TOÁN RỜI RẠC

Discrete Mathematics

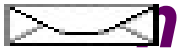
Fall 2009

NguyÔn §øc NghÜa

Bé m«n Khoa häc M¸y tÝnh

§¹i häc B¸ch khoa Hµ néi

Tel: 0438696121 (Off), 0903210111 (Mob)



nghiand@it-hut.edu.vn

Đề nghị với các lớp trưởng

- Hãy gửi cho tôi danh sách lớp theo địa chỉ email đã nêu

Toán rời rạc là gì?

- **What is discrete mathematics?**
  - Là bộ phận của toán học nghiên cứu các đối tượng rời rạc.
    - Rời rạc bao hàm ý các phần tử phân biệt hay không liên tục.
    - Các phép toán:
      - Tổ hợp: Đếm các đối tượng rời rạc
      - Các phép toán logic, quan hệ: nói lên mối quan hệ giữa các đối tượng rời rạc
    - Làm việc với: Các đối tượng rời rạc: tập hợp, mệnh đề.

Định nghĩa hình thức - Wikipedia

- **Discrete mathematics**, sometimes called **finite mathematics**, is the study of mathematical structures that are fundamentally discrete, in the sense of not supporting or requiring the notion of continuity. Most, if not all, of the objects studied in finite mathematics are countable sets, such as the integers.

- Discrete mathematics has become popular in recent decades because of its applications to computer science. Concepts and notations from discrete mathematics are useful to study or express objects or problems in computer algorithms and programming languages. In some mathematics curricula, finite mathematics courses cover discrete mathematical concepts for business, while discrete mathematics courses emphasize concepts for computer science majors.

- Discrete mathematics usually includes :
  - logic - a study of reasoning
  - set theory - a study of collections of elements
  - number theory
  - combinatorics - a study of counting
  - graph theory
  - algorithmics - a study of methods of calculation
  - information theory
  - the theory of computability and complexity - a study on theoretical limitations on algorithms …

Nhập môn Toán rời rạc

- Các ứng dụng của TRR:
  - Formal Languages (computer languages)
  - Machine translation
  - Compiler Design
  - Artificial Intelligence
  - Relational Database Theory
  - Network Routing
  - Algorithm Design
  - many more (almost all areas of computer science)…

A building block of computer science !

Nhập môn Toán rời rạc

- Các vấn đề chính được đề cập trong giáo trình này:
  - Cơ sở: logic, tập hợp, ánh xạ.
  - Lý thuyết tổ hợp (Combinatorial Theory)
    - Bài toán đếm
    - Bài toán tồn tại
    - Bài toán liệt kê
    - Bài toán tối ưu
  - Lý thuyết đồ thị (Graph theory):
    - Đồ thị, Đường đi, Liên thông
    - Biểu diễn đồ thị
    - Duyệt đồ thị
    - Các bài toán tối ưu trên đồ thị

Tài liệu tham khảo

1. **Rosen K.H.** *Discrete Mathematics and its Applications*. McGraw - Hill Book Company, 2003.

2. **Johnsonbaugh R.** *Discrete Mathematics.* Prentice Hall Inc., N. J., 1997.

3. **Grimaldi R.P.** *Discrete and Combinatorial Mathematics (an Applied Introduction),* Addison-Wesley, 5th edition, 2004.

4. **R. Graham, O. Patashnik, and D.E. Knuth.** *Concrete Mathematics*, Second Edition. Addison-Wesley, 1994.

Tài liệu tham khảo

5. **Phan Đình Diệu.** *Lý thuyết ôtômat hữu hạn và thuật toán*. NXB ĐHTHCN, Hà nội, 1977.
6. **Nguyễn Hữu Anh.** *Toán rời rạc*, NXB Giáo dục,1999.
7. **Nguyễn Xuân Quỳnh.** *Cơ sởToán rời rạc và ứng dụng*. NXB KHKT, Hà nội, 1996.
8. **Đỗ Đức Giáo.** *Toán rời rạc*. NXB KHKT, Hà nội, 2001.
9. **Hoàng Chúng.** *Đại cương về toán hữu hạn*. NXB Giáo dục, 1997.

Rosen's Book

http://www.mhhe.com/math/advmath/rosen/index.mhtml

Rosen K.H.

