BÀI 2: BÀI TOÁN ĐẾM VÀ BÀI TOÁN TỒN TẠI TỔ HỢP

Giới thiệu

Bài này giới thiệu những nét chính của lý thuyết tổ hợp bao gồm đối tượng nghiên cứu, một số tên gọi, thuật ngữ, ứng dụng và một số vấn đề mà lý thuyết tổ hợp đề ra, sau đó chủ yếu trình bày nội dung hai bài toán của lý thuyết tổ hợp là bài toán đếm và bài toán tồn tại.

Nội dung

- Giới thiệu về lý thuyết tổ hợp
- Bài toán đếm
- Bài toán tồn tại

Thời lượng học

- 12 tiết

Mục tiêu

Sau khi học bài này, các bạn có thể:

- Nắm được một số cấu hình cơ bản và các bài toán của lý thuyết tổ hợp.
- Sử dụng các nguyên lý cơ bản và các kỹ thuật đếm cơ bản trong việc giải quyết bài toán đếm.
- Sử dụng các nguyên lý cơ bản và các phương pháp trong việc giải quyết bài toán tồn tại.
- Biết cách sử dụng lập trình trong việc giải quyết bài toán đếm và bài toán tồn tại.

TÌNH HUỐNG DẪN NHẬP

Tình huống: Bài toán xếp khách của Lucas

Có một bàn tròn, xung quanh có 2n ghế. Cần sắp chỗ cho n cặp vợ chồng sao cho các ông ngồi xen kẽ với các bà và không có cặp vợ chồng nào ngồi cạnh nhau. Đây chính là bài toán xếp khách của François-Édouard-Anatole Lucas - Pháp (1842-1891).

Câu hỏi

Hỏi có bao nhiêu cách xếp khách thỏa mãn yêu cầu đề ra?

Bài này giới thiệu những nét chính của lý thuyết tổ hợp bao gồm đối tượng nghiên cứu, một số tên gọi, thuật ngữ, ứng dụng và một số vấn đề mà lý thuyết tổ hợp đề ra, sau đó chủ yếu trình bày nội dung hai bài toán của lý thuyết tổ hợp là bài toán đếm và bài toán tồn tại.

2.1. Giới thiệu về lý thuyết tổ hợp

2.1.1. Vài nét về lịch sử

Tư duy về tổ hợp được xuất hiện từ rất sớm trong lịch sử phát triển nhân loại qua một số bài toán cổ và những hình vẽ còn để lại, tuy nhiên lý thuyết tổ hợp được xem hình thành như một ngành toán học, vào quãng thế kỷ 17 bằng một loạt các công trình nổi tiếng của các nhà toán học xuất sắc như Pascal, Fermat, Leibnitz, Euler, ... và được phát triển mạnh mẽ, đặc biệt sau khi máy tính điện tử ra đời. Hiện nay lý thuyết tổ hợp được áp dụng trong nhiều lĩnh vực khác nhau như lý thuyết số, hình học hữu hạn, biểu diễn nhóm, đại số không giao hoán, quá trình ngẫu nhiên, lý thuyết xác suất, lý thuyết mật mã, quy hoạch thực nghiệm, ...

Lý thuyết tổ hợp nghiên cứu các luật phân bố phần tử của một tập hợp (thường là hữu hạn) theo những điều kiện nào đấy. Kết quả của những luật này là hình thành nên những nhóm phần tử khác nhau mà ta gọi chung là những *cấu hình tổ hợp* (gọi ngắn gọn là cấu hình).

Do sự phong phú của các luật phân bố được áp dụng trên nhiều đối tượng nên cấu hình tổ hợp rất đa dạng mà ta có thể thấy trong nhiều lĩnh vực hoạt động của con người: một thế cờ, một nhóm quân bài, một cách xếp hình, một lịch làm việc, một mạch điện, một công thức hóa học, một mạng máy tính, một phương án sản xuất, ... đều là những hình ảnh cụ thể của các cấu hình tổ hợp.

Sau đây trình bày một số cấu hình đơn giản nhất, chúng được dùng như những cấu hình cơ bản vì thường gặp trong thực tế.

2.1.2. Một số cấu hình cơ bản

- **Chỉnh hợp**

Khái niệm: Xét một tập hợp gồm n phần tử, từ tập này, ta xây dựng những bộ *có thứ tự* gồm m thành phần, trong đó mỗi thành phần là một phần tử nào đó của tập đang xét sao cho các phần tử *không được chọn lặp lại*. Mỗi bộ như thế được gọi là một *chỉnh hợp chập m của n phần tử*.

Ví dụ:

Ta có 12 chỉnh hợp chập 2 của 4 giá trị {1, 2, 3, 4} là (1, 2), (1, 3), (1, 4), (2, 1), (2, 3), (2, 4), (3, 1), (3, 2), (3, 4), (4, 1), (4, 2), (4, 3).

Chú ý: Điều kiện "có thứ tự" của chỉnh hợp có nghĩa là nếu hoán đổi giá trị (khác nhau) của 2 thành phần nào đó trong một chỉnh hợp thì ta nhận được một chỉnh hợp khác, chẳng hạn (1, 2) và (2, 1) là hai chỉnh hợp khác nhau.

Trong nhiều ứng dụng, việc chọn giá trị các thành phần của chỉnh hợp cho phép lặp lại (miễn là vẫn lấy trên tập giá trị được xét), khi đó chỉnh hợp được gọi là *chỉnh hợp lặp* để nhấn mạnh việc được *lặp lại giá trị* của mỗi thành phần.

Trong ví dụ trên, số các chỉnh hợp lặp sẽ là 16 (thêm 4 chỉnh hợp có lặp là (1, 1), (2, 2), (3, 3), (4, 4)).

Chú ý: Trong chỉnh hợp (không được lặp) số chập m (còn được gọi là độ dài của chỉnh hợp) không được lớn hơn số các giá trị n mà các thành phần có thể chọn, còn trong các chỉnh hợp lặp, m và n có thể lớn bé hơn nhau tùy ý.

Chỉnh hợp lặp được gặp trong khá nhiều ứng dụng.

Ví dụ: Để phân biệt các đối tượng được quản lý, người ta mã hóa mỗi đối tượng bằng một chuỗi ký hiệu (với độ dài cho trước) lấy từ một bảng (hữu hạn) các ký hiệu nào đấy, trong đó các ký hiệu trong chuỗi mã có thể trùng nhau (số báo danh, mã số thuế, số chứng minh thư, số đăng ký xe, ...).

Chúng là những chỉnh hợp lặp trên tập ký hiệu được xét. Điều kiện "không được lặp" phát sinh từ yêu cầu giá trị của các thành phần trong chỉnh hợp phải khác nhau, chẳng hạn một cách đặt tên cho m đối tượng (hai đối tượng khác nhau phải có tên khác nhau) chọn từ một bảng gồm n tên nào đấy, là một chỉnh hợp (không lặp) chập m của n tên.

- **Hoán vị**

  Khái niệm: Ta gọi một *hoán vị của n phần tử* là một cách xếp thứ tự của n phần tử đó.

  Ví dụ: với 3 phần tử {1, 2, 3} ta có 6 hoán vị sau: (1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1). Có thể thấy hoán vị của n phần tử chính là một chỉnh hợp chập n của n phần tử đang xét. Một lịch thực hiện n công việc là một hoán vị của n công việc này, sự thay đổi thứ tự thực hiện các công việc có tầm ảnh hưởng rất lớn đến chất lượng của các công việc, vì thế bài toán tìm một lịch tối ưu là một bài toán có ý nghĩa quan trọng trong thực tiễn.

- **Tổ hợp**

  Khái niệm: Ta gọi một *tổ hợp chập m của n phần tử* là một cách lấy ra *m* phần tử *không kể thứ tự* từ một tập *n* phần tử, nói khác đi, nó là một *tập con* m phần tử của một tập n phần tử (vì thế $m \leq n$).

  Có thể định nghĩa một tổ hợp chập *m* của *n* như một chỉnh hợp chập *m* của *n* trong đó thay điều kiện "*có thứ tự*" trong chỉnh hợp bằng điều kiện "*không kể thứ tự*" trong tổ hợp. Từ điều kiện này suy ra, các chỉnh hợp, chỉ khác nhau về thứ tự, là tương ứng với một tổ hợp.

  Chẳng hạn ta có 12 chỉnh hợp chập 2 của 4 giá trị {1, 2, 3, 4} (xem ví dụ trong 2.1.2.1) nhưng chỉ có 6 tổ hợp chập 2 của các giá trị này (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4) vì hai chỉnh hợp chỉ khác nhau về thứ tự được tính là một tổ hợp. Các tổ hợp liên quan đến những bài toán chia nhóm, trong đó không quan tâm đến thứ tự các thành viên, chẳng hạn cách lập một tốp ca của một lớp học sinh, cách rút một số quân bài từ một cỗ bài, cách chọn hai đội bóng để đấu, ... là những ví dụ về tổ hợp.

2.1.3. Các bài toán của lý thuyết tổ hợp

Người ta thường phân loại các bài toán của lý thuyêt tổ hợp theo bốn dạng dưới đây:

- **Bài toán đếm**

  Đây là các bài toán nhằm trả lời câu hỏi "có bao nhiêu cấu hình thỏa mãn điều kiện đã nêu?".

  Phương pháp đếm thường dựa vào một số nguyên lý cơ bản và một số kết quả đếm các cấu hình đơn giản. Ngoài việc rèn luyện các tư duy ban đầu về sự hình thành các cấu hình, bài toán đếm còn được dùng trong các công việc mang tính chất đánh giá như tính xác suất của một sự kiện, tính độ phức tạp của một thuật toán,...

- **Bài toán tồn tại:**

  Nếu trong bài toán đếm sự tồn tại của các cấu hình là hiển nhiên thì trong bài toán này, vấn đề "có hay không có" cấu hình là điều cần giải quyết.

  Bài toán tồn tại thường bị kẹt vào tình thế lưỡng nan: không chỉ ra được cấu hình nào nhưng cũng không khẳng định được chúng không tồn tại. Lịch sử toán học thường để lại những bài toán khó trong lĩnh vực này và việc cố gắng giải chúng đã thúc đẩy không ít sự phát triển của nhiều ngành toán học.

- **Bài toán liệt kê:**

  Bài toán này quan tâm đến tất cả các cấu hình có thể có, vì thế lời giải của nó cần được biểu diễn dưới dạng thuật toán "vét cạn" tất cả các cấu hình. Lời giải trong từng trường hợp cụ thể sẽ được máy tính giải quyết nhờ chạy một chương trình cài đặt theo thuật toán đã tìm.

  Bài toán liệt kê thường được "làm nền" cho nhiều bài toán khác. Hiện nay, một số bài toán tổ hợp vẫn chưa có cách nào giải ngoài cách giải liệt kê. Khó khăn chính của cách giải này là có quá nhiều cấu hình, tuy nhiên tính khả thi của phương pháp liệt kê ngày càng được nâng cao nhờ sự tiến bộ nhanh chóng về chất lượng của máy tính điện tử.

- **Bài toán tối ưu:**

  Khác với bài toán liệt kê, bài toán tối ưu chỉ quan tâm đến một cấu hình "tốt nhất" theo một nghĩa nào đấy. Đây là bài toán có nhiều ứng dụng trong thực tiễn và lý thuyết tổ hợp đã đóng góp một phần đáng kể trong việc xây dựng những thuật toán hữu hiệu.

2.2. Bài toán đếm

2.2.1. Giới thiệu bài toán

Một trong những vấn đề đầu tiên của việc nghiên cứu tổ hợp là đếm xem có bao nhiêu cấu hình được tạo ra với những quy tắc đã nêu? Để đếm chính xác, ta phải phân biệt được các cấu hình dựa vào các luật xây dựng chúng. Vì thế có thể xem các bài toán đếm là những bài luyện tập đầu tiên để con người làm quen với tư duy tổ hợp, điều này giải thích vì sao một số bài toán đếm (mặc dù dưới dạng trực giác), đã được đưa vào chương trình toán phổ thông từ những năm mới đi học.

Bài toán đếm rất phong phú kể cả dạng phát biểu lẫn cách giải. Độ khó của những bài toán đếm được trải rất rộng: từ những bài rất dễ với những số liệu cụ thể, có thể kiểm chứng bằng trực giác, đến những bài toán khó hơn, với dữ liệu đầu vào bằng chữ mà kết quả của nó được biểu diễn bằng một công thức toán học. Có những công thức được tìm qua một vài suy luận đơn giản nhưng cũng có những công thức mà việc tìm thấy chúng phải kéo dài hàng thế kỷ. Có những bài toán đếm gặp rất nhiều khó khăn (nhiều khi bế tắc) nếu giải bằng phương pháp trực tiếp, trong khi lời giải bằng quy nạp lại trở nên rõ ràng, đơn giản. Một số bài toán đếm mà luật tạo cấu hình có vẻ như không phức tạp nhưng công thức đếm thì hiện nay vẫn chưa tìm ra (hoặc chưa chứng minh được là không có).

Hiện nay, những bài toán loại này phải nhờ đến máy tính điện tử thực hiện các chương trình mang tính chất "vét cạn", vì thế kết quả còn nhiều hạn chế.

Ngoài việc rèn luyện tư duy tổ hợp, bài toán đếm còn là công cụ trong việc giải một số bài toán thuộc những lĩnh vực khác như tính xác suất của một sự kiện ngẫu nhiên (lý thuyết xác suất), đánh giá độ phức tạp của một thuật toán (lý thuyết thuật toán), ...

Bài này nhằm giới thiệu một số kỹ thuật đếm cơ bản nhằm giúp bạn đọc làm quen với tính khái quát cao của lời giải trên những bài toán đếm cụ thể.

Dưới đây là hai nguyên lý đơn giản mà sự đúng đắn của chúng là hiển nhiên, được dùng trong rất nhiều lập luận đếm.

2.2.2. Nguyên lý cộng và nguyên lý nhân

- **Nguyên lý cộng**

  Giả sử A, B là hai tập hợp thỏa mãn điều kiện *rời nhau* (nghĩa là giao của chúng là tập rỗng). Khi đó ta có:

  $$N(A \cup B) = N(A) + N(B)$$

  Công thức trên được gọi là *nguyên lý cộng*.

  Nguyên lý cộng được mở rộng một cách tự nhiên cho nhiều tập hợp như sau:

  Giả sử $\{A_1, A_2, ..., A_m\}$ là một *phân hoạch* trên tập X. Khi đó ta có:

  $$N(X) = N(A_1) + N(A_2) + ... + N(A_m)$$

  Chú ý:

- Các tập con A_1, A_2, ... trong công thức trên phải thỏa mãn là một phân hoạch của X, nghĩa là chúng *phủ* X và *rời nhau* từng cặp.
- Nguyên lý cộng dẫn việc đếm một tập hợp lớn về việc đếm những tập hợp nhỏ hơn. Chẳng hạn việc đếm số học sinh trong một lớp học được dẫn về việc đếm số học sinh trong từng tổ rồi cộng lại.
- Trong những bài toán đếm mà ta không tìm được lập luận chung cho tất cả các cấu hình, một trong những hướng giải quyết là cố gắng xây dựng một phân hoạch nào đấy trên tập được đếm sao cho trên mỗi lớp ta có một lập luận chung và nhờ thế, ta đếm được số phần tử của mỗi lớp.

