Hồ Sĩ Đàm (Chủ biên)
Đỗ Đức Đông - Lê Minh Hoàng - Trần Đỗ Hùng - Nguyễn Thanh Hùng

TÀI LIỆU CHUYÊN TIN HỌC

Quyển 3

NHÀ XUẤT BẢN GIÁO DỤC VIỆT NAM

HỒ SĨ ĐÀM (Chủ biên)
ĐỖ ĐỨC ĐÔNG – LÊ MINH HOÀNG
TRẦN ĐỖ HÙNG – NGUYỄN THANH HÙNG

TÀI LIỆU CHUYÊN
TIN HỌC
QUYỂN 3

NHÀ XUẤT BẢN GIÁO DỤC VIỆT NAM

LỜI NÓI ĐẦU

Bộ Giáo dục và Đào tạo đã ban hành chương trình chuyên môn Tin học cho các lớp chuyên 10, 11, 12. Dựa theo các chuyên đề chuyên sâu trong chương trình nói trên, các tác giả biên soạn bộ sách chuyên cho môn Tin học bao gồm các vấn đề cơ bản nhất về cấu trúc dữ liệu, thuật toán và cài đặt chương trình.

Bộ sách gồm ba quyển, quyển 1, 2 và 3. Cấu trúc mỗi quyển bao gồm: phần lí thuyết giới thiệu các khái niệm cơ bản, cần thiết, thường dùng nhất; phần áp dụng trình bày các bài toán thường gặp, cách giải và cài đặt chương trình; cuối cùng là các bài tập (ngoại trừ chuyên đề 10 là chuyên đề tham khảo chuyên sâu nên không có bài tập). Các chuyên đề trong bộ sách được lựa chọn mang tính hệ thống từ cơ bản đến chuyên sâu.

Với trải nghiệm nhiều năm tham gia giảng dạy, bồi dưỡng học sinh chuyên tin học của các trường chuyên có truyền thống và uy tín, các tác giả đã lựa chọn, biên soạn các nội dung cơ bản, thiết yếu nhất mà mình đã được sử dụng để dạy học với mong muốn bộ sách phục vụ không chỉ cho giáo viên và học sinh chuyên tin học các trường Trung học phổ thông mà cả cho giáo viên, học sinh chuyên tin học Trung học cơ sở làm tài liệu tham khảo cho việc dạy và học của mình.

Với kinh nghiệm nhiều năm tham gia bồi dưỡng học sinh, sinh viên tham gia các kì thi học sinh giỏi Quốc gia, Quốc tế, Hội thi Tin học trẻ Toàn quốc, Olympiad Sinh viên Tin học Toàn quốc, Kì thi lập trình viên Quốc tế khu vực Đông Nam Á, các tác giả đã lựa chọn giới thiệu các bài tập, lời giải có định hướng phục vụ cho không chỉ học sinh mà cả sinh viên làm tài liệu tham khảo khi tham gia các kì thi trên.

Lần đầu tập sách được biên soạn, thời gian và trình độ có hạn nên chắc chắn còn nhiều thiếu sót, các tác giả mong nhận được ý kiến đóng góp của bạn đọc, các đồng nghiệp, sinh viên và học sinh để bộ sách được ngày càng hoàn thiện hơn.

Các tác giả

Chuyên đề 8

HÌNH HỌC TÍNH TOÁN

Hình học tính toán (computational geometry) là một nhánh của ngành khoa học máy tính, chuyên nghiên cứu về thuật toán giải quyết các bài toán liên quan tới các đối tượng hình học. Trong toán học và công nghệ hiện đại, hình học tính toán có ứng dụng khá rộng rãi trong các lĩnh vực về đồ hoạ máy tính, thiết kế, mô phỏng...

Do giới hạn nội dung của cuốn sách, chúng ta sẽ chỉ khảo sát một số bài toán và thuật toán căn bản. Để giúp cho người đọc nhanh chóng nắm bắt được ý tưởng và cài đặt thuật toán, đa số công thức hình học được thừa nhận, không chứng minh.