Discrete Mathematics and its Applications. 5th Edition,

McGraw - Hill Book Company, 2003.

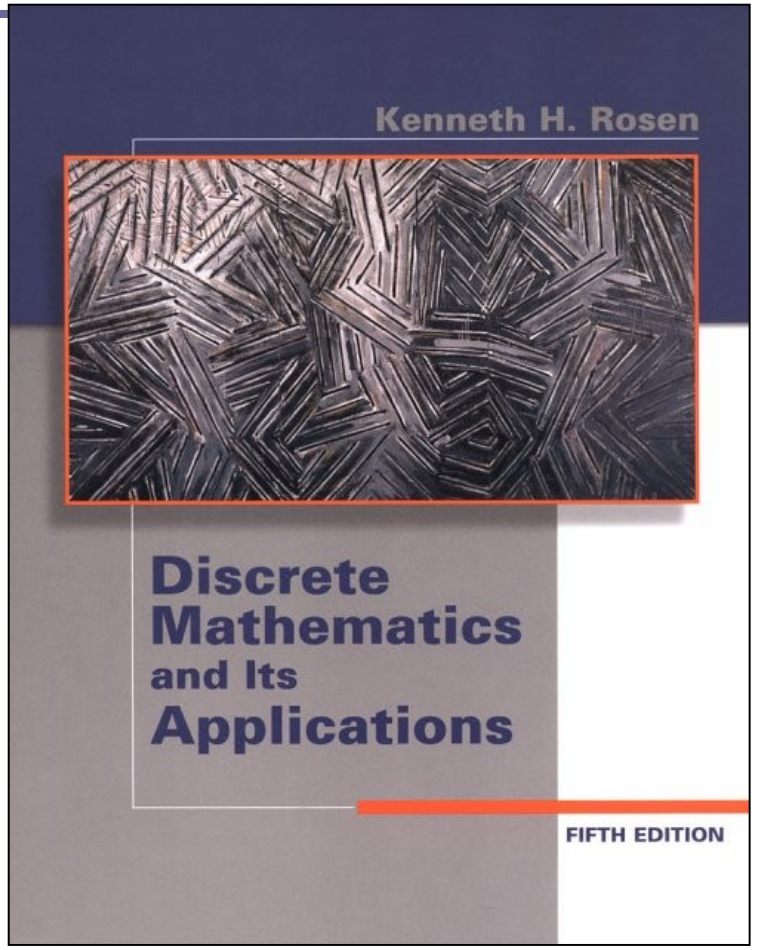


Table of Contents

Johnsonbaugh Book

Johnsonbaugh R.

Discrete Mathematics.