  Ví dụ:

  Cần đếm số các số nguyên trong khoảng từ 1 đến 10000 có tính chất chia hết cho 7, ta chia tập được xét theo thứ tự tăng dần thành từng nhóm 7 phần tử {1, 2, ..., 7}, {8, 9, ..., 14}, ...và nhận xét rằng trong mỗi nhóm, trừ nhóm cuối cùng không đủ 7 phần tử, đều có đúng một phần tử chia hết cho 7, từ đó nhận được số cần đếm là 10000/7 = 1428 (lấy thương nguyên).

  Chứng minh:

  Giả sử A là một tập con khác rỗng của X, khi đó X được phân hoạch thành hai lớp $\{A, \overline{A}\}$ trong đó $\overline{A}$ là phần bù của A trong X, từ đó nhận được:

  $$N(X) = N(A) + N(\overline{A}) \quad \text{hay} \quad N(A) = N(X) - N(\overline{A})$$

  Nghĩa là ta có thể đếm $N(\overline{A})$ thay cho $N(A)$ trong trường hợp đếm $N(\overline{A})$ thuận tiện hơn.

- **Nguyên lý nhân**

  Giả sử A, B là hai tập hợp nào đấy. Ghép chúng với nhau để được tập tích $A \times B$. Khi đó ta có:

$$N(A \times B) = N(A).N(B)$$

  Công thức trên được gọi là *nguyên lý nhân*.

  Nguyên lý nhân được mở rộng một cách tự nhiên cho nhiều tập hợp như sau:

  Giả sử A_1, A_2, ..., A_m là những tập hợp nào đấy. Khi đó ta có:

$$N(A_1 \times A_2 \times ... \times A_m) = N(A_1).N(A_2) ... N(A_m)$$

  Nói riêng

$$N(A \times A \times ... \times A) = N(A^m) = N(A)^m$$

  Trong nhiều ứng dụng, nguyên lý nhân thường được dùng một cách chi tiết cho từng phần tử như sau:

  Giả sử phải xây dựng các bộ có thứ tự gồm m thành phần (a_1, a_2, ..., a_m), trong đó a_1 có n_1 khả năng chọn giá trị, a_2 có n_2 khả năng chọn giá trị, ..., a_m có n_m khả năng chọn giá trị. Khi đó số các bộ có thể tạo ra là $n_1n_2...n_m$.

  Nguyên lý nhân cho phép đếm những cấu hình phức tạp được tạo bằng cách ghép các cấu hình đơn giản. Khi đấy việc đếm chúng được dẫn về việc đếm các cấu hình thành phần nói chung đơn giản hơn.

  Ví dụ:

  Mọi con đường đi từ A đến B đều phải lần lượt đi qua hai cây cầu C và D. Số con đường đi từ A đến cầu C là 7, số con đường đi từ cầu C đến cầu D là 4 và số con đường đi từ cầu D đến B là 5. Hỏi từ A đến B có mấy con đường?

  Giải:

  Một con đường đi từ A đến B có thể xem như được ghép từ 3 con đường: từ A đến cầu C, từ cầu C đến cầu D và từ cầu D đến B. Từ đó, theo nguyên lý nhân, số con đường từ A đến B là 7.4.5 = 140.

  Chú ý:

  - Số lượng đếm theo nguyên lý nhân tăng rất nhanh khi số thành phần hoặc giá trị của mỗi thành phần tăng, điều này dẫn đến các kết quả đếm có thể rất lớn dù số thành phần hay giá trị của mỗi thành phần là không lớn lắm.
  - Nguyên lý cộng và nguyên lý nhân là những nguyên lý đơn giản nhất. Trong các bài toán đếm thông dụng, hầu như đều sử dụng hai nguyên lý này. Ta bắt đầu minh họa việc dùng chúng qua bài toán đếm những cấu hình cơ bản.

2.2.3. Đếm các cấu hình cơ bản

- **Đếm các chỉnh hợp**

  Theo định nghĩa, một chỉnh hợp lặp chập m của n phần tử là một bộ có thứ tự gồm m thành phần, trong đó mỗi thành phần được chọn từ n phần tử đã cho và được lặp tùy ý, vì thế, các khả năng chọn giá trị cho mỗi thành phần là như nhau và cùng bằng n. Theo nguyên lý nhân, số các chỉnh hợp lặp chập m của n phần tử bằng tích của n với chính nó m lần, nghĩa là bằng n^m.

Ví dụ:

Các dữ liệu xử lý trong máy tính đều được mã hóa bằng một *dãy nhị phân* (mà người ta thường gọi là một *chuỗi bit*). Tùy yêu cầu xử lý mà mỗi kiểu dữ liệu được chọn một độ dài mã hóa thích hợp.

Thật vậy:

Một chuỗi bit độ dài n là một bộ có thứ tự gồm n thành phần nhận các giá trị 0 hoặc 1 một cách tùy ý, chính là một chỉnh hợp lặp chập n của 2 giá trị. Từ đó nhận được số các chuỗi n-bit là 2^n. Kết quả này cho ta xác định số lượng tối đa khả năng mã hóa dữ liệu bằng các chuỗi nhị phân tùy vào độ dài của chuỗi mã mà các kiểu dữ liệu quy định.

Thực tế:

Trong các ngôn ngữ lập trình, một giá trị nguyên có thể khai báo theo nhiều kiểu dữ liệu khác nhau với các kích thước 1 byte (8 bit), 2 byte (16 bit), 4 byte (32 bit).

Với các số nguyên 1 byte, ta có $2^8 = 256$ giá trị

Với các số nguyên 2 byte có $2^{16} = 65536$ giá trị

Với các số nguyên 4 byte có 2^{32} (khoảng hơn 4 tỷ) giá trị.

Hiểu được điều này, người lập trình sẽ tránh được những lỗi tràn số (là những lỗi mà các chương trình dịch thường không kiểm tra) dẫn đến sai kết quả rất khó tìm.

Việc đếm các chỉnh hợp (không lặp) chập *m* của *n* phần tử ($m \le n$) được tiến hành tương tự với chú ý giá trị chọn cho mỗi thành phần là *không được lặp*, do vậy các khả năng chọn cho mỗi thành phần lần lượt là n, n−1, n−2, ... Từ đó nhận được số các chỉnh hợp chập m của n phần tử là $n(n-1)(n-2)...(n-m+1)$.

Ví dụ:

Một lớp sinh viên cần phải hoàn thành việc thi 4 môn trong 6 ngày, trong đó mỗi ngày không được thi quá một môn. Hỏi có bao nhiêu cách xếp lịch thi?

Giải:

Biểu diễn mỗi lịch thi như là một bộ có thứ tự gồm 4 thành phần, thành phần thứ i ghi nhận ngày thi môn i (i = 1, 2, 3, 4). Do không có ngày nào thi quá một môn nên các thành phần này phải nhận các giá trị khác nhau. Từ đó ta nhận được mỗi lịch thi là một chỉnh hợp chập 4 của 6 giá trị và số lịch thi bằng 6.5.3.4 = 360.

- **Đếm các hoán vị**

  Vì mỗi hoán vị của n phần tử là một chỉnh hợp chập n của n phần tử nên số hoán vị của n phần tử bằng $n.(n-1)(n-2)...2.1 = n!$ (đọc là n giai thừa).

  Chú ý:

  n! tăng rất nhanh khi n tăng, ta có thể thấy điều đó qua việc tính một số giá trị cụ thể 3! = 6, 4! = 24, 5! = 120, 6! = 720, 7! = 5040, 8! = 40320, 9! = 362880, ...

  Cấu hình hoán vị có liên quan đến những bài toán lập lịch.

  Chẳng hạn để thực hiện 6 công việc trong ngày, ta có 6! = 720 cách sắp xếp thứ tự thực hiện khác nhau. Giá trị lớn của n! cũng cho thấy những bài toán lập lịch là những bài toán khó.

- **Đếm các tổ hợp**

  Tổ hợp khác với chỉnh hợp chỉ một thuộc tính: chỉnh hợp là bộ có thứ tự còn tổ hợp thì không. Vì vậy, mỗi tổ hợp chập m của n được đồng nhất với một lớp các chỉnh hợp (cùng kích thước) chỉ sai khác nhau về thứ tự giá trị các thành phần. Để ý rằng, các lớp này là một phân hoạch của tập các chỉnh hợp đang xét và số phần tử của mỗi lớp đều bằng m! (số hoán vị của m giá trị), ta nhận được theo nguyên lý cộng:

m!(số tổ hợp chập m của n) = số chỉnh hợp chập m của n

Trong toán học, số tổ hợp chập m được ký hiệu là C_n^m và số chỉnh hợp chập m của n được ký hiệu là A_n^m. Viết lại công thức trên theo các ký hiệu này, ta được:

$$m!C_n^m = A_n^m \text{ hay } C_n^m = \frac{A_n^m}{m!} = \frac{n(n-1)...(n-m+1)}{m!} = \frac{n!}{m!(n-m)!}$$

Ví dụ:

Trong vòng đấu bảng của một giải bóng đá gồm 32 đội tham gia, người ta chia làm 8 bảng (mỗi bảng 4 đội) và thi đấu vòng tròn (1 lượt) trong mỗi bảng, hỏi ban điều hành phải lo tổ chức bao nhiêu trận?

Giải:

Luật thi đấu vòng tròn đảm bảo 2 đội cùng bảng được gặp nhau đúng một trận. Việc lấy ra 2 đội từ 4 đội để tổ chức một trận đấu, là một tổ hợp chập 2 của 4, từ đó nhận được số trận đấu của một bảng là $C_4^2 = \frac{4.3}{2} = 6$, và số trận đấu của vòng bảng là 6.8 = 48.

Khái quát công thức trên, ta nhận được số trận đấu vòng tròn trong một bảng có n đội là $C_n^2 = \frac{n(n-1)}{2}$. Kết quả này cho thấy số trận đấu vòng tròn là một đại lượng tỷ lệ với bình phương số đội tham gia thi đấu, điều này giải thích tại sao trong những giải ngắn ngày, người ta phải chia nhỏ các đội thành từng bảng và chỉ đấu vòng tròn trong từng bảng (bạn đọc thử tính số trận đấu vòng tròn của cả 32 đội).

Kết quả C_n^2 còn được gặp trong những bài toán khác, chẳng hạn, nó là số đường thẳng được tạo khi nối tất cả các cặp điểm từ n điểm (không chứa 3 điểm nào thẳng hàng) cho trước. Những bài toán như vậy là *cùng bản chất tổ hợp* dù được phát biểu khác nhau.

Các số C_n^m được gặp khá nhiều trong các hệ thức toán học, chúng được gọi chung là các *hệ số tổ hợp*. Dưới đây là một số tính chất của các hệ số này:

- Đối xứng: $C_n^m = C_n^{n-m}$
- Điều kiện đầu: $C_n^0 = C_n^n = 1,\ n \geq 0$
- Đệ quy: $C_n^m = C_{n-1}^{m-1} + C_{n-1}^m,\ n > m > 0$

Các tính chất trên có thể chứng minh từ định nghĩa tổ hợp hoặc bằng công thức xác định C_n^m.

Từ điều kiện ban đầu và công thức đệ quy, ta có thể tính mọi hệ số tổ hợp chỉ bằng phép cộng như sau: viết các hệ số này vào một bảng dạng tam giác trong đó mỗi dòng lần lượt là các hệ số tổ hợp của n (bắt đầu n = 0), cụ thể dòng đầu tương ứng với n = 0 là C_0^0, dòng thứ hai tương ứng với n = 1 là C_1^0, C_1^1, dòng thứ ba tương ứng với n = 2 là C_2^0, C_2^1, C_2^2, ... sau đó điền các giá trị tương ứng của các hệ số vào vị trí của chúng lần lượt từ dòng đầu, ..., theo công thức điều kiện đầu, các giá trị ở hai đầu dòng đều bằng 1 (như thế dòng thứ nhất và dòng thứ hai được xác định), từ dòng thứ ba, các giá trị bên trong dòng được xác định từ các giá trị trên dòng trước đấy nhờ công thức đệ quy. Bảng số nhận được, được gọi là *tam giác Pascal* (kích thước n). Chẳng hạn, ta có tam giác Pascal kích thước 7:

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1

trong đó chứa tất cả các hệ số tổ hợp C_n^m, $0 \le m \le n$, $0 \le n \le 7$.

Các hệ số tổ hợp còn liên quan đến các hệ số trong khai triển nhị thức theo công thức:

$$(a+b)^n = C_n^0 a^n + C_n^1 a^{n-1} b + C_n^2 a^{n-2} b^2 + \ldots + C_n^{n-1} a b^{n-1} + C_n^n b^n = \sum_{k=0}^{n} C_n^k a^{n-k} b^k$$

Việc chứng minh nó có thể dùng quy nạp. Cũng vì thế các hệ số tổ hợp còn có tên là các *hệ số nhị thức*.

Nhờ tam giác Pascal, ta có thể tính nhanh các khai triển nhị thức (chỉ bằng phép cộng), chẳng hạn để khai triển $(a + b)^7$ ta tính tam giác Pascal đến kích thước 7 và "lắp" dòng cuối cùng vào:

$$(a+b)^7 = a^7 + 7a^6 b + 21a^5 b^2 + 35a^4 b^3 + 35a^3 b^4 + 21a^2 b^5 + 7ab^6 + a^7.$$

Có thể thấy các công thức khai triển trên là sự mở rộng của các "hằng đẳng thức đáng nhớ" của đại số sơ cấp:

$$(a + b)^2 = a^2 + 2ab + b^2,\ (a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3, \ldots$$

Từ công thức khai triển nhị thức, khi thay a, b bằng các giá trị cụ thể, ta được các hệ thức khác nhau giữa các hệ số tổ hợp, chẳng hạn khi thay a = b = 1, ta được:

$$C_n^0 + C_n^1 + C_n^2 + \ldots + C_n^{n-1} + C_n^n = 2^n$$

Hoặc khi thay a = 1, b = −1, ta được:

$$C_n^0 - C_n^1 + C_n^2 - \ldots + (-1)^{n-1} C_n^{n-1} + (-1)^n C_n^n = 0.$$

2.2.4. Các kỹ thuật đếm đơn giản qua các ví dụ

Trong mục này, ta sẽ giải một số bài toán đếm đơn giản như những ví dụ minh họa cho những kỹ thật đếm khác nhau. Nói chung, các bài toán loại này dựa vào các nguyên lý cộng, nhân và các cấu hình cơ bản. Bạn đọc cần chú ý đến tính khái quát của lời giải mặc dù được trình bày trên những vấn đề cụ thể.

Ví dụ 1.

Có bao nhiêu cách xếp 5 người thành một hàng ngang sao cho A không đứng cạnh B (A và B là hai người nào đó trong 5 người)?