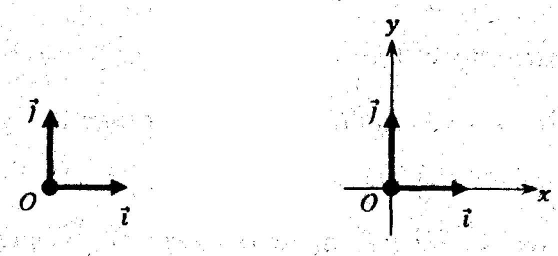
I. Một số khái niệm cơ bản

1. Hệ toạ độ Đề-các

Trong mặt phẳng, chọn một điểm O và hai vectơ đơn vị (vectơ có độ dài 1) $\vec{i}, \vec{j}$ vuông góc với nhau. Khi đó bộ ba $(O, \vec{i}, \vec{j})$ được gọi là hệ toạ độ Đề-các vuông góc (hay còn gọi là một mục tiêu Ơclit hai chiều, mục tiêu trực chuẩn).

Đơn vị độ dài là một khái niệm quy ước, có thể là cm, mm, inch... Ràng buộc về hai vectơ $\vec{i}, \vec{j}$ có thể viết dưới dạng biểu thức của tích vô hướng (tích chấm): $\vec{i}^2 = \vec{j}^2 = 1$ và $\vec{i}.\vec{j} = 0$.

Ta cũng kí hiệu mục tiêu đó là Oxy với Ox và Oy là hai tia gốc O có vectơ chỉ phương lần lượt là $\vec{i}$ và $\vec{j}$.



Hình 8.1. Mục tiêu Ơclit

Điểm O gọi là gốc toạ độ, đường thẳng Ox được gọi là trục hoành và Oy được gọi là trục tung.

Trong kĩ thuật vẽ hình, hai vectơ $\vec{i}$ và $\vec{j}$ thường được vẽ sao cho chiều quay từ $\vec{i}$ tới $\vec{j}$ ngược với chiều kim đồng hồ. Ta gọi chiều quay từ vectơ $\vec{i}$ tới vectơ $\vec{j}$ là *chiều thuận, chiều quay trái* hay *ngược chiều kim đồng hồ* (*counterclockwise-ccw*). Ngược lại, ta gọi chiều quay từ vectơ $\vec{j}$ tới vectơ $\vec{i}$ là *chiều nghịch, chiều quay phải* hay *chiều kim đồng hồ* (*clockwise-cw*).

2. Toạ độ

Xét mặt phẳng trực chuẩn $(O, \vec{i}, \vec{j})$, với một vectơ $\vec{v}$ bất kì của mặt phẳng, tồn tại duy nhất một cặp số thực $(x; y)$ sao cho $\vec{v} = x\vec{i} + y\vec{j}$. Cặp số $(x; y)$ khi đó được gọi là toạ độ của vectơ $\vec{v}$ đối với mục tiêu đã cho, kí hiệu $\vec{v} = (x; y)$.

Các kết quả sau có thể dễ dàng chứng minh bằng định nghĩa toạ độ:

- Vectơ $\vec{i}$ có toạ độ $(1; 0)$ và vectơ $\vec{j}$ có toạ độ $(0; 1)$.
- Nếu $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$ thì $\vec{u} = \vec{v}$ khi và chỉ khi $x_u = x_v$ và $y_u = y_v$.
- Nếu $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$ thì tổng vectơ $\vec{u} + \vec{v} = (x_u + x_v; y_u + y_v)$ và tích vô hướng $\vec{u}.\vec{v} = x_u x_v + y_u y_v$.
- Nếu $\vec{v} = (x;\ y)$ và $k \in \mathbb{R}$ thì $k\vec{v} = (kx; ky)$.

Trên mặt phẳng trực chuẩn $(O, \vec{i}, \vec{j})$ ta lấy một điểm M, khi đó toạ độ $(x; y)$ của vectơ $\overrightarrow{OM}$ được gọi là toạ độ của điểm M, kí hiệu $M = (x; y)$.

Ta có mối liên hệ giữa toạ độ của vectơ và toạ độ của điểm:

Nếu $M=(x_M; y_M)$ và $N = (x_N; y_N)$ thì $\overrightarrow{MN} = \overrightarrow{ON} - \overrightarrow{OM} = (x_N - x_M; y_N - y_M)$.

3. Đổi hệ toạ độ

Cho hai hệ toạ độ trực chuẩn trên mặt phẳng: $(O, \vec{i}, \vec{j})$ và $\left(O', \vec{i'}, \vec{j'}\right)$. Giả sử điểm M có toạ độ $(x; y)$ đối với hệ toạ độ $\left(O', \vec{i'}, \vec{j'}\right)$, bài toán đặt ra là xác định toạ độ điểm M đối với hệ toạ độ $(O, \vec{i}, \vec{j})$.

