đồ thị, máy Turing tất định dừng M khẳng định $DIPATH$ được xây dựng như sau.

$M =$ "Trên mỗi từ vào $\langle G, u, v \rangle$, trong đó G là đồ thị có hướng chứa hai đỉnh u và v:

1. Đánh dấu đỉnh u.
2. Thực hiện một qui trình lặp sau đây cho đến khi không còn đỉnh nào chưa được đánh dấu:
   - ⋆ Xem xét tất cả các cung của G với đỉnh đầu đã được đánh dấu. Khi cung (x, y) với đỉnh đầu x đã được đánh dấu và đỉnh cuối y chưa được đánh dấu, đánh dấu đỉnh y.
3. Nếu v được đánh dấu thì *chấp nhận*; ngược lại *bác bỏ*".

- Tổng số bước mà M cần thực hiện không vượt quá $1 + m$, tức là một đa thức theo số đỉnh của đồ thị.

Bài toán RELPRIME:

Giả sử RELPRIME là bài toán đòi hỏi kiểm tra xem hai số nguyên dương bất kì cho trước có phải là hai số nguyên tố cùng nhau hay không. Khi đó, ngôn ngữ tương ứng của bài toán này được xác định bởi

$$RELPRIME = \{\langle x, y \rangle | \, x \text{ và } y \text{ nguyên tố cùng nhau}\}.$$

Định lý 2.2

$$RELPRIME \in \mathbf{P}.$$

Lớp NP và ngôn ngữ kiểm chứng nhanh

Định nghĩa 2.8

$$\mathbf{NP} = \bigcup_k \mathbf{NTIME}(n^k)$$

Nói cách khác, $\mathbf{NP}$ là lớp tất cả các ngôn ngữ được khẳng định bởi máy Turing không tất định thời gian đa thức.

Lớp $\mathbf{NP}$ là một trong vài lớp phức tạp quan trọng nhất của lý thuyết độ phức tạp tính toán. Ta cần lưu ý:

- Lớp $\mathbf{NP}$ không phụ thuộc vào cách lựa chọn mô hình tính toán không tất định thích hợp (một băng hay nhiều băng) vì các mô hình ấy đều tương đương đa thức với nhau.
- Việc diễn tả và phân tích các thuật toán không tất định thời gian đa thức tương tự cách ta đã làm và phân tích nhằm để chứng tỏ rằng mỗi nhánh tính toán của thuật toán sử dụng không quá một số đa thức các phép biến đổi cơ bản.

Lớp NP và ngôn ngữ kiểm chứng nhanh

Định nghĩa 2.9 Cho L là một ngôn ngữ trên bảng chữ Σ. Ta nói rằng ngôn ngữ L là **kiểm chứng được trong thời gian đa thức** (polynomial time verifiable), hay đơn giản là **kiểm chứng được nhanh** (quickly verifiable), nếu tồn tại một máy Turing tất định thời gian đa thức V và một tập C nào đó sao cho, đối với mọi từ $\omega \in \Sigma^*$,

$$\omega \in L \Longleftrightarrow \exists c \in C : V \text{ chấp nhận } \langle \omega, c \rangle,$$

và hơn thế nữa, máy V chấp nhận $\langle \omega, c \rangle$ trong thời gian đa thức chỉ theo độ dài của từ ω. Khi đó, V được gọi là **máy kiểm chứng thời gian đa thức** (polynomial time verifier) đối với ngôn ngữ L, mỗi phần tử của tập C được gọi là **chứng cứ** (certificate), còn chứng cứ $c \in C$ mà theo đó V chấp nhận $\langle \omega, c \rangle$ được gọi là **bằng chứng** (proof) để khẳng định rằng ω là thành viên của L.

Lớp NP và ngôn ngữ kiểm chứng nhanh

Định lý 2.3 Cho L là một ngôn ngữ trên bảng chữ Σ. Khi ấy

$$L \in \mathbf{NP} \Longleftrightarrow L \text{ là kiểm chứng được trong thời gian đa thức.}$$

Ví dụ một số bài toán thuộc lớp $\mathbf{NP}$:

1. $DIHAMPATH = \{\langle G, u, v\rangle | \, G$ là một đồ thị có hướng chứa đường đi Hamilton từ u đến $v\}$.
2. $SUBSETSUM = \{\langle S, t\rangle | \, S = \{x_1, \ldots, x_m\}$ và có một họ con nào đó $\{y_1, \ldots, y_k\} \subseteq S$, sao cho $\sum y_i = t\}$.
3. $SAT = \{\langle \Phi\rangle | \, \Phi$ là công thức Boole thỏa được$\}$.