Giải:

Mỗi cách xếp 5 người là một hoán vị của 5 phần tử. Dựa vào vị trí của A và B, ta có tất cả 4 trường hợp:

a) giữa A và B có 1 người

b) giữa A và B có 2 người

c) giữa A và B có 3 người

d) A và B cạnh nhau

Như thế, số cách xếp cần đếm thuộc về 3 trường hợp đầu và phần còn lại thuộc về trường hợp cuối. Với tình huống này, ta nên đếm số cách xếp thuộc trường hợp cuối rồi lấy tổng số trừ đi. Đây là một cách dùng nguyên lý cộng.

Tổng số cách xếp 5 vị trí là 5! = 120. Để đếm số cách xếp A cạnh B, ta đồng nhất vị trí của A và B làm một và bài toán đưa về cách xếp 4 vị trí. Chú ý có 2 cách đồng nhất A và B (A bên trái B và A bên phải B) vì vậy số cách xếp A đứng cạnh B là 4!2 = 48. Từ đó nhận được số cách xếp A không cạnh B là 120 − 48 = 72.

Ví dụ 2.

Ta có số đường chéo của tam giác bằng 0, của tứ giác bằng 2, của ngũ giác bằng 5, ..., nghĩa là số đường chéo của một đa giác tăng theo số đỉnh của nó. Bài toán đặt ra là tìm một công thức phản ánh sự phụ thuộc này: xác định số đường chéo của một đa giác (lồi) n đỉnh.

Giải.

Xét tất cả các đường thẳng được tạo ra bằng cách nối các cặp đỉnh của đa giác. Số các đường thẳng này là C_n^2. Nhận xét rằng, các đường thẳng này thuộc một trong hai loại: hoặc là cạnh của đa giác, hoặc là đường chéo của nó. Số cạnh của đa giác bằng n, từ đó nhận được số đường chéo của đa giác là:

$$C_n^2 - n = \frac{n(n-1)}{2} - n = \frac{n(n-3)}{2}, \ (n \geq 3).$$

Ví dụ 1 và 2 cho một cách dùng nguyên lý cộng, đó là *đếm phần bù* thay cho đếm trực tiếp nếu việc đếm phần bù đơn giản hơn.

Ví dụ 3.

Mã số sinh viên của một trường đại học được ghép từ 2 chữ cái lấy từ 4 chữ cái A, B, C, D (phân biệt ngành đào tạo) và 3 chữ số lấy từ 10 chữ số 0, 1, ..., 9 (phân biệt người trong một ngành). Hỏi hệ thống mã này quản lý được tối đa bao nhiêu sinh viên?

Giải.

Số sinh viên tối đa được quản lý bằng số các chuỗi mã được tạo. Theo nguyên lý nhân, giá trị này bằng $4^2.10^3$ = 16000 (chú ý các ký tự trong chuỗi mã được phép lặp lại).

Ví dụ 4.

Một lớp học sinh bầu một lớp trưởng và hai lớp phó. Danh sách đề cử gồm 6 người. Hỏi có bao nhiêu kết cục khác nhau?

Giải.

Biểu diễn mỗi kết cục bầu như một bộ 2 thành phần (có thứ tự), trong đó thành phần đầu ghi tên lớp trưởng và thành phần sau ghi tên hai lớp phó. Để chọn lớp trưởng từ 6 người, ta có 6 cách, để chọn hai lớp phó từ 5 người còn lại ta có $C_5^2 = 10$ cách (chú ý hai lớp phó không phân biệt thứ tự). Theo nguyên lý nhân số các kết cục là 6.10 = 60, (bạn đọc thử chọn hai lớp phó trước rồi chọn lớp trưởng sau để thấy được cùng một kết quả).

Các ví dụ 3 và 4 cho thấy nguyên lý nhân được dùng đếm các cấu hình được tạo từ các phép ghép các thành phần. Việc đếm các khả năng cho mỗi thành phần được thực hiện trực tiếp hoặc bằng những lập luận đơn giản.

Ví dụ 5.

Đếm số tập con của một tập n phần tử?

Giải.

Một tập con của một tập n phần tử được xác định bằng cách liệt kê tất cả n trạng thái "chọn/không chọn" của các phần tử đang xét. Như thế mỗi tập con này được biểu diễn (duy nhất) bằng một chuỗi n bit. Từ đó nhận được số cần đếm là 2^n (xem mục đếm các chỉnh hợp).

Cũng có thể giải bài này bằng nhận xét số cần đếm là vế trái của công thức:

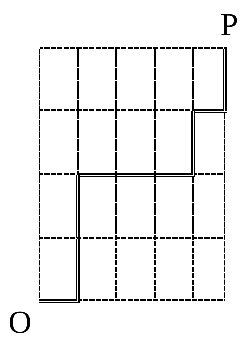
$$C_n^0 + C_n^1 + C_n^2 + \ldots + C_n^{n-1} + C_n^n = 2^n$$

Việc biểu diễn một tập hợp bằng một chuỗi n bit được dùng trong việc xây dựng kiểu dữ liệu tập hợp trong máy tính, chẳng hạn kiểu tập hợp của Turbo Pascal được tổ chức bằng một chuỗi 32 bit (4 byte).

Chú ý rằng 2^n là rất lớn với n không lớn lắm. Trong lý thuyết thuật toán, những thuật toán nào có độ phức tạp cỡ 2^n được xem là không khả thi.

Ví dụ 6.

Trên mặt phẳng, xét một lưới tọa độ nguyên có gốc tọa độ là O. Giả thiết P là điểm có tọa độ nguyên dương (m, n). Một đường đi từ O đến P được định nghĩa theo các điều kiện: chỉ được đi theo các cạnh của lưới và trên mỗi cạnh chỉ được đi theo hướng từ trái sang phải hay từ dưới lên trên. Hỏi số đường đi từ O đến P?



Hình vẽ trên mô tả một đường đi từ O đến P (đường gạch đôi) có tọa độ (5, 4).

Giải.

Theo định nghĩa, một đường đi bất kỳ từ O đến P đều đi qua m + n cạnh, trong đó có đúng m cạnh đi từ trái sang phải và n cạnh đi từ dưới lên trên. Như thế ta có thể biểu diễn đường đi này bằng một chuỗi m + n bit (mỗi bít biểu diễn trạng thái của một cạnh), trong đó có đúng m bit bằng 1 (tương ứng với đi ngang) và n bit bằng 0 (tương ứng với đi dọc). Theo ví dụ 3, mỗi chuỗi này biểu diễn một tập con m phần tử của một tập m+n phần tử, nghĩa là một tổ hợp chập m của m+n. Từ đó nhận được số đường đi từ O đến P là $C_{m+n}^{m} = C_{m+n}^{n}$. Trên hình vẽ, số đường đi này là $C_7^4 = C_7^3 = \frac{7.6.5}{2.3} = 35$.

Các ví dụ 4 và 5 cho thấy một kỹ thuật quan trọng trong việc đếm, đó là việc tìm một *biểu diễn thích hợp* cho các cấu hình cần đếm. Biểu diễn này phải đảm bảo sự *tương ứng một-một* và việc đếm các biểu diễn đó là thuận tiện (thường được đưa về việc đếm các cấu hình cơ bản).

Ví dụ 7.

Xét chuỗi ký tự, trong đó ký tự thứ i được lặp k_i lần ($k_i \geq 1$, i = 1, 2, m). Hoán vị các ký tự trong chuỗi. Hỏi có bao nhiêu chuỗi khác nhau?

Giải.

Độ dài của chuỗi là $k_1 + k_2 + ... + k_m = n$. Nếu xem các ký tự trong chuỗi này đều khác nhau thì số chuỗi là n!. Tuy nhiên trong số này, khi đồng nhất k_1 ký tự thứ nhất, ta phân hoạch được chúng thành những lớp mà số phần tử của mỗi lớp bằng $k_1!$. Trong mỗi lớp như vậy, ta lại đồng nhất k_2 ký tự thứ hai và nhận được một phân hoạch mà mỗi lớp có số phần tử bằng $k_1!k_2!$... Cuối cùng ta nhận được một phân hoạch mà mỗi lớp có số phần tử bằng $k_1!k_2!...k_m!$. Mỗi lớp của phân hoạch này tương ứng với một chuỗi cần đếm. Từ đó nhận được số hoán vị khác nhau của chuỗi đã cho là:

$$\frac{n!}{k_1!k_2!...k_m!} = \frac{(k_1 + k_2 + ... + k_m)!}{k_1!k_2!...k_m!}$$

Hoán vị trong bài tập này được gọi là *hoán vị lặp*, các giá trị k_1, k_2, ... được gọi là các *tần suất* của phần tử tương ứng. Hoán vị theo nghĩa thông thường là trường hợp riêng của hoán vị lặp với các tần suất phần tử đều bằng 1. Chẳng hạn, các hoán vị của từ SUCCESS là các hoán vị lặp của các ký tự S, U, C, E, trong đó tần suất của S là 3, của C là 2 và của U, E là 1, số lượng của chúng, theo công thức trên là $\frac{7!}{3!2!} = 420$.

Có thể giải bài toán được xét bằng một lập luận khác. Một hoán vị cần đếm được xác định từ việc chọn vị trí cho các ký tự trong chuỗi. Tổng số các vị trí được chọn là độ dài của chuỗi ($n = k_1 + k_2 + ... + k_m$). Mỗi cách chọn k_1 vị trí cho ký tự thứ nhất là một tổ hợp chập k_1 của n, mỗi cách chọn k_2 vị trí còn lại cho ký tự thứ hai là một tổ hợp chập k_2 của $n - k_1$, ... Theo nguyên lý nhân, số tất các các cách chọn vị trí (cũng là số cần đếm) bằng:

$$C_n^{k_1} C_{n-k_1}^{k_2} ... C_{n-k_1-k_2-...-k_{m-1}}^{k_m}$$

$$= \frac{n!}{k_1!(n-k_1)!} \frac{(n-k_1)!}{k_2!(n-k_1-k_2)!} ... \frac{(n-k_1-k_2-...-k_{m-1})!}{k_m!} = \frac{n!}{k_1!k_2!...k_m!}$$

Cách giải trên cho ta thấy kết quả tìm được cũng là số cách chia n phần tử thành m nhóm (có thứ tự) sao cho nhóm thứ i có k_i phần tử ($k_i \geq 1$, i = 1, 2,…, m). Chẳng hạn, khi chia 52 quân bài cho 4 người (mỗi người 13 quân), số kết cục khác nhau sẽ là $\frac{52!}{(13!)^4}$.

Ví dụ 8.

Có 10 cái kẹo (giống nhau) được chia hết cho 3 cháu bé sao cho cháu nào cũng có ít nhất 1 cái. Hỏi số cách chia?

Giải.

Xếp 10 cái kẹo thành một hàng (có 9 khe hở giữa hai cái kẹo) và dùng 2 que đặt vào các khe hở này để chia thành 3 phần. Theo yêu cầu cách chia, mỗi khe chỉ được đặt nhiều lắm là một que. Hình vẽ dưới đây mô tả một cách chia (mỗi dấu * mô tả một cái kẹo) trong đó cháu ở đầu được 2 cái, cháu ở giữa được 5 cái và cháu ở cuối được 3 cái:

* * | * * * * * | * * *

Như vậy một cách chia kẹo tương ứng với một cách chọn 2 vị trí (không phân biệt thứ tự đặt que) từ 9 vị trí, nghĩa là một tổ hợp chập 2 của 9, từ đó nhận được số cách chia là $C_9^2 = \frac{9.8}{2} = 36.$

Nếu ký hiệu x_1, x_2, x_3 lần lượt là số kẹo chia cho 3 cháu thì mỗi cách chia sẽ là một nghiệm nguyên dương của phương trình:

$$x_1 + x_2 + x_3 = 10$$

từ đó nhận được số nghiệm nguyên dương của phương trình trên là C_9^2.

Lập luận trên được mở rộng cho việc chia n cái kẹo thành k phần và ta được kết quả số nghiệm nguyên dương của phương trình:

$$x_1 + x_2 + ... + x_k = n \qquad (1)$$

là C_{n-1}^{k-1}.

Việc đếm tất cả các nghiệm nguyên không âm của phương trình trên được dẫn về việc đếm số các nghiệm nguyên dương của một phương trình cùng loại. Thật vậy, đặt $y_i = x_i + 1$, i = 1, 2, ..., k, ta được:

$$(y_1 - 1) + (y_2 - 1) + ... + (y_k - 1) = n$$

hay $$y_1 + y_2 + ... + y_k = n + k \qquad (2)$$

Mỗi nghiệm nguyên không âm của phương trình (1) tương ứng với một nghiệm nguyên dương của phương trình (2), từ đó nhận được số các nghiệm nguyên không âm của phương trình (1) là: $C_{n+k-1}^{k-1} = C_{n+k-1}^{n}$.

Mỗi nghiệm nguyên không âm của phương trình (1) lại có thể được giải thích như việc chọn ra n phần tử từ một tập các phần tử gồm k loại, sao cho không phân biệt các phần tử cùng loại và mỗi loại được lặp tùy ý từ 0 tới n (vì thế cần giả thiết mỗi loại có ít nhất n phần tử). Một cách chọn như vậy được gọi là một tổ hợp lặp chập n của k (chú ý không có ràng buộc gì giữa n và k). Theo kết quả vừa tìm, số các tổ hợp lặp chập n của k bằng C_{n+k-1}^{n}.

Chẳng hạn có một sọt quả gồm 3 loại cam, táo, lê trong đó mỗi loại có ít nhất 4 quả. Số cách chọn ra 4 quả không phân biệt các quả cùng loại là số tổ hợp lặp chập 4 của 3 và bằng $C_6^4 = C_6^2 = \frac{6.5}{2} = 15$.

Các ví dụ 7 và 8 là những minh họa cho *tính khái quát* của một bài toán tổ hợp. Một bài toán tổ hợp có thể tiếp cận dưới những góc độ khác nhau và một lời giải của một bài toán cụ thể có thể mở rộng cho nhiều bài toán khác. Các kết quả trong các ví dụ 7 và 8 cho ta một sự mở rộng của các cấu hình hoán vị và tổ hợp, trong đó có xét đến việc lặp lại của các phần tử.

Bài toán đếm còn được dùng cho việc tính xác suất và đánh giá độ phức tạp của một thuật toán như trong hai ví dụ 9, 10 dưới đây.

Ví dụ 9.

Một đợt phát hành số xố với các sêri có dạng XXNNNN, trong đó X lấy từ tập 26 chữ cái A, B, ..., Z và N lấy từ tập 10 chữ số 0, 1, ..., 9. Một người mua ngẫu nhiên một vé. Hỏi xác suất để anh ta trúng giải độc đắc?

Giải.