Giả sử trong hệ toạ độ $(O, \vec{i}, \vec{j})$ điểm $O' = (p; q)$, vectơ $\vec{i'} = (a; b)$, vectơ $\vec{j'} = (c; d)$. Khi đó theo định nghĩa:

$$
\begin{aligned}
\overrightarrow{OO'} &= p\vec{i} + q\vec{j} \\
\vec{i'} &= a\vec{i} + b\vec{j} \\
\vec{j'} &= c\vec{i} + d\vec{j}
\end{aligned}
\tag{8.1}
$$

Điểm M có toạ độ $(x; y)$ đối với hệ toạ độ $\left(O', \vec{i'}, \vec{j'}\right)$, tức là

$$\overrightarrow{O'M} = x\vec{i'} + y\vec{j'}$$

Suy ra

$$
\begin{aligned}
\overrightarrow{OM} &= \overrightarrow{OO'} + \overrightarrow{O'M} \\
&= p\vec{i} + q\vec{j} + x(a\vec{i} + b\vec{j}) + y(c\vec{i} + d\vec{j}) \\
&= \vec{i}(ax + cy + p) + \vec{j}(bx + dy + q)
\end{aligned}
\tag{8.2}
$$

Công thức (8.2) cho biết rằng điểm M có toạ độ $(ax + cy + p; bx + dy + q)$ đối với hệ toạ độ $(O, \vec{i}, \vec{j})$. Nếu ta gọi $(\bar{x}; \bar{y})$ là toạ độ điểm M đối với hệ toạ độ $(O, \vec{i}, \vec{j})$ thì công thức (8.2) có thể viết dưới dạng ma trận:

$$\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} p \\ q \end{pmatrix} \tag{8.3}$$

Chú ý: Công thức đổi toạ độ:

$$\begin{cases} \bar{x} = ax + cy + p \\ \bar{y} = bx + dy + q. \end{cases}$$

Ta có thể nhớ như sau cho dễ: các hệ số ở phương trình thứ nhất là hoành độ của ba vectơ $\vec{i'}, \vec{j'}$ và $\overrightarrow{OO'}$ trong khi các hệ số ở phương trình thứ hai là tung độ của ba vectơ đó.

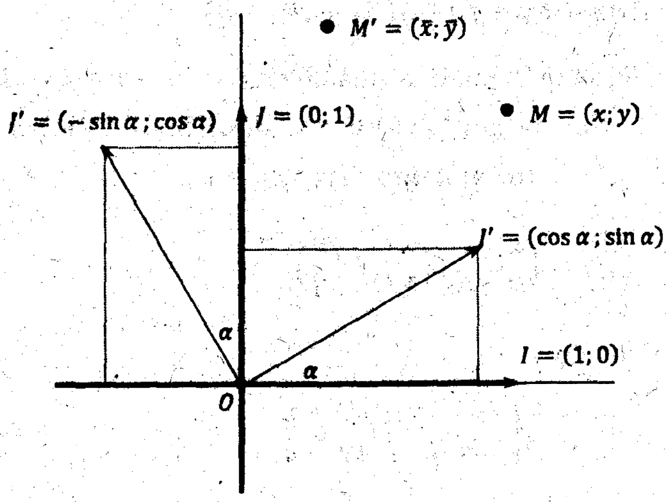
4. *Xây dựng công thức biến đổi toạ độ*

Trong hình học phẳng, một trong những kĩ thuật quan trọng là xây dựng công thức biến đổi toạ độ cho các phép đồng dạng (tịnh tiến, quay, đối xứng trục, vị tự...). Kĩ thuật chung có thể mô tả như sau:

- Trên hệ toạ độ trực chuẩn ban đầu $(O, \vec{i}, \vec{j})$, xác định điểm $I = (1; 0)$ và điểm $J = (0; 1)$.

- Với một phép đồng dạng f, ta thực hiện f trên ba điểm O, I, J để nhận được ba ảnh tương ứng của chúng theo thứ tự là O', I', J'. Từ đó xác định toạ độ điểm $O' = (p; q)$, vectơ $\vec{i'} = \overrightarrow{O'I'} = (a; b)$ và vectơ $\vec{j'} = \overrightarrow{O'J'} = (c; d)$.
- Nhận xét rằng nếu điểm M có toạ độ $(x; y)$ đối với hệ toạ độ trực chuẩn đã cho $(O, \vec{i}, \vec{j})$ thì sau phép biến đổi f, điểm M sẽ biến thành điểm M' cũng có toạ độ $(x; y)$ nhưng với hệ toạ độ trục chuẩn trực chuẩn $\left(O', \vec{i'}, \vec{j'}\right)$.
- Áp dụng công thức (8.2) hoặc (8.3) để xây dựng công thức toạ độ của điểm M' theo hệ toạ độ ban đầu.