Bài toán "P vs NP"

- Câu hỏi $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ là bài toán lớn chưa được giải quyết của khoa học tính toán cũng như của toán học hiện đại.
- Về tương quan giữa hai lớp $\mathbf{P}$ và $\mathbf{NP}$, có thể xảy ra hai khả năng $\mathbf{P} = \mathbf{NP}$ hoặc $\mathbf{P} \neq \mathbf{NP}$.
- Bài toán $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ được Viện Toán học Clay chọn là một trong 7 bài toán của thế kỷ XXI và **treo thưởng 1 triệu đô la Mỹ** cho ai tìm được lời giải.
- Những thuật toán tốt nhất được biết đến cho đến hôm nay đối với các bài toán thuộc $\mathbf{NP}$ đều đòi hỏi thời gian không quá hàm mũ, tức là

$$\mathbf{NP} \subseteq \mathbf{E} = \bigcup_k \mathbf{TIME}(2^{n^k}),$$

nhưng ta vẫn chưa rõ liệu $\mathbf{NP}$ có thể chứa trong một lớp phức tạp thời gian tất định bé hơn không.

Định lý 2.4 (Cook-Levin)

$$SAT \in \mathbf{P} \Longleftrightarrow \mathbf{P} = \mathbf{NP}.$$

Định nghĩa 2.10 Cho một hàm $f : X^* \to X^*$. Ta nói rằng f là hàm **tính được trong thời gian đa thức** (polynomial time computable function), nếu nó là hàm tính được bởi máy Turing thời gian đa thức.

Định nghĩa 2.11 Cho hai ngôn ngữ A và B trên bảng chữ Σ. Ta nói rằng ngôn ngữ A qui dẫn được trong thời gian đa thức (polynomial time reducible) đến ngôn ngữ B, ta viết $A \preceq_P B$ nếu tồn tại một hàm tính được trong thời gian đa thức $f : \Sigma^* \to \Sigma^*$ sao cho, đối với mỗi $\omega \in \Sigma^*$,

$$\omega \in A \Longleftrightarrow f(\omega) \in B.$$

Hàm f được gọi là phép qui dẫn thời gian đa thức (polynomial time reduction) từ A đến B.

Độ phức tạp không gian

Xét một máy Turing tất định một băng dừng với bảng chữ vào Σ, trạng thái ban đầu q_0 và các trạng thái kết thúc y, n. Khi đó quá trình tính toán của M trên mỗi từ vào $\omega \in \Sigma^*$ là một quá trình chuyển đổi các cấu hình của máy, kể từ cấu hình ban đầu $C_0 = q_0\omega$ đến cấu hình kết thúc C_k:

$$C_0 \vdash_M C_1 \vdash_M \cdots \vdash_M C_t \vdash_M C_{t+1} \vdash_M \cdots \vdash_M C_{k-1} \vdash_M C_k,$$

Trong đó $C_t = u_t q_t v_t$ là cấu hình của M tại thời điểm t trong quá trình tính toán trên ω, $(t = 0, 1, \ldots, k)$.

- Ta kí hiệu $s_M(\omega, t)$ và $s_M(\omega)$ lần lượt là số ô trên băng tại thời điểm t và số ô trên băng tối đa mà M sử dụng tại mỗi thời điểm. Ta có $s_M(\omega, t) = |u_t v_t|$, $s_M(\omega) = \max\{s_M(\omega, t) | t = 0, 1, \ldots, k\}$.
- Khi đó độ phức tạp không gian $s_M(n)$ của M được xác định bởi

$$s_M(n) = \max\{s_M(\omega) | \omega \in \Sigma^n\}.$$

Độ phức tạp không gian

Định nghĩa 2.12 Cho M là một máy Turing tất định dừng. **Độ phức tạp không gian** (space complexity) của M là hàm $s : \mathbb{N} \longrightarrow \mathbb{N}$, mà giá trị $s(n)$ là số tối đa các ô trên băng mà M sử dụng trong quá trình nó tính toán trên bất cứ từ vào nào độ dài n. Khi độ phức tạp không gian của M là $s(n)$ thì ta nói rằng M hoạt động trong không gian $s(n)$ hay M là máy Turing không gian $s(n)$.

Định nghĩa 2.13 Cho một máy Turing không tất định dừng N. **Độ phức tạp không gian của máy Turing không tất định** N là hàm $s : \mathbb{N} \longrightarrow \mathbb{N}$, mà giá trị $s(n)$ là số tối đa các ô trên băng được sử dụng trong quá trình tính toán của N (theo nhánh bất kỳ trong cây tính toán của máy) trên bất cứ từ vào nào độ dài n.

Độ phức tạp không gian

Định nghĩa 2.14 Cho một hàm số $s : \mathbb{N} \longrightarrow \mathbb{R}^+$. Các lớp phức tạp không gian (space complexity class) $\mathbf{SPACE}(s(n))$ và $\mathbf{NSPACE}(s(n))$ được xác định như sau.

$$\mathbf{SPACE}(s(n)) = \{L \mid L \text{ là ngôn ngữ được khẳng định bởi máy Turing tất định không gian } O(s(n))\}.$$

$$\mathbf{NSPACE}(s(n)) = \{L \mid L \text{ là ngôn ngữ được khẳng định bởi máy Turing không tất định không gian } O(s(n))\}.$$

TRÂN TRỌNG CẢM ƠN!