Theo lý thuyết xác suất, trong trường hợp đồng khả năng, xác suất để xảy ra một sự kiện bằng tỷ số của số khả năng xảy ra sự kiện đó trên tổng số khả năng có thể xảy ra. Như vậy xác suất cần tìm bằng 1/n, trong đó n là tổng số vé được phát hành. Theo nguyên lý nhân, $n = 26^2.10^4 = 6760000$ và xác suất cần tìm là $1/6760000 \approx 0,00000015$. Trên thực tế, những sự kiện có xác suất như thế này (nhỏ hơn một phần triệu) hầu như không xảy ra.

Ví dụ 10.

Thuật toán "nổi bọt" là một trong những thuật toán sắp xếp dễ cài đặt nhất. Thuật toán này dựa vào kết quả so sánh hai phần tử kề nhau để đổi chỗ nếu cần. Hãy đếm số phép so sánh cần thực hiện để sắp xếp một dãy gồm n phần tử.

Giải.

Nhận xét rằng trong thuật toán "nổi bọt", hai phần tử bất kỳ trong dãy được so sánh với nhau đúng một lần. Như thế, việc đếm các phép so sánh không khác gì việc đếm số trận đấu "vòng tròn" (một lượt) của n đội và ta nhận được số phép so sánh cần thực hiện là $C_n^2 = \frac{n(n-1)}{2}$. Từ kết quả này, người ta nói rằng thuật toán "nổi bọt" có độ phức tạp cỡ $O(n^2)$, nó chậm hơn thuật toán sắp xếp nhanh (quick-sort) có độ phức tạp cỡ O(n.logn), tuy nhiên nó dễ cài đặt hơn và thường được dùng trong việc sắp xếp một dãy có kích thước không lớn lắm.

Dưới đây trình bày một nguyên lý, xem như việc mở rộng của nguyên lý cộng, được dùng cho những bài toán đếm phức tạp hơn.

2.2.5. Nguyên lý bù trừ

Nguyên lý cộng cho ta một công thức đơn giản đếm số phần tử của một hợp các tập hợp với giả thiết các tập này là *rời nhau từng cặp*. Trong trường hợp tổng quát, không có giả thiết này, kết quả sẽ phức tạp hơn vì phát sinh thêm những mối quan hệ "giao nhau" giữa các tập hợp. Để đơn giản, trước hết ta xét cho hai tập hợp A, B. Khi đó dễ dàng nhận được công thức:

$$N(A \cup B) = N(A) + N(B) - N(A \cap B)$$

Dùng công thức trên hai lần, ta mở rộng được cho ba tập hợp A, B, C:

$$N(A \cup B \cup C) =$$
$$N(A) + N(B) + N(C) - (N(A \cap B) + N(A \cap C) + N(B \cap C)) + N(A \cap B \cap C).$$

Bằng cách đưa vào các ký hiệu:

$$N_1 = N(A) + N(B) + N(C),$$
$$N_2 = N(A \cap B) + N(A \cap C) + N(B \cap C),$$
$$N_3 = N(A \cap B \cap C)$$

ta có thể mở rộng cho các tập hợp $A_1, A_2, ..., A_m$ nhờ quy nạp như sau:

$N(A_1 \cup A_2 \cup ... \cup A_m) = N_1 - N_2 + ... + (-1)^{m-1} N_m$, trong đó các N_k (k = 1, 2, ..., m) được xác định từ các công thức $N_k = \sum N(A_{i_1} \cap A_{i_2} \cap ... \cap A_{i_k})$ với tổng được lấy theo các tổ hợp $(i_1, i_2, ..., i_k)$ của tập m chỉ số {1, 2, ..., m}.

Công thức này được gọi là *nguyên lý cộng tổng quát*, trong đó nguyên lý cộng đã xét là trường hợp riêng (các N_k, trừ k = 1, đều bằng 0).

Tuy nhiên, trên thực tế, ta hay gặp bài toán đếm phần bù của hợp các tập hợp, dưới dạng phát biểu tổng quát sau:

Bài toán. Trên tập hữu hạn X xét m tính chất $A_1, A_2, ..., A_m$. Hãy đếm xem có bao nhiêu phần tử của X, *không thỏa mãn bất kỳ* một tính chất đã cho nào.

Giải. Đồng nhất mỗi tính chất A_i (i = 1, 2, ..., m) như tập con A_i của X. Số phần tử cần đếm thuộc phần bù của hợp các tập hợp này. Gọi N là số phần tử của X và $\overline{N}$ là số phần tử cần đếm, theo công thức vừa tìm, ta được:

$$\overline{N} = N - N(A_1 \cup A_2 \cup ... \cup A_m) = N - N_1 + N_2 - ... + (-1)^m N_m$$

trong đó các N_k (k = 1, 2, ..., m) được xác định từ các công thức $N_k = \sum N(A_{i_1} \cap A_{i_2} \cap ... \cap A_{i_k})$ với tổng được lấy theo các tổ hợp $(i_1, i_2, ..., i_k)$ của tập m chỉ số {1, 2, ..., m}.

Công thức trên được gọi là *nguyên lý bù trừ*. Thực chất nguyên lý này là một cách phát biểu khác của nguyên lý cộng tổng quát, nó chuyển bài toán tìm $\overline{N}$ về bài toán tính các N_k mà trong nhiều tình huống tìm được một lập luận đơn giản hơn.

Ví dụ 1.

Đếm xem có bao nhiêu có bao nhiêu số nguyên từ 1 đến 10000 thỏa mãn tính chất không chia hết cho bất cứ số nào trong các số 3, 4 và 7.

Giải:

Gọi X là tập các số nguyên từ 1 đến 10000, số lượng của nó là N = 10000.

Trên X, xét các tính chất:

- A_1 – chia hết cho 3
- A_2 – chia hết cho 4
- A_3 – chia hết cho 7

Bài toán được đưa về việc xác định số $\overline{N}$ của nguyên lý bù trừ với 3 tính chất A_1, A_2, A_3 cho trên X, ta được:

$$\overline{N} = N - N_1 + N_2 - N_3$$

trong đó:

$$N_1 = N(A_1) + N(A_2) + N(A_3)$$
$$N_2 = N(A_1 \cap A_2) + N(A_1 \cap A_3) + N(A_2 \cap A_3)$$
$$N_3 = N(A_1 \cap A_2 \cap A_3)$$

Nhận xét:

$N(A_1)$ là số các số nguyên từ 1 đến 10000 chia hết cho 3, ta được giá trị này bằng 10000/3 = 3333 (lấy thương nguyên)

Tương tự: $N(A_2)$ = 10000/4 = 2500, $N(A_3)$ = 10000/7 = 1428 và nhận được N_1 = 3333 + 2500 + 1428 = 7261.

$N(A_1 \cap A_2)$ là số các số nguyên từ 1 đến 10000 vừa chia hết cho 3, vừa chia hết cho 4, nghĩa là chia hết cho 12, được tính bằng 10000/12 = 833.

Cũng vậy $N(A_1 \cap A_3)$ là số các số nguyên từ 1 đến 10000 chia hết cho 3.7 = 21 được tính bằng 10000/21 = 476 và $N(A_2 \cap A_3)$ là số các số nguyên từ 1 đến 10000 chia hết cho 4.7 = 28 được tính bằng 10000/28 = 357. Ta được N_2 = 833 + 476 + 357 = 1666.

Tương tự $N_3 = N(A_1 \cap A_2 \cap A_3)$ là số các số nguyên từ 1 đến 10000 chia hết cho 3.4.7 = 84 được tính bằng 10000/84 = 119.

Cuối cùng ta nhận được $\overline{N} = N - N_1 + N_2 - N_3 = 10000 - 7261 + 1666 - 119 = 4286$.

Bài toán dưới đây được lấy từ một ví dụ thường được nêu trong các giáo trình "Lý thuyết xác suất".

Ví dụ 2 (Bài toán bỏ thư).

Có n lá thư và n phong bì ghi sẵn địa chỉ. Bỏ ngẫu nhiên mỗi lá thư vào mỗi phong bì (không xem địa chỉ). Hỏi xác suất xảy ra sự kiện không có lá thư nào bỏ đúng địa chỉ là bao nhiêu?

Giải:

Mỗi cách bỏ thư là một hoán vị của n phần tử, vì thế số lượng của chúng là n!. Bây giờ cần đếm số cách bỏ thư để tất cả các lá thư đều sai địa chỉ.

Gọi X là tập hợp tất cả các cách bỏ thư và A_i là tính chất lá thư thứ i bỏ đúng địa chỉ (i = 1, 2, ..., n). Khi đó số cần đếm chính là số $\overline{N}$ trong nguyên lý bù trừ của n tính chất A_i cho trên X và ta được:

$$\overline{N} = N - N_1 + N_2 - ... + (-1)^n N_n$$

trong đó N là số phần tử của X và bằng n! còn các N_k (k = 1, 2, ..., n) được xác định từ các công thức $N_k = \sum N(A_{i_1} \cap A_{i_2} \cap ... \cap A_{i_k})$ với tổng được lấy theo các tổ hợp $(i_1, i_2, ..., i_k)$ của tập n chỉ số {1, 2, ..., n}.

Để tính N_k ta để ý rằng $N(A_{i_1} \cap A_{i_2} \cap ... \cap A_{i_k})$ là số cách bỏ thư sao cho có k lá thư $i_1, i_2, ..., i_k$ đúng địa chỉ do đó nó bằng (n − k)!.

Mỗi tổ hợp chập k của n xác định một số hạng như thế, vì vậy N_k là tổng của C_n^k số hạng, mỗi số hạng đều bằng (n − k)!

Do đó: $N_k = C_n^k(n-k)! = \dfrac{n!}{k!(n-k)!}(n-k)! = \dfrac{n!}{k!}$.

Thay các giá trị tìm được vào vế phải công thức $\overline{N} = N - N_1 + N_2 - ... + (-1)^n N_n$, ta được:

$$\overline{N} = n! - \frac{n!}{1!} + \frac{n!}{2!} - ... + (-1)^n \frac{n!}{n!} = n!(1 - \frac{1}{1!} + \frac{1}{2!} - ... + (-1)^n \frac{1}{n!}).$$

Cuối cùng, ta nhận được xác suất p cần tìm:

$$p = \frac{\overline{N}}{n!} = 1 - \frac{1}{1!} + \frac{1}{2!} - ... + (-1)^n \frac{1}{n!}.$$

Điều thú vị là khi cho $n \to \infty$, vế phải của công thức trên dần đến e^{-1} (xem khai triển hàm e^{-x} thành chuỗi lũy thừa) nghĩa là xác suất cần tìm lớn hơn 1/3 khi n đủ lớn. Số $\overline{N}$ trong bài toán trên được gọi là *số mất thứ tự* và được ký hiệu là D_n. Công thức xác định D_n chứng tỏ nó là một vô cùng lớn cùng bậc với n! vì thế nó tăng rất nhanh khi n tăng.

Dưới đây là các giá trị được tính của D_n với n = 1, 2, ..., 11 để bạn đọc hình dung tốc độ tăng của nó:

n	1	2	3	4	5	6	7	8	9	10	11
D_n	0	1	2	9	44	265	1854	14833	133496	1334961	4890741

Việc tìm một công thức cho bài toán đếm không phải lúc nào cũng đơn giản ngay cả khi các điều kiện tạo cấu hình có vẻ không phức tạp. Với những bài toán như vậy, người ta phải phân tích chúng để đưa về các bài toán đếm đơn giản hơn.

Ví dụ dưới đây là một bài toán nổi tiếng của Lucas (1891) mà việc dùng nguyên lý bù trừ là một khâu trong quá trình tìm lời giải.

Ví dụ 3 (Bài toán xếp khách của Lucas).

Có một bàn tròn, xung quanh có 2n ghế. Cần sắp chỗ cho n cặp vợ chồng sao cho các ông ngồi xen kẽ với các bà và không có cặp vợ chồng nào ngồi cạnh nhau. Hỏi có bao nhiêu cách xếp?

Giải:

Xếp chỗ cho các bà trước theo nguyên tắc giữa hai bà để một ghế trống (dành cho các ông). Dễ dàng tính được số cách xếp như thế bằng 2n!. Sau khi xếp cho các bà, ta xếp chỗ cho các ông ngồi vào các ghế trống sao cho thỏa mãn nốt điều kiện không ông nào ngồi cạnh vợ mình. Nếu tìm được số cách xếp này, ký hiệu là U_n thì số cách xếp, tính cả các cặp sẽ là $(2n!)U_n$. Như vậy bài toán được dẫn về xác định số U_n, đơn giản hơn vì đã bớt đi một điều kiện.

Đánh số các bà từ 1 đến n theo trật tự vòng tròn (sau khi đã xếp), đánh số các ông tương ứng với các bà (ông i là chồng bà i) và đánh số các ghế trống theo qui tắc: ghế số i là ghế ở giữa bà i và bà i + 1 (các phép cộng được hiểu theo thứ tự vòng tròn nghĩa là n + 1 = 1). Việc xếp ghế cho các ông là một song ánh F từ tập các ghế vào tập các ông với quy ước F(i) = j có nghĩa là ghế số i được xếp cho ông j. Khi đó số cần tìm U_n là số song ánh F từ tập {1, 2, ..., n} vào chính nó thỏa mãn điều kiện $F(i) \neq i$ và $F(i) \neq i + 1$ với mọi i = 1, 2, ..., n.

Các bất đẳng thức này gợi ý cho ta xác định n tính chất P_i {F(i) = i} và n tính chất Q_i {F(i) = i + 1} trên tập hợp các song ánh F đang xét. Như thế số cần tìm U_n là số $\overline{N}$ trong nguyên lý bù trừ của 2n tính chất này. Trong toán học, số U_n được gọi là *số phân bố*. Theo nguyên lý bù trừ, nó được tính bằng:

$$U_n = \overline{N} = N - N_1 + N_2 - ... + (-1)^{2n} N_{2n}$$

Trong đó N = n! (số các song ánh của n phần tử) và N_k là tổng tất cả các song ánh thỏa mãn k tính chất lấy ra từ 2n tính chất đang xét.

Chú ý:

Vì F là song ánh nên không thể đồng thời P_i, Q_i hoặc Q_i, P_{i+1} cùng được thỏa mãn, vì thế việc lấy ra k tính chất từ 2n tính chất cần phải thỏa mãn điều kiện P_i, Q_i hoặc Q_i, P_{i+1} không cùng được lấy ra.

Gọi số các cách lấy này là g(2n, k) (nói riêng, g(2n, k) = 0 khi k > n). Với mỗi cách lấy ra k tính chất như vậy ($k \le n$), ta có (n − k)! song ánh thỏa mãn chúng.

Từ đó nhận được:

$$N_k = g(2n, k)(n-k)! \text{ và}$$

$$U_n = n! - g(2n,1)(n-1)! + g(2n,2)(n-2)! - ... + (-1)^n g(2n,n)$$

Bây giờ bài toán được dẫn về việc tính các hệ số g(2n, k). Xếp các tính chất đang xét trên vòng tròn theo thứ tự P_1, Q_1, P_2, Q_2, ..., P_n, Q_n, ta thấy rằng g(2n, k) chính là cách lấy ra k điểm từ 2n điểm xếp trên vòng tròn sao cho không có hai điểm nào kề nhau cùng được lấy ra.