a) Phép quay



Hình 8.2. Phép quay tâm O góc α

Ta xét phép quay quanh tâm O một góc α. Phép quay này giữ bất biến điểm O, tức là $O' = (0; 0)$, điểm $I = (1; 0)$ biến thành $I' = (\cos\alpha; \sin\alpha)$, điểm $J = (0; 1)$ biến thành $J' = (-\sin\alpha; \cos\alpha)$. Như vậy:

$$O' = (0; 0);$$
$$\overrightarrow{O'I'} = (\cos\alpha; \sin\alpha);$$
$$\overrightarrow{O'J'} = (-\sin\alpha; \cos\alpha).$$

Áp dụng công thức (8.2) hoặc (8.3), phép quay tâm O góc α sẽ biến điểm $(x; y)$ thành điểm $(\bar{x}; \bar{y})$, trong đó:

$$\begin{pmatrix}\bar{x}\\ \bar{y}\end{pmatrix} = \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix}$$

tức là:

$$\begin{cases}\bar{x} = x\cos\alpha - y\sin\alpha\\ \bar{y} = x\sin\alpha + y\cos\alpha.\end{cases} \tag{8.4}$$

Trường hợp đặc biệt:

Nếu góc $\alpha = 90^o$, ta có $\cos\alpha = 0$ và $\sin\alpha = 1$. Công thức (8.4) trở thành

$$\begin{cases}\bar{x} = -y\\ \bar{y} = x.\end{cases}$$

Nếu góc $\alpha = 180^o$, đây là phép đối xứng tâm O, $\cos\alpha = -1$ và $\sin\alpha = 0$. Công thức (8.4) trở thành:

$$\begin{cases}\bar{x} = -x\\ \bar{y} = -y.\end{cases}$$

b) Phép tịnh tiến

Ta xét phép tịnh tiến theo vectơ $(a; b)$. Phép tịnh tiến này biến điểm $O = (0; 0)$ thành $O' = (a; b)$. Điểm $I = (1; 0)$ biến thành $I'(a + 1; b)$ và điểm $J = (0; 1)$ biến thành $J' = (a; b + 1)$. Như vậy:

$$O' = (a; b);$$
$$\overrightarrow{O'I'} = (1; 0);$$
$$\overrightarrow{O'J'} = (0; 1).$$

Áp dụng công thức (8.2) hoặc (8.3), phép tịnh tiến theo vectơ $(a; b)$ sẽ biến điểm $(x; y)$ thành điểm $(\bar{x}; \bar{y})$, trong đó:

$$\begin{pmatrix}\bar{x}\\ \bar{y}\end{pmatrix} = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}a\\ b\end{pmatrix}$$

tức là:

$$\begin{cases}\bar{x} = x + a\\ \bar{y} = y + b.\end{cases} \tag{8.5}$$

c) Phép quay quanh một điểm cho trước

Công thức phép quay góc α quanh một điểm A có thể xây dựng bằng phương pháp tương tự trên, tuy nhiên ta có thể làm theo cách sau:

Thực hiện phép tịnh tiến theo vectơ $\overrightarrow{AO}$, sau đó thực hiện phép quay tâm O góc α, rồi thực hiện tiếp phép tịnh tiến theo vectơ $\overrightarrow{OA}$.

Công thức cụ thể như thế nào xin dành cho bạn đọc.

d) Phép vị tự

Để thực hiện phép vị tự tâm $A = (x_A; y_A)$ tỉ số k, ta có thể dùng phương pháp như sau:

- Tịnh tiến theo vectơ $\overrightarrow{AO}$: Điểm $(x; y)$ sẽ biến thành điểm $(x'; y')$ thoả mãn:

$$\begin{cases} x' = x - x_A \\ y' = y - y_A. \end{cases}$$

- Thực hiện phép vị tự tâm O tỉ số k. Phép vị tự này giữ bất biến điểm O, điểm $I = (1; 0)$ biến thành $I' = (k; 0)$, điểm $J = (0; 1)$ biến thành $J' = (0; k)$. Như vậy:

$$O' = (0; 0);$$
$$\overrightarrow{O'I'} = (k; 0);$$
$$\overrightarrow{O'J'} = (0; k).$$