Prentice Hall Inc.,

N. J., 1997.

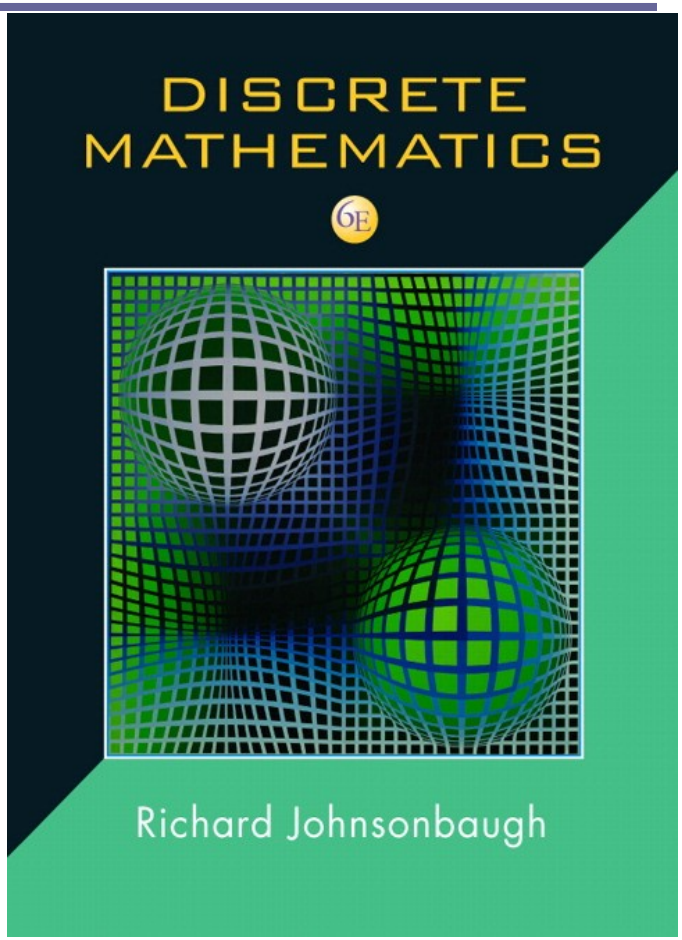


Table of Contents

Grimadi's Book

Grimaldi R.P.
Discrete and Combinatorial Mathematics (an Applied Introduction),
Addison-Wesley, 5th edition, 2001.

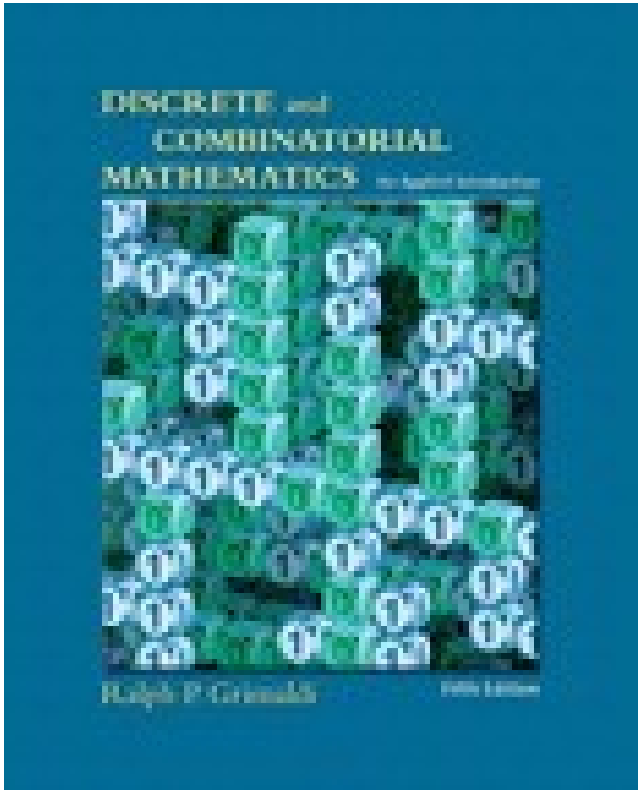


Table of Contents

Graham, Knuth, Patashnik's Book

Ronald L. Graham

Donald E. Knuth

Oren Patashnik

Concrete Mathematics:

A Foundation for Computer Science,

Addison-Wesley Professional

1994, 672 pp

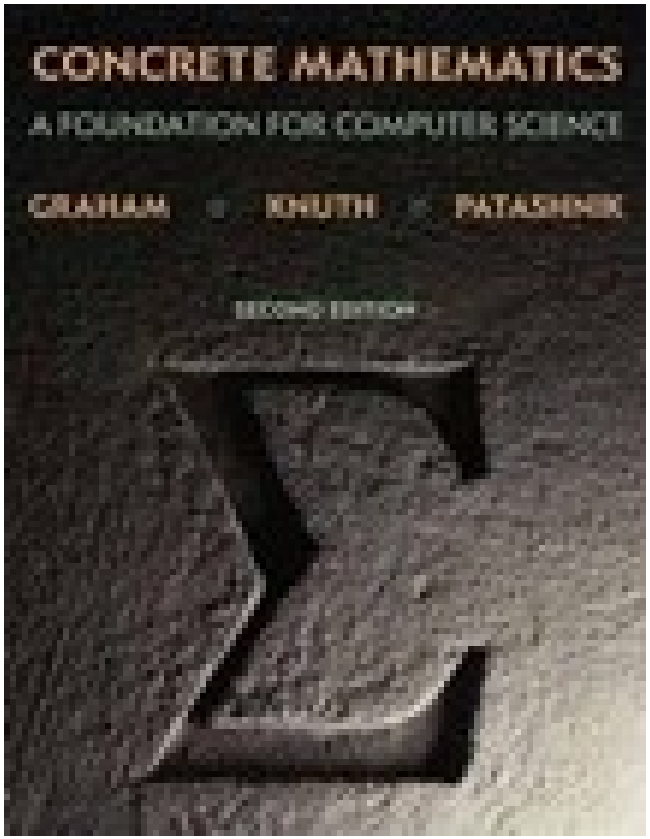


Table of Contents

Tài liệu tham khảo chính

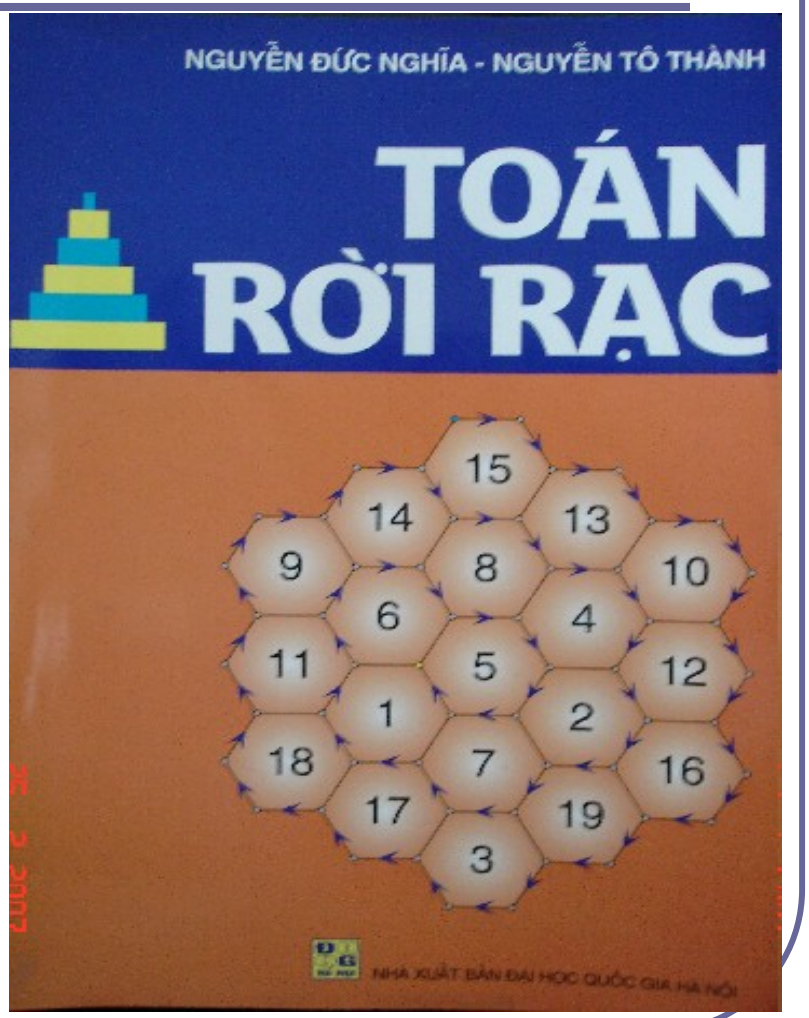