Để giải bài toán này ta lại phân thành hai giai đoạn: đầu tiên giải quyết bài toán với các điểm xếp trên đường thẳng, sau đó giải quyết bài toán với các điểm xếp trên vòng tròn bằng cách đưa về trường hợp các điểm xếp trên đường thẳng. Cuối cùng ta nhận được $g(2n,k) = \frac{2n}{2n-k} C_{2n-k}^{k}$ với k = 1, 2, ..., n (việc giải các bài toán này dành cho bạn đọc xem như bài tập).

Thay các giá trị g(2n, k) vừa tính vào công thức xác định U_n, ta được:

$$U_n = n! - \frac{2n}{2n-1} C_{2n-1}^{1}(n-1)! + \frac{2n}{2n-2} C_{2n-2}^{2}(n-2)! - ... + (-1)^n \frac{2n}{n} C_n^n$$

Dưới đây là một số giá trị của U_n được tính từ công thức trên:

n	2	3	4	5	6	7	8	9	10
U_n	0	1	2	13	80	579	4738	43387	439792

2.2.6. Kỹ thuật quy nạp

Giống như những bài toán khác, bài toán đếm có thể dùng quy nạp để giải, nghĩa là dẫn bài toán đang xét về những bài toán tương tự với kích thước nhỏ hơn.

Thông thường các công thức đếm xác định số cấu hình trực tiếp từ những giá trị tham số đầu vào qua một số hữu hạn các phép tính (ta gọi các công thức này là những *công thức tường minh* hay *công thức trực tiếp*), chẳng hạn các công thức tính các số chỉnh hợp A_n^m, các số tổ hợp C_n^m, các số mất thứ tự D_n, các số phân bố U_n, ... Tuy nhiên việc tìm các công thức này không phải dễ dàng.

Người ta nhận thấy rằng có một lớp các bài toán đếm mà việc tìm kiếm một lời giải dưới dạng một công thức tường minh là rất khó khăn hoặc lâm vào bế tắc, trong khi việc tìm một công thức dẫn kết quả của bài toán đang xét về kết quả của các bài toán tương tự với kích thước nhỏ hơn lại đơn giản, dễ tìm. Nhờ công thức làm "cầu nối" này, từ một vài giá trị ban đầu (kích thước nhỏ, được tính trực tiếp), ta có thể tính mọi giá trị còn lại khác. Một công thức như vậy được gọi là *công thức truy hồi* hay *công thức đệ quy*.

Do tính kế thừa và đơn giản, công thức truy hồi rất thích hợp với việc lập trình trên máy tính, nó cho phép giảm đáng kể độ phức tạp cũng như gia tăng độ ổn định của quá trình tính toán.

Dưới đây là một số ví dụ minh họa việc dùng quy nạp để giải bài toán đếm. Qua các ví dụ này, bạn đọc cũng có thể thấy những đặc điểm của một lớp các bài toán đếm mà việc giải bằng quy nạp dẫn đến những công thức truy hồi đơn giản, trong khi công thức trực tiếp nếu tìm được, cũng phức tạp hơn nhiều.

Ví dụ 1.

Trên mặt phẳng kẻ n đường thẳng sao cho không có hai đường nào song song và không có ba đường nào đồng quy. Đếm số phần mặt phẳng được chia.

Giải:

Lúc ban đầu, khi không có đường thẳng nào, mặt phẳng gồm 1 phần, khi kẻ một đường thẳng, số phần mặt phẳng là 2, khi kẻ thêm đường thẳng thứ hai cắt đường thẳng này, số phần mặt phẳng là 4, vẽ thêm đường thẳng thứ ba, cắt tất cả hai đường thẳng cũ và không đi qua các giao điểm cũ, số phần mặt phẳng là 7, ... Bằng cách đếm trực tiếp số phần mặt phẳng được chia trong các trường hợp n = 1, 2, 3, ... ta khó nhận diện được môt quy luật nào để có một lập luận khả dĩ dẫn đến một công thức tường minh.

Thay vì gắng tìm một công thức như vậy, ta thử tìm một "cầu nối" giữa kết quả của việc kẻ n đường thẳng với kết quả trước đấy của việc kẻ n − 1 đường thẳng. Ký hiệu S_{n-1} là số phần mặt phẳng được chia khi kẻ n − 1 đường thẳng (thỏa mãn các điều kiện đã nêu). Bây giờ ta kẻ thêm đường thẳng thứ n sao cho đường thẳng này cắt tất cả các đường thẳng cũ và không đi qua các giao điểm cũ. Khi đó số phần mặt phẳng được chia là S_n. Nhận xét rằng, đường thẳng mới vẽ cắt tất cả n − 1 đường thẳng cũ tại những giao điểm riêng biệt, từ đó nhận được số giao điểm mới là n − 1 và điều này kéo theo số phần mặt phẳng được thêm là n. Như vậy giữa giá trị S_n và S_{n-1} có mối liên hệ đơn giản sau:

$$S_n = S_{n-1} + n \qquad (n = 1, 2, ...)$$

Công thức trên là công thức truy hồi, nó là mối liên hệ cho phép xác định S_n từ S_{n-1}. Để xác định được mọi S_n, trong trường hợp này cần một giá trị ban đầu mà việc xác định nó là trực tiếp $S_0 = 1$. Từ giá trị này, ta thực hiện n lần các phép toán trong công thức truy hồi (trong ví dụ này chỉ gồm một phép cộng), sẽ nhận được giá trị cần tính S_n. Chẳng hạn bảng dưới đây cho ta quá trình tính từ S_1 đến S_9 bằng 9 lần thực hiện các phép cộng:

n	0	1	2	3	4	5	6	7	8	9
S_n	1	2	4	7	11	16	22	29	37	46

Trong công thức truy hồi lần lượt cho n = 1, 2, ... và chú ý giá trị ban đầu $S_0 = 1$, ta nhận được các đẳng thức:

$$S_0 = 1$$
$$S_1 = S_0 + 1$$
$$S_2 = S_1 + 2$$
$$. . . . .$$
$$S_n = S_{n-1} + n$$

Cộng tất cả các đẳng thức này theo từng vế và ước lượng các số hạng giống nhau ở hai vế, ta nhận được:

$$S_n = 1 + 1 + 2 + ... + n = 1 + \frac{n(n+1)}{2}$$

Đây là công thức tường minh tính số phần mặt phẳng khi kẻ n đường thẳng thỏa mãn các điều kiện đã nêu. Quá trình đi từ công thức truy hồi đến công thức tường minh được gọi là quá trình *khử quy nạp*.

Nói chung, các công thức tường minh phức tạp hơn công thức truy hồi vì nó là kết quả tích lũy của nhiều bước quy nạp. Trong ví dụ này, công thức truy hồi chỉ chứa một phép cộng, trong khi công thức tường minh chứa cả phép nhân và phép chia.

Ví dụ 2. Trong ví dụ 2 của mục 2.2.5, ta đã giải bài toán đếm số mất thứ tự bằng nguyên lý bù trừ, ở đây ta giải lại bài toán này bằng phương pháp quy nạp.

Theo định nghĩa, một mất thứ tự của n phần tử là một cách bỏ n lá thư vào n phong bì sao cho tất các lá thư đều sai địa chỉ. Cấu hình này có thể biểu diễn như là một hoán vị $(a_1, a_2, ..., a_n)$ của tập {1, 2, ..., n} thỏa mãn điều kiện $a_i \neq i$ với mọi i = 1, 2, ..., n. Ta sẽ đếm số hoán vị này tương ứng với a_1 đã được chọn với một giá trị $k \neq 1$ nào đó. Xét hai trường hợp tùy thuộc vào $a_k = 1$ và $a_k \neq 1$ như sau:

- Nếu $a_k = 1$, khi đó các thành phần còn lại (khác a_1 và khác a_k) sẽ được xác định như một mất thứ tự của n − 2 phần tử. Theo giả thiết quy nạp, số các mất thứ tự trong trường hợp này là D_{n-2}
- Nếu $a_k \neq 1$, khi đó các thành phần khác a_1 sẽ được xác định như một mất thứ tự của n−1 phần tử. Theo giả thiết quy nạp, số các mất thứ tự trong trường hợp này là D_{n-1}

Theo nguyên lý cộng, số mất thứ tự với a_1 đã được chọn là $D_{n-1}+D_{n-2}$. Vì có n − 1 khả năng chọn a_1 nên theo nguyên lý nhân, số mất thứ tự của n phần tử sẽ là $(n-1)(D_{n-1}+ D_{n-2})$ và ta nhận được công thức truy hồi:

$$D_n = (n-1)(D_{n-1} + D_{n-2}),\ n > 2.$$

Để xác định được giá trị của mọi D_n, trong trường hợp này ta cần có hai giá trị ban đầu. Điều này dễ dàng thực hiện nhờ tìm trực tiếp $D_1 = 0$, $D_2 = 1$. Các giá trị còn lại được tính từ công thức truy hồi sau một số lần lặp tương ứng.

Từ công thức này, ta có thể khử nó để nhận được công thức trực tiếp đã tìm thấy bằng nguyên lý bù trừ. Quá trình này dành cho bạn đọc xem như bài tập.

Điều đáng nói ở đây là việc tính D_n từ công thức truy hồi đơn giản hơn nhiều với việc dùng công thức trực tiếp. Để tính D_n từ công thức trực tiếp, ta phải thực hiện các phép cộng, trừ, nhân, chia các số nguyên đôi khi rất lớn, trong khi dùng công thức truy hồi ta chỉ cần thực hiện các phép cộng và nhân các số nguyên không lớn lắm. Sự tiện lợi của các công thức truy hồi càng được thấy rõ khi giải trên máy tính: việc lập trình tính theo công thức trực tiếp thường gặp phải lỗi tràn số và sai số lớn do phải nhân, chia các số nguyên lớn, trong khi lập trình theo công thức truy hồi kết quả được ổn định và chính xác hơn nhiều. Anh chị có thể tính các giá trị của các D_n theo công thức truy hồi rồi điền vào bảng để được kết quả đã nêu trong ví dụ 2, mục 2.2.5, sau đó thử tính lại các giá trị này bằng công thức trực tiếp để so sánh.

Ví dụ dưới đây là một bài toán do nhà toán học Ý Fibonaci đưa ra vào thế kỷ 13, cho thấy có những trường hợp công thức truy hồi rất đơn giản nhưng công thức trực tiếp lại rất phức tạp.

Ví dụ 3 (dãy Fibonaci).

Một cặp thỏ mới sinh (một đực, một cái) được thả lên một hòn đảo. Giả sử khi đầy hai tháng tuổi, chúng bắt đầu sinh sản, mỗi tháng một cặp thỏ con (một đực, một cái). Các cặp thỏ con này cũng sinh trưởng và sinh sản theo quy tắc trên. Hỏi số thỏ trên đảo ở tháng thứ n với giả thiết không có con nào chết?

Giải:

Gọi F_n là số cặp thỏ ở tháng thứ n. Tại tháng thứ nhất và tháng thứ hai không có cặp thỏ nào được sinh nên $F_1 = F_2 = 1$. Bắt đầu từ n = 3, số cặp thỏ của tháng này bằng số cặp thỏ của tháng trước là F_{n-1} cộng với số cặp thỏ mới sinh, số này bằng số cặp thỏ của tháng trước nữa tức là F_{n-2} (vì đến lúc này, số cặp thỏ đó đã có ít nhất hai tháng tuổi).

Ta nhận được công thức truy hồi:

$$F_n = F_{n-1} + F_{n-2}\ ,\ n = 3, 4, \ldots$$

Từ công thức này và từ hai giá trị ban đầu ta tính được mọi giá trị F_n chỉ bằng phép cộng. Dãy số F_1, F_2, ..., được gọi là *dãy Fibonaci*. Dưới đây là một số số hạng đầu tiên của dãy: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ..

Khi khử quy nạp từ công thức truy hồi đơn giản này, người ta nhận được công thức trực tiếp tính số hạng F_n theo n:

$$F_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right].$$

Công thức trên cho thấy nó chỉ mang tính lý thuyết chứ không mang tính áp dụng, để tính giá trị nguyên F_n theo công thức này, ta phải dùng các phép toán khai căn và mũ, cho các giá trị trung gian là vô tỷ. Chẳng hạn để tính F_5, nếu dùng công thức truy hồi ta chỉ cần thực hiện 3 phép cộng 1+1 = 2, 1 + 2 = 3, 2 + 3 = 5, trong khi dùng công thức trực tiếp ta phải tính một biểu thức phức tạp (thay n = 5 vào công thức này) để nhận được một giá trị gần đúng.

Bạn đọc nào muốn tìm hiểu thêm về kỹ thuật quy nạp và các phương pháp khử chúng có thể tham khảo [2].

2.2.7. Đếm bằng lập trình

Như đã thấy trong các mục trước, bên cạnh những bài toán đếm đơn giản, có những bài toán đếm mà việc tìm một công thức cho nó (ngay cả truy hồi) là rất khó khăn, thậm chí lâm vào bế tắc vì còn chưa biết có tồn tại một công thức như thế hay không. Trong những trường hợp như vậy, thay vì phải đi tìm công thức, người ta xây dựng một thuật toán cho phép "vét cạn" các cấu hình cần đếm. Việc liệt kê các cấu hình cần phải đảm bảo *không bỏ sót* và *không trùng lặp*. Mỗi lần đưa ra được một cấu hình ta lại đếm chúng. Như vậy, kết quả đếm sẽ được tìm thấy khi việc liệt kê kết thúc. Công việc "nhàm chán" này được giao cho máy tính thi hành bằng việc cài đặt một chương trình thực hiện thuật toán liệt kê đã xây dựng.

Ý tưởng đếm bằng liệt kê thật ra là cách giải quyết tự nhiên nhất mà con người đã thực hiện từ thời thượng cổ, tuy nhiên nó chỉ dùng trong những trường hợp cụ thể, đơn giản và kích thước nhỏ. Sau này khi lý thuyết tổ hợp phát triển, việc đếm bằng liệt kê được dùng để tạo ra những giá trị ban đầu, hoặc hỗ trợ cho việc phát hiện cũng như kiểm chứng các công thức đếm. Khi chưa có máy tính, công việc này được thực hiện bằng thủ công nên kết quả rất hạn chế. Tình hình trở nên thay đổi khi máy tính ra đời. Nhờ sự phát triển của máy tính mà phương pháp lập trình ngày càng được dùng rộng rãi, nó cho phép giải quyết những bài toán khó khi chưa tìm ra được một giải pháp truyền thống khả dĩ nào. Hạn chế của phương pháp này là tính khả thi thấp do sự bùng nổ của các cấu hình tổ hợp. Để đếm bằng lập trình, ta phải đợi cho đến khi chương trình chạy xong, và việc có chờ đợi được không phụ thuộc vào nhiều yếu tố như kích thước của bài toán, tốc độ của máy tính, độ phức tạp của thuật toán, ... Chẳng hạn, để đếm những cấu hình là các tập con của một tập nào đấy, thỏa mãn một số điều kiện đã cho, ta tiến hành duyệt tất cả các tập con này và kiểm tra các điều kiện cần thỏa mãn đối với mỗi cấu hình được duyệt. Giả sử tập được xét gồm 64 phần tử (không lớn lắm) và máy tính mất một giây (khá nhanh) để kiểm tra xong một cấu hình, khi đó chương trình phải duyệt tất cả 2^{64} = 18 446 744 073 709 551 616 tập con và ta phải đợi khoảng 500 tỷ năm để được kết quả. Mặc dù còn hạn chế như vậy, nhưng việc lập trình liệt kê là một kỹ thuật không thể bỏ qua để giải các bài toán đếm trong những trường hợp chưa tìm được một giải pháp nào, chí ít nó cũng giải được với những bài toán có kích thước thích hợp, hỗ trợ hay phát hiện những sai lầm trong quá trình tìm kiếm lời giải tổng quát. Ngoài ra, chính những khó khăn trong việc giải quyết tính khả thi đã thúc đẩy con người đi sâu nghiên cứu hơn nữa những thuật toán hiệu quả cũng như kỹ thuật chế tạo những máy tính có tốc độ cao.