Áp dụng công thức (8.2) hoặc (8.3), sau phép vị tự tâm O tỉ số k, điểm $(x'; y')$ sẽ biến thành điểm $(x''; y'')$ thoả mãn:

$$\begin{cases} x'' = kx' \\ y'' = ky'. \end{cases}$$

- Cuối cùng, ta thực hiện phép tịnh tiến theo vectơ $\overrightarrow{OA}$:

$$\begin{cases} \bar{x} = x'' + x_A \\ \bar{y} = y'' + y_A. \end{cases}$$

Tổng hợp lại ta được:

$$\begin{cases} \bar{x} = k(x - x_A) + x_A \\ \bar{y} = k(y - y_A) + y_A. \end{cases} \quad (8.6)$$

5. *Một số khái niệm khác*

a) Tích chấm

Tích chấm (*dot product*) hay tích vô hướng của hai vectơ $\vec{u}$ và $\vec{v}$, kí hiệu $\vec{u}.\vec{v}$ là một số thực được tính bằng tích độ dài hai vectơ $\vec{u}$ và $\vec{v}$ nhân với *cosin* của góc xen giữa hai vectơ đó. Góc xen giữa hai vectơ này là góc không định hướng, có số đo từ 0 tới π.

Hàm số *cosin* là hàm nghịch biến trong khoảng $[0;\pi]$, nó đạt giá trị lớn nhất bằng 1 khi góc giữa hai vectơ bằng 0 (hai vectơ cùng chiều), đạt giá trị nhỏ nhất bằng -1 khi góc giữa hai vectơ bằng π (hai vectơ ngược chiều) và đạt giá trị 0 khi hai vectơ vuông góc (trực giao).

Biểu thức của tích chấm giữa hai vectơ $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$ có thể diễn giải như sau:

Ta có:

$$\vec{u} = x_u\vec{i} + y_u\vec{j}$$

$$\vec{v} = x_v\vec{i} + y_v\vec{j}.$$

Vậy:

$$\begin{aligned} \vec{u}.\vec{v} &= (x_u\vec{i} + y_u\vec{j}).(x_v\vec{i} + y_v\vec{j}) \\ &= x_u x_v \underbrace{\vec{i}^2}_{=1} + y_u y_v \underbrace{\vec{j}^2}_{=1} + (x_u y_v + y_u x_v)\underbrace{\vec{i}.\vec{j}}_{=0} \\ &= x_u x_v + y_u y_v. \end{aligned} \quad (8.7)$$

Cũng từ định nghĩa tích chấm, ta suy ra công thức tính cosin của góc α hợp bởi hai vectơ $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$:

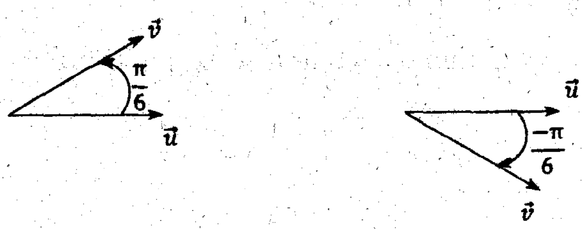
$$\cos\alpha = \frac{\vec{u}.\vec{v}}{|\vec{u}|.|\vec{v}|} = \frac{x_u x_v + y_u y_v}{\sqrt{(x_u^2 + y_u^2)(x_v^2 + y_v^2)}} \quad (8.8)$$

Tích vô hướng có thể coi như một độ đo về mức độ cùng chiều giữa hai vectơ. Quan trọng hơn, nó là cơ sở để xây dựng các khái niệm về khoảng cách và góc. Ví dụ nếu $A = (x_A; y_A)$ và $B = (x_B; y_B)$ thì độ dài đoạn thẳng AB được tính bằng:

$$\sqrt{\overrightarrow{AB}^2} = \sqrt{(x_B - x_A)^2(y_B - y_A)^2}$$

b) Tích chéo

Tích chéo (*cross product*) của hai vectơ $\vec{u}$ và $\vec{v}$, kí hiệu $\vec{u} \times \vec{v}$ là một số thực được tính bằng tích độ dài hai vectơ $\vec{u}$ và $\vec{v}$ nhân với *sin* của góc xen giữa hai vectơ đó. Góc xen giữa hai vectơ này là góc định hướng, có số đo từ $-\pi$ tới π, số đo mang dấu dương nếu chiều quay từ $\vec{u}$ tới $\vec{v}$ là chiều thuận (ngược chiều kim đồng hồ) và mang dấu âm nếu chiều quay từ $\vec{u}$ tới $\vec{v}$ là chiều nghịch (theo chiều kim đồng hồ).