Nguyễn Đức Nghĩa,

Nguyễn Tô Thành

TOÁN RỜI RẠC

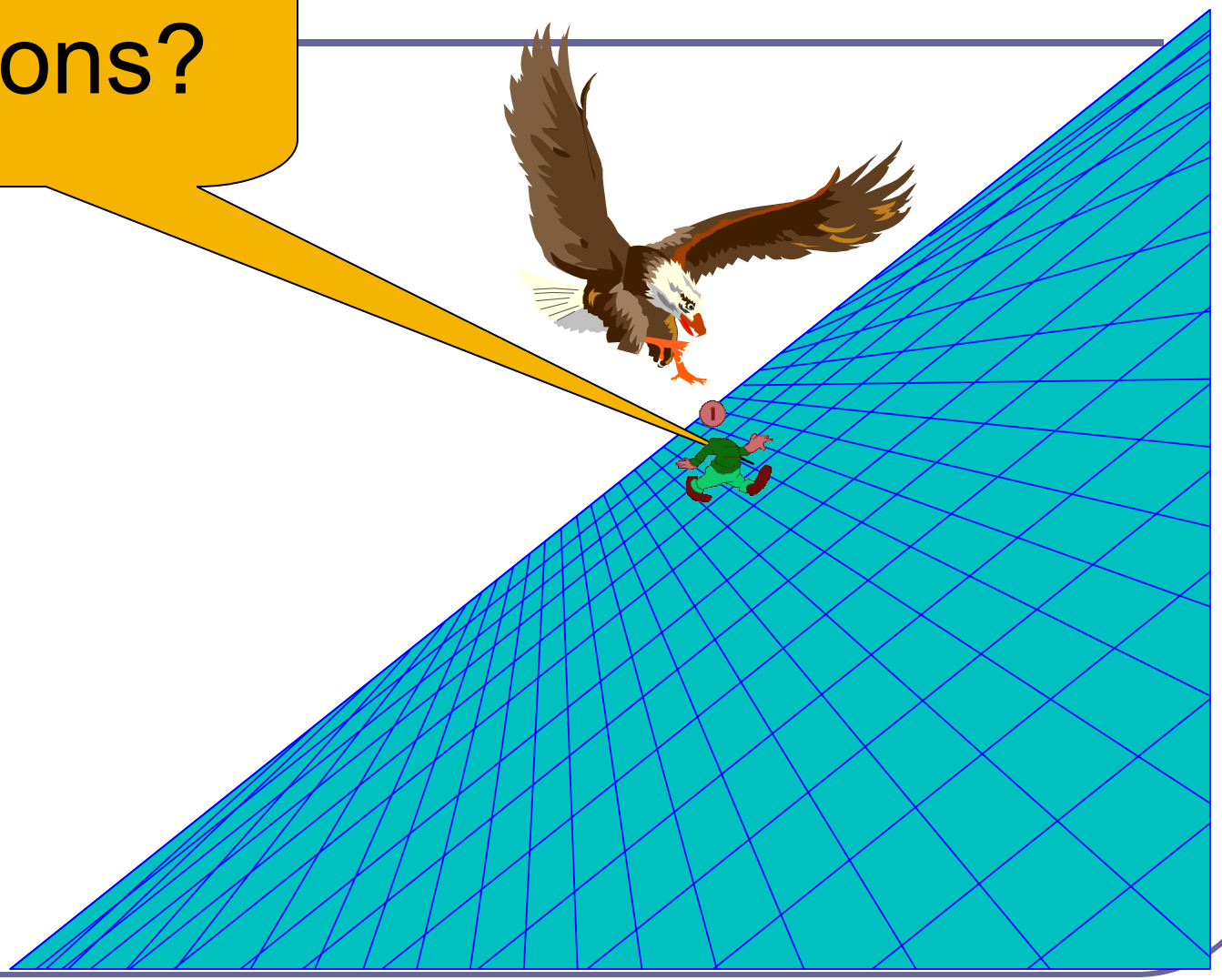
(in lần thứ ba)

Nhà xuất bản Đại học Quốc gia Hà nội, 2003, 290 trang



Mục lục

Quetions?