Việc liệt kê các cấu hình sao cho không bỏ sót và không trùng lặp sẽ được bàn đến một cách hệ thống trong mục bài toán liệt kê. Trong những trường hợp đơn giản như liệt kê các số nguyên, các ký tự, ... việc viết một chương trình để đếm là khá dễ dàng.

Để làm ví dụ, ta giải lại bài toán cho trong ví dụ 1 của mục 2.2.5 (nguyên lý bù trừ): "đếm số các số nguyên từ 1 đến 10000 thỏa mãn điều kiện không chia hết cho các số 3, 4 và 7". Thuật toán đếm đơn giản như sau:

- Duyệt các số nguyên i từ 1 đến 10000
- Với mỗi số nguyên i được duyệt, kiểm tra tính chia hết của i cho các số 3, 4 và 7, nếu cả ba phép chia này có dư thì đếm i
- Khi kết thúc duyệt, ta nhận được giá trị đếm.

Việc viết cụ thể chương trình (trên C hoặc Pascal) thực hiện thuật toán trên được dành cho bạn đọc xem như bài tập. Bạn có thể chạy chương trình để thấy lại kết quả đã giải bằng nguyên lý bù trừ. Điều thú vị là chương trình vừa viết có thể dễ dàng sửa đổi để giải quyết một lớp các bài toán tương tự, như thay đổi miền giá trị các số nguyên cần đếm, thay đổi các giá trị 3, 4, 7 bằng các giá trị khác, trong khi việc thay đổi các lập luận trong việc giải bằng nguyên lý bù trừ cho phù hợp với những điều kiện mới lại không dễ dàng.

2.3. Bài toán tồn tại

2.3.1. Giới thiệu bài toán

Bài toán tồn tại quan tâm đến việc một cấu hình, thỏa mãn điều kiện nào đấy, có tồn tại hay không? Như thế bài toán tồn tại xem như được giải quyết nếu ta khẳng định hoặc phủ định việc có cấu hình. Tuy nhiên cả hai hướng tìm kiếm đều gặp khó khăn: trong khi chưa tìm thấy được một cấu hình nào, ta cũng chưa có một cơ sở nào để phủ định sự tồn tại của nó. Trong bài này, ta sẽ tìm hiểu một số phương pháp thường dùng để giải một số bài toán tồn tại đơn giản. Ngoài ra, bài này cũng giới thiệu một số bài toán tồn tại nổi tiếng trên thế giới, qua đấy, bạn đọc cũng thấy được vai trò của máy tính trong việc giải quyết những bài toán này.

2.3.2. Phương pháp phản chứng

Trong nhiều bài toán cần khẳng định sự tồn tại của cấu hình, giả thiết thường rất "mờ nhạt", khó có thể căn cứ vào những dữ kiện đã cho để chỉ ra một cấu hình cụ thể. Trong những trường hợp như vậy, phương pháp phản chứng thường được nghĩ đến. Như đã trình bày trong phần giới thiệu các ứng dụng của lý thuyết lôgic (mục 1.2.4.1 bài 1), phương pháp phản chứng cho ta một cách gián tiếp chỉ ra sự tồn tại cấu hình mà không cần xây dựng cụ thể cấu hình đó. Theo cách này, từ nội dung bài toán, ta cố gắng thiết lập một mệnh đề sao cho việc chứng minh mệnh đề này đúng kéo theo sự tồn tại của cấu hình và sau đó dùng phản chứng để chứng minh mệnh đề này đúng.

Để minh họa, ta xét một số ví dụ dưới đây.

Ví dụ 1.

Cho 7 đoạn thẳng mà độ dài của chúng nằm trong khoảng từ 10 đến 100 (tính theo đơn vị độ dài nào đấy). Chứng minh rằng luôn tìm được 3 đoạn để có thể ghép thành một tam giác.

Giải.

Chú ý rằng, để 3 đoạn thẳng ghép được thành tam giác, cần và đủ là độ dài của đoạn dài nhất phải nhỏ hơn tổng độ dài của hai đoạn kia. Điều này gợi ý ta sắp lại 7 đoạn này theo chiều tăng dần của độ dài và gọi a_1, a_2, ..., a_7 là dãy độ dài sau khi được sắp. Khi đó, theo giả thiết ta có dãy bất đẳng thức $10 \le a_1 \le a_2 \le \ldots \le a_7 \le 100$.

Việc tìm được 3 đoạn thỏa mãn điều kiện đã nêu được suy ra từ giá trị đúng của mệnh đề:

$$\exists i(a_{i+2} < a_i + a_{i+1}) \quad i = 1, 2, \ldots, 5$$

Bây giờ ta chứng minh mệnh đề này đúng bằng cách dùng phản chứng. Giả sử ngược lại, ta có mệnh đề phủ định của nó là đúng:

$$\forall i(a_{i+2} \ge a_i + a_{i+1}) \quad i = 1, 2, \ldots, 5$$

Giả sử mạng đó là tồn tại. Đếm tất cả kênh nối của mạng này bằng nhận xét khi đếm các kênh nối với từng máy rồi cộng lại, mỗi kênh sẽ được tính 2 lần, ta được số kênh nối là $31 \times 5/2$. Kết quả thu được không phải là số nguyên và đấy là điều vô lý.

Một hệ quả của việc chứng minh phản chứng là nguyên lý được trình bày trong mục dưới đây.

2.3.3. Nguyên lý Dirichlet

Định lý. Nếu ta chia một tập hợp n phần tử thành nhiều lắm là k lớp thì chắc chắn tìm được một lớp có số phần tử ít nhất là n/k.

Chứng minh bằng phản chứng: Giả sử mọi lớp đều có số phần tử nhỏ hơn n/k thì tất cả các lớp cộng lại có số phần tử nhỏ hơn n.

Nguyên lý trên hay được dùng trong trường hợp riêng với phát biểu: "nếu nhốt n con chim bồ câu vào ít hơn n lồng thì chắc chắn có lồng chứa từ 2 con trở lên". Vì thế nguyên lý này còn có tên gọi là *nguyên lý lồng chim bồ câu*.

Hiện nay, định lý vừa phát biểu được mang tên *nguyên lý Dirichlet*, tên nhà toán học Đức vào thế kỷ 19, để kỷ niệm nhiều công trình xuất sắc của ông, trong đó có nhiều chứng minh, ông đã vận dụng một cách khéo léo nguyên lý này.

Nguyên lý Dirichlet tưởng chừng như đơn giản nhưng nó lại tỏ ra khá hiệu quả trong việc chứng minh sự tồn tại của một cấu hình. Việc áp dụng nguyên lý này khá đa dạng và linh hoạt. Có những bài toán việc dùng nguyên lý này được thấy một cách hiển nhiên (nhiều khi không cần nói tên nguyên lý), nhưng cũng có bài toán cần phải tổ chức một cách khôn khéo để có thể áp dụng được nguyên lý này. Dưới đây là một số ví dụ minh họa cho việc dùng nguyên lý Dirichlet.

Ví dụ 1.

Khẳng định rằng, trong 367 người bất kỳ luôn tìm được ít nhất hai người có cùng sinh nhật.

Giải:

Thật vậy, một năm dương lịch có nhiều nhất 366 ngày (kể cả năm nhuận). Số này được phân bổ cho 367 người vì vậy có ít nhất một ngày chung cho hai người trở lên.

Ví dụ 2.

Một lớp mẫu giáo gồm 33 cháu tập tô mầu một bức tranh gồm 5 đồ vật bằng bút chì xanh/đỏ (mỗi đồ vật tô thuần nhất một mầu). Chứng minh rằng luôn tìm được ít nhất hai cháu tô mầu giống hệt nhau.

Giải.

Một cách tô mầu 5 đồ vật bằng bút chì xanh/đỏ là một chỉnh hợp lặp chập 5 của 2, vì thế số cách tô mầu bằng $2^5 = 32$. Có 32 cách tô mầu cho 33 cháu, vậy ít nhất có hai cháu có cùng cách tô?

Ví dụ 3.

Bài thi các môn học trong một trường đại học được chấm theo thang điểm là các số nguyên từ 0 đến 20. Hỏi một lớp cần phải có ít nhất bao nhiêu sinh viên để đảm bảo trong mọi môn thi đều tìm được ít nhất 5 sinh viên của lớp nhận cùng điểm thi?

Giải.

Với mỗi môn thi, phân hoạch các sinh viên của lớp đang xét sao cho hai sinh viên thuộc cùng một lớp phân hoạch khi và chỉ khi hai người này nhận cùng điểm thi. Vì thang điểm có 21 bậc nên phân hoạch này có nhiều nhất 21 lớp.

Để chắc chắn trong mọi tình huống đều tìm được một lớp phân hoạch có ít nhất 5 người, theo nguyên lý Dirichlet, số sinh viên của lớp phải lớn hơn $21 \times 4 = 84$, tức là ít nhất bằng 85.

Ví dụ 4.

Một đội bóng phải hoàn thành 19 trận đấu trong 27 ngày, trong đó mỗi ngày đấu không quá một trận. Chứng minh rằng, dù xếp lịch thế nào, đội bóng cũng phải đấu ít nhất 3 trận liền.

Giải.

Theo giả thiết, đội bóng có 8 ngày nghỉ. Những ngày này tạo thành nhiều nhất là 9 khoảng trống trên tập 27 ngày. Các trận đấu liền nhau được phân bố vào các khoảng trống này. Theo nguyên lý Dirichlet, tìm được ít nhất một khoảng trống chứa ít nhất 19/9, nghĩa là 3 trận đấu.

Các ví dụ trên là những bài đơn giản, trong đó việc áp dụng Dirichlet là dễ thấy. Các ví dụ dưới đây là những bài toán khó hơn và việc vận dụng nguyên lý Dirichlet nhiều khi cho những kết quả bất ngờ.

Ví dụ 5.

Chứng minh rằng trong một phòng họp, bao giờ cũng tìm được hai người có số người quen trong số những người dự họp là bằng nhau.

Giải.

Bài này hầu như không có giả thiết gì ngoại trừ việc công nhận mối quan hệ quen nhau là đối xứng.

Gọi số người dự họp là n. Chia những người dự họp thành những nhóm theo quy tắc hai người có số người quen bằng nhau được xếp vào một nhóm và đánh số các nhóm theo số người quen này (nhóm số 1 gồm những người quen một người, nhóm số 2 gồm những người quen hai người,...).

Việc chia nhóm như trên là một phân hoạch của tập hợp những người dự họp. Nếu mọi người dự họp đều quen ít nhất một người thì số hiệu nhóm thấp nhất là 1 và cao nhất là $n - 1$, nếu trái lại, có một người dự họp không quen ai cả thì số hiệu nhóm thấp nhất là 0 và cao nhất là $n - 2$. Trong cả hai tình huống ta đều có số nhóm nhiều nhất là $n - 1$, theo nguyên lý Dirichlet, bao giờ cũng tìm được một nhóm có ít nhất hai người.

Bài toán trên có thể phát biểu dưới dạng hình học như sau: Cho một số hữu hạn điểm, trong đó có một số điểm được nối với nhau bởi các đoạn thẳng. Khi đó bao giờ cũng tìm được hai điểm có cùng số cạnh nối đi ra từ chúng.

Ví dụ 6.

Chứng minh rằng trong $n + 1$ số nguyên dương, mỗi số không lớn hơn $2n$, bao giờ cũng tìm được hai số sao cho số này chia hết cho số kia.

Giải.

Gọi các số đã cho là $a_1, a_2, ..., a_{n+1}$. Phân tích các số nguyên này theo nhân tử 2, ta được:

$$a_i = 2^{k_i} q_i, \quad i = 1, 2, ..., n + 1$$

trong đó k_i là số nguyên không âm, q_i là số lẻ. Vì a_i không lớn hơn 2n nên n + 1 số lẻ $q_1, q_2, ..., q_{n+1}$ chỉ có thể nhận n giá trị lẻ trong phạm vi từ 1 đến 2n. Theo nguyên lý Dirichlet, phải có hai số lẻ q_i và q_j nào đấy có cùng giá trị q. Khi đó $a_i = 2^{k_i} q$, $a_j = 2^{k_j} q$ và một trong hai số này chia hết cho số còn lại.

Ví dụ 7.

Chứng minh rằng từ 100 số nguyên dương bất kỳ, ta có thể chọn ra một số các số nào đấy để tổng của chúng chia hết cho 100.

Giải.

Gọi các số nguyên đã cho là $a_1, a_2, ..., a_{100}$. Lập các tổng $s_1 = a_1$, $s_2 = a_1 + a_2$, ..., $s_{100} = a_1 + a_2 + ... + a_{100}$, ta được 100 số nguyên dương tăng dần $s_1, s_2, ..., s_{100}$.

Nếu có một tổng s_i nào đó chia hết cho 100 thì bài toán được chứng minh.

Bây giờ giả thiết không có một s_i nào chia hết cho 100, điều này có nghĩa 100 số này được phân bố vào 99 lớp đồng dư theo môđun 100 (lớp chia 100 dư 1, lớp chia 100 dư 2, ..., lớp chia 100 dư 99). Theo nguyên lý Dirichlet, bao giờ cũng tìm được s_i, s_j với $i < j$ nào đó thuộc cùng một lớp. Khi đó ta có $s_j - s_i = a_{i+1} + a_{i+2} + ... + a_j$ chia hết cho 100 là tổng cần tìm.

Trong lập luận trên có thể thay giá trị 100 bằng bất cứ giá trị nguyên dương nào và ta được kết quả tổng quát: "từ n số nguyên dương bất kỳ, ta có thể chọn ra một số các số nào đấy để tổng của chúng chia hết cho n".

Ví dụ 8.

Trong mặt phẳng cho 6 điểm được nối với nhau từng đôi một bởi các cung màu xanh hoặc màu đỏ. Chứng minh rằng luôn tìm được 3 điểm sao cho các cung nối chúng có cùng một màu.

Giải.

Gọi một điểm nào đó là P. Từ P có 5 cung đi ra nối với 5 điểm còn lại. Theo nguyên lý Dirichlet trong số các cung này phải có 3 cung cùng màu, chẳng hạn màu xanh. Gọi 3 điểm được nối với cung này là A, B, C. Nếu cả 3 cung AB, AC, BC đều có màu đỏ thì bài toán được chứng minh, trái lại có một cung nào đó màu xanh, chẳng hạn AB có màu xanh, khi đó cả 3 cung PA, PB, AB đều có màu xanh và bài toán được chứng minh.

Trong bài toán trên, ta xem mỗi điểm như là một người và màu của cung nối hai điểm mô tả hai trạng thái quen/không quen giữa hai người. Khi đó ta nhận được một kết quả thú vị về mối quan hệ quen nhau, mặc dù không biết mối quan hệ đó như thế nào: "trong 6 người bất kỳ, luôn tìm được 3 người, hoặc tất cả đều quen nhau, hoặc không ai quen ai cả".

Vấn đề đặt ra trong ví dụ 8 được mở rộng cho việc xây dựng các số Ramsey, mang tên nhà toán học người Anh, người đã chứng minh được định lý nổi tiếng mang cùng tên, được xem như sự tổng quát hóa nguyên lý Dirichlet.

Để xem thêm về định lý Ramsey, bạn đọc có thể tham khảo [2].

2.3.4. Hệ đại diện phân biệt

Trong nhiều tình huống, sự tồn tại của cấu hình còn phụ thuộc vào những ràng buộc nằm ngoài các dữ kiện của định nghĩa. Trong những trường hợp như vậy, người ta quan tâm đến những điều kiện đảm bảo cho cấu hình là tồn tại.

Mục này giới thiệu một cấu hình tổ hợp mà sự tồn tại của nó có nhiều ứng dụng trong thực tế.

Xét một họ tập $\{S_1, S_2, ..., S_m\}$ nào đấy (các S_i không nhất thiết khác nhau). Ta gọi một bộ có thứ tự $(a_1, a_2, ..., a_m)$ là một *hệ đại diện phân biệt* của họ tập đã cho nếu a_i thuộc S_i và $a_i \neq a_j$ $(i \neq j)$. Thành phần a_i của hệ được gọi là *đại diện* cho tập S_i.

Ví dụ cho họ $S_1 = \{2, 5\}$, $S_2 = \{2, 5\}$, $S_3 = \{1, 2, 3, 4\}$, $S_4 = \{1, 2, 5\}$. Khi đó có thể thấy (2, 5, 3, 1) là một hệ đại diện phân biệt của họ đã cho. Cũng có thể thấy một hệ đại diện phân biệt khác là (5, 2, 4, 1). Thay trong họ trên, tập $S_4 = \{2, 5\}$ (giữ nguyên các tập khác), ta được một họ không có hệ đại diện phân biệt nào vì không đủ 3 đại diện khác nhau cho 3 tập $S_1 = S_2 = S_3 = \{2, 5\}$.

Ví dụ trên cho thấy việc có hay không một hệ đại diện phân biệt của một họ tập cho trước còn phụ thuộc vào tương quan giữa các tập trong họ, nói chung, một cách trực giác, ta thấy rằng nếu có một hợp các tập trong họ có ít phần tử quá thì họ này có thể không có một hệ đại diện phân biệt nào.

P. Hall đã chứng minh định lý dưới đây, cho một điều kiện cần và đủ để một họ tập hợp có hệ đại diện phân biệt.

Định lý Hall. Để họ tập $\{S_1, S_2, ..., S_m\}$ có hệ đại diện phân biệt, cần và đủ là các bất đẳng thức sau được thỏa mãn:

$$N(S_{i_1} \cup S_{i_2} \cup ... \cup S_{i_k}) \geq k$$

trong đó $(i_1, i_2, ..., i_k)$ là một tổ hợp chập k của tập chỉ số $\{1, 2, ..., m\}$, $1 \leq k \leq m$.

Điều kiện cần được chứng minh tương đối hiển nhiên: nếu có một bất đẳng thức ngược lại

$$N(S_{i_1} \cup S_{i_2} \cup ... \cup S_{i_k}) < k$$

với một nhóm tập $S_{i_1}, S_{i_2}, ..., S_{i_k}$ nào đó thì không thể tìm được k đại diện khác nhau cho các tập này.

Để chứng minh điều kiện đủ, Hall đã đưa ra một đánh giá cận dưới của số các hệ đại diện phân biệt. Gọi t là số phần tử ít nhất của các tập đang xét, nếu $t \leq m$ thì cận dưới bằng t!, nếu $t > m$ thì cận dưới bằng $t!/(t - m)!$. Giá trị dương của các cận dưới này chứng tỏ sự tồn tại của hệ đại diện phân biệt (có thể xem chứng minh chi tiết trong [2]).

Chú ý: Mỗi tập con khác rỗng k phần tử của tập chỉ số $\{1, 2, ..., m\}$ tương ứng với một bất đẳng thức trong định lý Hall, do đó số lượng bất đẳng thức này bằng $2^m - 1$. Đây là một con số rất lớn, vì thế việc thử tất cả các bất đẳng thức của điều kiện Hall là không khả thi. Trên thực tế, người ta có những thuật toán hiệu quả hơn để kiểm tra sự tồn tại của các hệ đại diện phân biệt.

Sự tồn tại của hệ đại diện phân biệt có nhiều ý nghĩa trong thực tế. Dưới đây ta xét một số bài toán mà việc giải quyết nó đưa về việc xây dựng hệ đại diện phân biệt.

Bài toán đám cưới vùng quê. Tại một làng quê có m chàng trai đến tuổi lấy vợ. Với mỗi chàng trai i, ta biết tập S_i các cô gái mà chàng ta thích. Hỏi rằng có thể ghép mỗi cô cho mỗi chàng để chàng nào cũng vừa ý hay không? Câu trả lời là tùy vào việc có hệ đại diện phân biệt của họ $\{S_i\}$ hay không. Trong trường hợp tồn tại, mỗi hệ đại diện phân biệt của họ này sẽ tương ứng với một cách ghép mong muốn.

Bài toán phân công. Có m người và một số công việc. Biết S_i là tập công việc phù hợp với người i. Hỏi rằng có thể phân công mỗi người mỗi việc để mọi người đều phù hợp với công việc của mình hay không? Rõ ràng một cách phân công như thế là một hệ đại diện phân biệt của họ $\{S_i\}$ và bài toán được dẫn về xét sự tồn tại của hệ đại diện phân biệt.

Bài toán nối mạch. Một hệ thống truyền tin tại một thời điểm có m đầu vào đòi hỏi được nối với các đầu ra thông qua một hệ thống các mạch nối, Mỗi mạch nối được dùng phải trong trạng thái rỗi, nghĩa là chưa phục vụ cho đầu vào nào. Giả thiết biết S_i là tập các mạch nối còn rỗi mà theo đó đầu vào i có thể chuyển tới một đầu ra. Nếu họ các S_i có hệ đại diện phân biệt thì đòi hỏi của m đầu vào được thỏa mãn: mỗi hệ đại diện này sẽ là một phương án nối mạch, trái lại ta phải chỉnh lại các hệ thống nối mạch để chọn cách nối khác.

Một số cấu hình đã biết có thể đưa về dạng hệ đại diện phân biệt như chỉ ra dưới đây:

- Một chỉnh hợp chập k của $S = \{1, 2, ..., n\}$ ($k \le n$), là một hệ đại diện phân biệt của họ $S_1 = S_2 = ... = S_k$ (= S). Nói riêng, một hoán vị của S là một hệ đại diện phân biệt của họ $S_1 = S_2 = ... = S_n$ (= S).
- Một mất thứ tự của $S = \{1, 2, ..., n\}$, tức là một hoán vị $a_1, a_2, ..., a_n$ của S thỏa mãn $a_i \neq i$ với mọi i (xem bài toán bỏ thư), là một hệ đại diện phân biệt của họ $S_1 = S - \{1\}, S_2 = S - \{2\}, ..., S_n = S - \{n\}$.
- Một phân bố của $S = \{1, 2, ..., n\}$, tức là một song ánh F từ S vào chính nó thỏa mãn $F(i) \neq i$ và $F(i) \neq i + 1$ (xem bài toán xếp khách của Lucas), là một hệ đại diện phân biệt của họ $S_1 = S - \{1, 2\}, S_2 = S - \{2, 3\}, ..., S_n = S - \{n, 1\}$.

2.3.5. Giải bài toán tồn tại bằng lập trình

Để trả lời câu hỏi “có hay không có cấu hình?” người ta phải chọn lựa một trong hai hướng: hoặc cố gắng tìm cách chỉ ra một cấu hình, hoặc cố gắng chứng minh cấu hình đó không tồn tại. Tuy nhiên tình huống thường gặp là nếu hướng này gặp khó khăn thì hướng kia cũng vậy. Nếu trước đây con người gặp bế tắc vì chỉ dựa vào sức mình để tính toán, chứng minh, ..., thì bây giờ người ta đã nghĩ đến sự hỗ trợ của máy tính. Theo cách này, thay vì những giải pháp truyền thống của suy luận toán học, người ta xây dựng một thuật toán cho phép “vét cạn” các cấu hình và thực thi nó bằng một chương trình viết trên máy tính. Lời giải của bài toán sẽ nhận được từ kết quả chạy chương trình: nếu trong quá trình chạy, chương trình đưa ra được một cấu hình nào đấy thì sự tồn tại của cấu hình được khẳng định, trái lại, nếu kết thúc chương trình mà ta không nhận được cấu hình nào thì việc không có cấu hình được chứng minh. Lịch sử toán học để lại nhiều bài toán khó thuộc lĩnh vực bài toán tồn tại mà quá trình tìm kiếm lời giải kéo dài hàng thế kỷ và nhiều bài toán loại này cuối cùng đã được giải quyết bằng sự hỗ trợ của máy tính.

Bằng việc lập trình cho máy tính, ta có thể tìm kiếm lời giải đồng thời theo cả hai hướng: khẳng định hoặc phủ định việc có cấu hình. Chẳng hạn trong trò chơi xếp chữ, ta có thể viết một chương trình duyệt tất cả các cách xếp và kiểm tra mỗi cách xếp được duyệt. Chương trình kết thúc khi tìm thấy một cách xếp thỏa mãn điều kiện đã nêu hoặc khi duyệt hết các cách xếp.

Khó khăn chính của phương pháp này là số cấu hình phải duyệt thường rất lớn, vượt quá khả năng chờ đợi của con người mặc dù có máy tính hỗ trợ. Đấy là chưa kể những bài toán khó thường là những bài toán phải xét trên những tập hợp vô hạn (chẳng hạn tập số nguyên). Trong những trường hợp này, người ta phải dẫn bài toán đã cho về những bài toán mà việc duyệt là hữu hạn và điều này không phải lúc nào cũng dễ dàng.

Hiện nay, bên cạnh những giải pháp truyền thống của suy luận toán học, phương pháp lập trình để giải các bài toán tổ hợp là không thể bỏ qua, nó giúp con người mở rộng được diện giải quyết những bài toán tổ hợp vốn rất đa dạng mà nhiều bài toán không thể giải được bằng các suy luận thông thường.

Mục dưới đây trình bày hai bài toán tồn tại nổi tiếng trên thế giới, sau một thời gian dài tìm kiếm lời giải, cuối cùng đã được giải quyết bằng sự hỗ trợ của máy tính.

2.3.6. Bài toán 36 sỹ quan và bài toán 4 màu

- **Bài toán 36 sỹ quan.** Bài toán này do Euler đề xuất dưới dạng một bài toán đố vui: "Có 6 trung đoàn, mỗi trung đoàn cử 6 sỹ quan mang 6 cấp bậc khác nhau: thiếu úy, trung úy, đại úy, thiếu tá, trung tá, đại tá, về dự duyệt binh. Hỏi có thể xếp 36 sỹ quan này vào một đội hình hình vuông 6 hàng, 6 cột sao cho nhìn theo bất cứ một hàng nào cũng như nhìn theo bất cứ một cột nào ta đều thấy 6 người thuộc 6 trung đoàn khác nhau và mang 6 cấp bậc khác nhau".

  Euler dùng 6 chữ cái lớn đầu tiên A, B, C, D, E, F để chỉ trung đoàn và 6 chữ cái nhỏ tương ứng a, b, c, d, e, f để chỉ cấp bậc. Như thế mỗi sỹ quan được biểu diễn bằng một cặp chữ cái, một lớn chỉ trung đoàn và một nhỏ chỉ cấp bậc, chẳng hạn các sỹ quan Ab, Ed, ... và bài toán xếp sỹ quan trở thành bài toán xếp chữ. Cũng vì cách biểu diễn này mà bài toán còn có tên gọi là bài toán *về hình vuông la tinh trực giao*.

  Cấp 6 trong bài toán vừa phát biểu không phải là ngẫu nhiên. Với cấp 2, ta dễ dàng duyệt tất cả các cách xếp và kết quả là không có một cách xếp nào thỏa mãn yêu cầu. Ba hình dưới đây cho các cách xếp tương ứng với các cấp 3, 4 và 5:

Aa	Bb	Cc
Bc	Ca	Ab
Cb	Ac	Ba

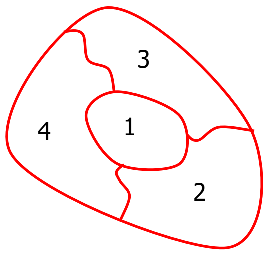
Ab	Dd	Ba	Cc
Bc	Ca	Ad	Db
Cd	Bb	Dc	Aa
Da	Ac	Cb	Bd

Aa	Bb	Cc	Dd	Ee
Cd	De	Ea	Ab	Bc
Eb	Ac	Bd	Ce	Da
Be	Ca	Db	Ec	Ad
Dc	Ed	Ae	Ba	Cb

  Euler đã mất rất nhiều công sức để xếp cho trường hợp cấp 6 nhưng không thành công vì thế ông đã đưa ra giả thuyết là cách xếp này không tồn tại, hơn thế nữa, cùng với trường hợp cấp 2, ông còn đưa ra giả thuyết tổng quát hơn: không tồn tại hình vuông la tinh trực giao cấp $n = 4k + 2$ với mọi $k = 0, 1, 2, \ldots$ Năm 1901, nhà toán học Pháp Tarri đã chứng minh không tồn tại hình vuông la tinh trực giao cấp 6 bằng cách duyệt tất cả mọi khả năng Tuy nhiên giả thuyết tổng quát của Euler còn kéo dài suốt hai thế kỷ, mãi đến năm 1960, ba nhà toán học Mỹ là Boce, Parker, Srikanda, mới chỉ ra được một lời giải với n = 10 và sau đó chỉ ra phương pháp xây dựng hình vuông la tinh trực giao cho mọi n = 4k + 2.

- **Bài toán 4 màu.** Một nguyên tắc được dùng cho việc tô màu bản đồ là hai nước cạnh nhau phải tô màu khác nhau. Vấn đề đặt ra là với số màu ít nhất là bao nhiêu để có thể tô dược mọi bản đồ?