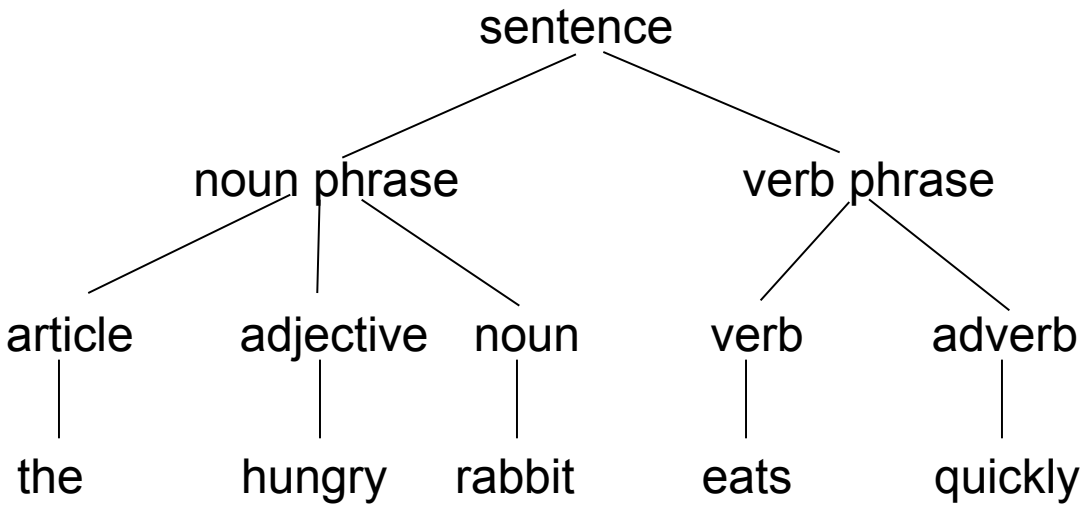


Ứng dụng: Ngôn ngữ hình thức Formal Language

- Formal language:
  - Ngôn ngữ được sinh bởi ngữ pháp (grammars)
  - Grammars:
    - Sinh ra các từ (words) của ngôn ngữ
    - Xác định một từ có thuộc ngôn ngữ hay không
  - Từ (Words):
    - Có thể tổ hợp bằng nhiều cách
    - Ngữ pháp cho biết tổ hợp từ có là một câu hợp lệ (valid sentence) hay không
  - Ứng dụng:
    - Thiết kế các Ngôn ngữ lập trình và Chương trình dịch (Programming Languages and Compilers)

Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 1:
  - Ngôn ngữ tự nhiên: English
    - Có phải “the hungry rabbits eats quickly” là câu tiếng Anh?
    - Cây cú pháp của câu (Derivation tree of the sentence):



Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 2:
  - Bài toán điển hình trong xây dựng Chương trình dịch.
  - Xác định từ *cbab* có thuộc ngôn ngữ sinh bởi ngữ pháp G = (V, T, S, P), trong đó:
    - V = { a, b, c, A, B, C, S }
    - T = { a, b, c }
    - S là ký tự đầu tiên (starting symbol)
    - P là các luật sản xuất (productions):

$S \rightarrow AB$ $A \rightarrow Ca$ $B \rightarrow Ba$

$B \rightarrow Cb$ $B \rightarrow b$ $C \rightarrow cb$

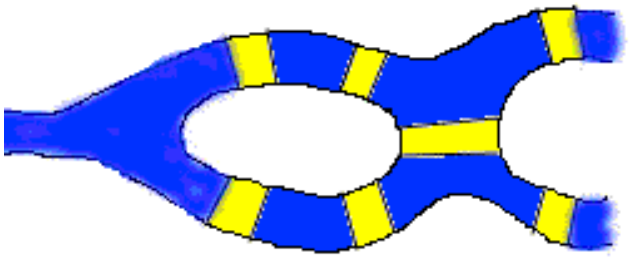
$C \rightarrow b$

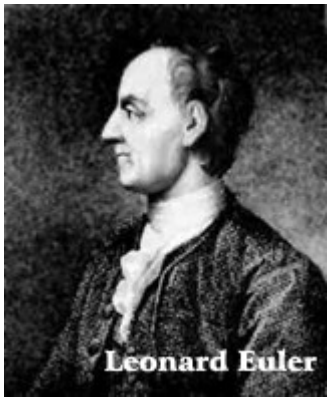
Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 2 (tiếp tục):
  - Giải:
    - Bắt đầu bởi S và tìm cách dẫn ra *cbab* sử dụng dãy các luật sản xuất.
      - Do chỉ có một luật bắt đầu vớiS, ta phải bắt đầu với
        $$S \Rightarrow AB$$
      - Sử dụng luật đối với A để thu được:
        $$S \Rightarrow AB \Rightarrow CaB$$
      - Sử dụng luật đối với $C \rightarrow cb$ vì *cbab* bắt đầu bởi cb:
        $$S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB$$
      - Cuối cùng sử dụng luật đối với $B \rightarrow b$:
        $$S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB \Rightarrow cbab$$