  Sau khi bài toán được phát biểu, người ta đã chứng minh được rằng với 5 màu là đủ tô mọi bản đồ, ngoài ra người ta cũng chỉ ra được một ví dụ mà với 3 màu thì không đủ tô như hình vẽ dưới đây:



Từ đó, bài toán 4 màu được phát biểu như sau: "với 4 màu có đủ tô mọi bản đồ hay không?".

Bài toán này có thể phát biểu dưới dạng một bài toán tồn tại (giống như ví dụ vừa chỉ ra): "có tồn tại một bản đồ mà 4 màu không đủ tô hay không?". Nếu có một bản đồ như vậy thì 5 là số màu ít nhất, trái lại 4 sẽ là số màu ít nhất.

Câu trả lời dường như dễ tìm này đã cuốn hút rất nhiều người tham gia tìm kiếm, từ những người làm toán nghiệp dư đến những người làm toán chuyên nghiệp, nhiều chứng minh tưởng như đúng đã được công bố nhưng sau lại bị phát hiện sai, trong đó có một chứng minh sai có lẽ thuộc loại nổi tiếng nhất trong toán học, là của Alfred Kempe, một luật sư, nhà làm toán nghiệp dư thành London vào năm 1879. Giới toán học đã chấp nhận cách chứng minh này cho tới năm 1890, khi Percy Heawood phát hiện ra sai lầm. Tuy nhiên cách chứng minh của Kempe lại là cơ sở giúp cho Kenneth Appel và Wolfgang Haken, hai nhà toán học Mỹ, hoàn thành việc chứng minh của mình vào năm 1976, trong đó khẳng định rằng mọi bản đồ đều được tô với ít nhất 4 màu. Hai ông đã chứng tỏ rằng nếu có một phản ví dụ về một bản đồ mà 4 màu không đủ tô thì nó phải thuộc về khoảng gần 2000 loại khác nhau và bằng sự hỗ trợ của máy tính, họ đã chỉ ra rằng không có loại nào dẫn đến phản ví dụ cả. Trong chứng minh của mình, hai ông đã dùng hơn 1000 giờ máy. Cách chứng minh này hiện đã được công nhận dù đương thời đã gây ra nhiều tranh cãi.

TÓM LƯỢC CUỐI BÀI

Qua bài học, các bạn đã nắm được những nét chính của lý thuyết tổ hợp bao gồm đối tượng nghiên cứu, một số tên gọi, thuật ngữ, ứng dụng và một số vấn đề mà lý thuyết tổ hợp đề ra. Ngoài ra, các bạn đã được nghiên cứu nội dung hai bài toán của lý thuyết tổ hợp là bài toán đếm và bài toán tồn tại.

Các bạn cần ghi nhớ các vấn đề sau :

- Một số cấu hình cơ bản của lý thuyết tổ hợp
- Bài toán đếm và bài toán tồn tại
- Sử dụng những nguyên lý đếm cơ bản: nguyên lý cộng, nguyên lý nhân và nguyên lý bù trừ trong việc đếm các cấu hình cơ bản.
- Hiểu phương pháp và cách giải quyết bài toán đếm bằng kỹ thuật quy nạp.
- Sử dụng phương pháp phản chứng, nguyên lý Dirichlet và lý thuyết về hệ đại diện phân biệt trong việc giải quyết bài toán tồn tại.
- Giải bài toán đếm và bài toán tồn tại bằng lập trình

BÀI TẬP

Bài toán đếm

1. Có bao nhiêu chuỗi 10 bit:

a) có bit đầu tiên và bit cuối cùng là 1?

b) có 2 bit đầu bằng 0 hoặc 3 bit cuối bằng 1?

c) có đúng 4 bit 1?

d) có ít nhất 4 bit 1?

2. Có bao nhiêu xâu các chữ cái thường có độ dài 4:

a) các chữ là khác nhau?

b) chứa chữ x?

c) có đúng một chữ x?

d) có ít nhất hai chữ x?

3. Trong các số nguyên dương có đúng 4 chữ số, có bao nhiêu số:

a) chia hết cho 9?

b) lẻ?

c) có các chữ số khác nhau?

d) không chia hết cho 3?

e) chia hết cho cả 5 và 7?

f) chia hết cho 5 hoặc cho 7?

g) chia hết cho 5 nhưng không chia hết cho 7?

h) không chia hết cho 5 hoặc cho 7?

4. Có bao nhiêu số lẻ trong các số nguyên dương nhỏ hơn 600000 có đúng 6 chữ số khác nhau?

5. Có 10 cuốn sách khác nhau, trong đó có 5 cuốn về tin học, 3 cuốn về toán học và 2 cuốn về nghệ thuật. Hỏi có bao nhiêu cách chọn ra 2 cuốn thuộc các lĩnh vực khác nhau từ 10 cuốn sách đã cho?

6. Xếp 10 cuốn sách cho trong bài 5 lên một giá sách thành một hàng ngang. Hỏi có bao nhiêu cách xếp:

a) theo trật tự tùy ý?

b) các cuốn về tin học xếp phía bên trái và các cuốn về nghệ thuật xếp phía bên phải?

c) các cuốn thuộc cùng lĩnh vực xếp cạnh nhau?

d) các cuốn về toán học không xếp cạnh nhau?

7. Một phiếu trắc nghiệm có 10 câu hỏi. Mỗi câu có 4 phương án trả lời. Hỏi có bao nhiêu cách điền vào một phiếu:

a) nếu mọi câu hỏi đều được trả lời?

b) nếu có những câu hỏi bỏ trống?

8. Có 6 người A, B, C, D, E, F được đề cử vào ban lãnh đạo cuộc họp. Hỏi có bao nhiêu cách chọn ra 3 người: 1 chủ tịch, 1 phó chủ tịch và 1 thư ký:

a) tùy ý?

b) một trong hai người A, B là chủ tịch?

c) D và F là thành viên của ban lãnh đạo?

9. Một biển số xe gồm 2 chữ số đầu chỉ mã tỉnh, tiếp theo 3 chữ cái lớn chỉ loại xe và cuối cùng 4 chữ số chỉ số đăng ký. Hỏi có bao nhiêu biển số:

a) tùy ý?

b) số đăng ký là số tiến (nghĩa là chữ số sau lớn hơn chữ số trước)?

c) số đăng ký có mặt cả hai số 6 và 8?

10. Tên của mọi biến trong ngôn ngữ C là một xâu gồm không quá 8 ký tự là các chữ thường, chữ hoa, chữ số và dấu gạch dưới, trong đó ký tự đầu tiên không được là chữ số. Hỏi có bao nhiêu cách đặt tên biến trong C?

11. Có 7 nam và 4 nữ. Cần chọn ra 3 người. Hỏi số cách chọn:

a) có ít nhất 1 nam?

b) có ít nhất 1 nữ?

c) có cả nam và nữ?

12. Có 5 nhà toán học nam, 3 nhà toán học nữ và 4 nhà vật lý nam. Cần lập một đoàn công tác gồm 3 người. Hỏi có bao nhiêu cách:

a) có cả nam và nữ?

b) có cả nhà toán học và nhà vật lý?

c) có cả nam và nữ cũng như có cả nhà toán học và nhà vật lý?

13. Cho tam giác ABC. Trên cạnh AB lấy m điểm, trên cạnh AC lấy n điểm. Kẻ các đường thẳng nối B với các điểm đã chọn trên AC và nối C với các điểm đã chọn trên AB. Hỏi:

a) có bao nhiêu giao điểm của các đường thẳng nằm trong tam giác?

b) các đường thẳng chia tam giác thành bao nhiêu phần?

14. Cho một đa giác lồi n đỉnh.

a) hỏi có tất cả bao nhiêu tam giác có các đỉnh được chọn từ đỉnh của đa giác?

b) vẽ tất cả các đường chéo của đa giác và giả thiết không có 3 đường chéo nào đồng quy, hỏi số giao điểm nằm trong đa giác của các đường chéo?

15. Cho tập X gồm n phần tử. Có bao nhiêu bộ có thứ tự (A, B), trong đó A, B là các tập hợp thỏa mãn $A \subseteq B \subseteq X$.

16. Có bao nhiêu xâu ký tự nhận được từ xâu MISSISSIPPI bằng cách hoán vị các chữ cái trong xâu?

17. Có 9 thợ, người nào cũng làm được các công việc nề, mộc và điện. Hỏi có bao nhiêu cách chia thành 3 nhóm công việc gồm 4 người làm nề, 3 người làm mộc và 2 người làm điện?

18. Có 3 giỏ đựng các loại quả cam, táo, lê. Mỗi giỏ chỉ đựng một loại quả và giỏ nào cũng chưa ít nhất 8 quả. Các quả cùng loại được xem như nhau. Hỏi có bao nhiêu cách lấy ra 8 quả từ các giỏ đang xét:

a) các loại là tùy ý?

b) loại nào cũng ít nhất 2 quả?

19. Hỏi bất phương trình: $x_1 + x_2 + x_3 \leq 11$
có bao nhiêu nghiệm nguyên dương?

20. Một cuộc thi có 5 câu hỏi với số điểm tổng cộng là 20. Hỏi có bao nhiêu cách cho điểm mỗi câu (là số nguyên) sao cho câu nào cũng ít nhất được 3 điểm?

21. Có bao nhiêu cách gọi 3 người để trả lời hết 10 câu hỏi (mỗi câu hỏi gọi một người trả lời):
a) Tùy ý?
b) Người nào cũng trả lời ít nhất một câu?

22. Có bao nhiêu số nguyên dương không quá 100 vừa là số chẵn, vừa không phải là số chính phương, vừa không phải là số lập phương?

23. Một lớp gồm 50 học sinh làm bài kiểm tra gồm 3 câu hỏi. Biết rằng mỗi học sinh làm được ít nhất một câu và số học sinh làm được câu 1 là 40, câu 2 là 35, câu 3 là 30. Chứng minh rằng số học sinh làm được cả 3 câu không vượt quá 27.

24. Một luật chơi xổ số như sau: chọn 7 số từ 80 số nguyên dương đầu tiên. Nếu 7 số này nằm trong 11 số do hội đồng xổ số chọn thì trúng giải. Hỏi xác suất trúng giải là bao nhiêu?

25. Hỏi xác suất xảy ra sự kiện tổng các số ở mặt trên bằng 9 khi gieo 3 con súc sắc?

26. Một chuỗi bit được gọi là "thưa" nếu nó không chứa hai bit 1 liên tiếp.
a) Tìm một công thức truy hồi xác định số các chuỗi n bit thưa.
b) Có bao nhiêu chuỗi 10 bit thưa?

27. Một người lên cầu thang có thể bước một, hai, hoặc ba bậc một lần.
a) Tìm một công thức truy hồi đếm số cách mà người đó lên n bậc cầu thang.
b) Có bao nhiêu cách lên 8 bậc cầu thang theo kiểu đã nêu?

28. Một người cần thực hiện n công việc trong n ngày, mỗi ngày một công việc. Sau khi đã sắp xếp xong (ngày thứ i thực hiện công việc i, i = 1, 2, ..., n), người đó thấy rằng mỗi công việc đều có thể làm sớm hơn một ngày (trừ công việc đầu) và muộn hơn một ngày (trừ công việc cuối).
a) Tìm một công thức truy hồi xác định số lịch thực hiện n công việc.
b) Lập một chương trình thực hiện công thức này. Cho biết kết quả chạy chương trình với n = 30.

Bài toán tồn tại

29. Có 12 cầu thủ bóng rổ đeo áo với số từ 1 đến 12 đứng tập trung thành một vòng tròn. Chứng minh rằng luôn tìm được 3 người liên tiếp có tổng các số trên áo là lớn hơn hoặc bằng 20.

30. Có 51 ngôi nhà trong một phố. Mỗi ngôi nhà có địa chỉ nằm từ 1000 đến 1099. Chứng minh rằng ít nhất có hai nhà có địa chỉ là hai số nguyên liên tiếp.

31. Xét một nhóm người xếp thành một hàng ngang trong đó có n nữ. Chứng minh rằng:
a) Nếu số nam lớn hơn n + 1 thì mọi cách xếp đều tìm được hai nam cạnh nhau.
b) Nếu số nam bằng n + 1 thì mọi cách xếp hai nữ cạnh nhau đều tìm được hai nam cạnh nhau.

32. Tên họ viết tắt của một người gồm 3 chữ cái. Chứng minh rằng tìm được ít nhất 4 người ở California (dân số 25 triệu) trùng tên họ viết tắt và sinh nhật.

33. Một phiếu bầu cử ghi tên 8 ứng cử viên. Người đi bầu có thể gạch tên hoặc để nguyên bất cứ tên nào. Hỏi số người đi bầu ít nhất là bao nhiêu để luôn tìm được 100 phiếu bầu giống hệt nhau?

34. Một lớp đại học gồm 55 sinh viên đăng ký thầy hướng dẫn làm đề tài tốt nghiệp. Người ta nhận thấy rằng trong 10 người bất kỳ luôn tìm được 2 người cùng đăng ký một thầy. Chứng minh rằng trong lớp có ít nhất 7 người cùng đăng ký một thầy.

35. Một kỳ thủ tham gia thi đấu giành chức vô địch trong 75 ngày liền, mỗi ngày thi đấu ít nhất một ván, nhưng toàn bộ không quá 125 ván. Chứng minh rằng có những ngày liên tiếp anh ta đã đấu đúng 24 ván.

36. a) Cho 5 điểm phân biệt trên mặt phẳng có các tọa độ nguyên. Chứng minh rằng luôn tìm được hai điểm để trung điểm của đoạn thẳng nối chúng cũng có tọa độ nguyên.

b) Cho 9 điểm phân biệt trong không gian có các tọa độ nguyên. Chứng minh rằng luôn tìm được hai điểm để trung điểm của đoạn thẳng nối chúng cũng có tọa độ nguyên.

c) Mở rộng kết quả trên cho không gian n chiều.

37. Có 17 giáo sư đôi một trao đổi với nhau về 3 chủ đề, trong đó mỗi cặp chỉ trao đổi với nhau về 1 chủ đề. Chứng minh rằng luôn tìm được 3 người đôi một trao đổi với nhau về cùng 1 chủ đề.

38. Một bảng số có 100 ô gồm 10 hàng, 10 cột. Người ta điền mỗi ô một số từ 1 đến 100 (hai ô khác nhau điền hai số khác nhau). Chứng minh rằng luôn tìm được hai ô chung cạnh mà hiệu số của hai số ghi trong chúng lớn hơn 5.

39. Chứng minh rằng trong 101 người có chiều cao khác nhau đứng thành một hàng, có thể tìm được 11 người có chiều cao tăng dần hoặc giảm dần mà không thay đổi thứ tự của họ trong hàng.

CÂU HỎI THƯỜNG GẶP

1 Trong việc giải quyết bài toán đếm thông dụng, những nguyên lý nào thường hay được sử dụng?

2 Nhiệm vụ chính của bài toán đếm?

3 Nhiệm vụ chính của bài toán tồn tại?