Ứng dụng: Graph Theory

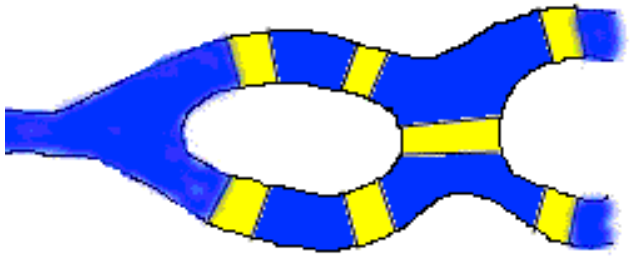
- **Ví dụ 3**
  - Bài toán về 7 cái cầu (Konigsberg 7-bridge problem)
    - Konigsberg là thành phố của Nga
    - Có 4 vùng đất, và 7 cái cầu nối chúng
    - Euler giải được bài toán năm 1736; là khởi nguồn của lý thuyết đồ thị





Ứng dụng: Graph Theory

- Bài toán: Vẽ đường đi (hoặc vòng kín) bởi bút chì sao cho có thể đi qua mỗi cái cầu đúng một lần mà không được nhấc đầu bút khỏi mặt giấy (vẽ một nét)



Ứng dụng: Lý thuyết tập hợp (Set Theory)

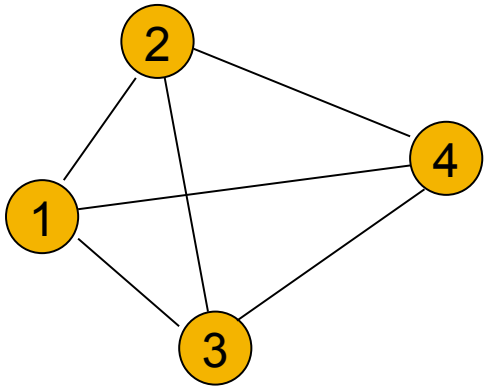
- Có phải số lượng số nguyên là nhiều hơn số lượng số nguyên dương?
- Có phải số lượng số nguyên là nhiều hơn số lượng số thực?

Ứng dụng: Lý thuyết độ phức tạp (Complexity Theory)

- Ví dụ 5:
  - Bài toán người du lịch (The Traveling Salesman Problem)
  - Có ứng dụng quan trọng trong
    - Thiết kế mạch (circuit design)
    - Hướng lộ trên mạng (network routing)
    - và nhiều bài toán trong tin học khác
  - Cho:
    - n thành phố $c_1, c_2, \ldots, c_n$
    - khoảng cách giữa thành phố i và j là d_{ij}
  - Tìm hành trình ngắn nhất.

Ứng dụng: Lý thuyết độ phức tạp

- Ví dụ 5 (tiếp):



  - Có bao nhiêu hành trình khác nhau?
    - Thành phố đầu tiên có thể chọn bởi n cách,
      - thành phố thứ hai, n-1 cách,
      - thành phố thứ ba, n-2 cách,
      - v.v...
    - # hành trình = n (n-1) (n-2) . . . .(2) (1) = n! (Tổ hợp)
  - Tính độ dài của một hành trình đòi hỏi n-1 phép cộng.
    - Tổng số phép cộng = (n-1) × n! (Qui tắc nhân - Rule of Product)

Ứng dụng: Lý thuyết độ phức tạp

- Ví dụ 5 (tiếp):
  - Giả sử có PC tốc độ rất cao:
    - 1 GHz = 1,000,000,000 ops/sec

      ⇒ 1 flop = 1 nanosecond

      = 10^{-9} sec.
    - Nếu n=8, T(n) = 7•8! = 282,240 flops << 1 second.
    - TUY NHIÊN . . . . . .
    - Nếu n=50, T(n) = 49•50! = 1.48 10^{66}

      = 1.49 10^{57} seconds

      = 4.73 10^{49} năm.
    - ... quá lâu. Không ai trong số chúng ta có thể chờ được thời điểm kết thúc.
    - Có rất nhiều bài toán mà chúng ta còn chưa biết liệu có thuật toán hiệu quả để giải hay không!

Ứng dụng: Lý thuyết độ phức tạp

- Ví dụ 5 (tiếp):
  - Chúng ta có thể làm gì ?
    - Không có thời gian để chờ Do not waste time unless you are a genius to save the world
    - Mục đích khiêm tốn hơn
      - Với xác suất 90%, có thể tìm được hành trình tối ưu
      - Thuật toán tìm hành trình không tồi hơn 1.1 lần hành trình tối ưu