Hình 8.3. Góc có hướng

Tích chéo là một khái niệm suy ra từ khái niệm tích có hướng trong không gian vectơ Ơclit nhiều chiều. Bằng các công cụ đại số tuyến tính, người ta đã chứng minh được công thức của tích chéo giữa hai vectơ $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$:

$$\vec{u} \times \vec{v} = x_u y_v - x_v y_u = \begin{vmatrix} x_u & x_v \\ y_u & y_v \end{vmatrix} \tag{8.9}$$

tức là giá trị của tích chéo bằng định thức của ma trận $\begin{pmatrix} x_u & x_v \\ y_u & y_v \end{pmatrix}$.

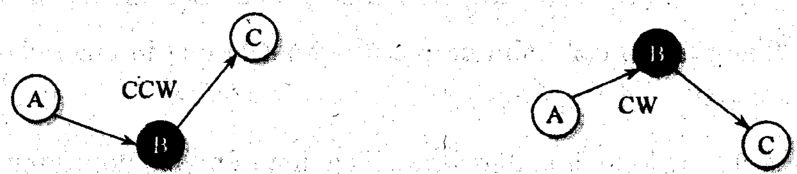
Ta cũng suy ra công thức tính sin của góc định hướng α giữa hai vectơ $\vec{u} = (x_u; y_u)$ và $\vec{v} = (x_v; y_v)$:

$$\sin\alpha = \frac{\vec{u} \times \vec{v}}{|\vec{u}|.|\vec{v}|} = \frac{x_u y_v - x_v y_u}{\sqrt{(x_u^2 + y_u^2)(x_v^2 + y_v^2)}}.$$

Về mặt hình học, giá trị tuyệt đối của **tích chéo** $\vec{u} \times \vec{v}$ là diện tích hình bình hành $OABC$, trong đó O là gốc toạ độ, $\overrightarrow{OA} = \vec{u}$, $\overrightarrow{OC} = \vec{v}$ và $\overrightarrow{OB} = \vec{u} + \vec{v}$.

Tích chéo có một ứng dụng quan trọng trong việc khảo sát chiều: Giả sử ta đi từ điểm A sang điểm B theo đường thẳng và đi tiếp sang điểm C theo đường thẳng, khi đó:

- Tích chéo $\overrightarrow{AB} \times \overrightarrow{BC}$ sẽ là số dương nếu chỗ rẽ tại B là "rẽ trái" (hay nói đúng hơn là bẻ góc ngược chiều kim đồng hồ);
- Tích chéo $\overrightarrow{AB} \times \overrightarrow{BC}$ là số âm nếu chỗ rẽ tại B là "rẽ phải";
- Tích chéo $\overrightarrow{AB} \times \overrightarrow{BC} = 0$ có nghĩa là ba điểm A, B, C thẳng hàng.



Hình 8.4. Rẽ trái và rẽ phải

Ta lấy ví dụ về một ứng dụng của tích chấm và tích chéo: Trên mặt phẳng cho ba điểm $A = (x_A; y_A)$, $B = (x_B; y_B)$ và $C = (x_C; y_C)$, hãy cho biết điểm C có nằm trên đoạn thẳng AB hay không.

Điều kiện cần và đủ để C nằm trên đoạn thẳng AB là: A, B, C thẳng hàng và hai vectơ $\overrightarrow{AC}$, $\overrightarrow{BC}$ không cùng hướng hoặc một trong số chúng là vectơ $\vec{0}$. Điều kiện này có thể viết bởi

$$\overrightarrow{AC} \times \overrightarrow{BC} = 0 \text{ và } \overrightarrow{AC}.\overrightarrow{BC} \leq 0.$$

c) Đường thẳng

Có nhiều cách biểu diễn một đường thẳng và do đó có nhiều cách biểu diễn đường thẳng trong máy tính. Sau đây là một số cách biểu diễn thường được sử dụng khi giải quyết các bài toán tin học:

- Dạng: $y = ax + b$. Mỗi đường thẳng được đặc trưng bởi một cặp hai hệ số a và b, tuy nhiên dạng biểu diễn này không thể hiện được các đường thẳng song song với trục Oy.

- Dạng tổng quát: $ax + by = c$. Mỗi đường thẳng có thể được biểu diễn bởi bộ ba hệ số (a, b, c). Đây vẫn là cách lưu trữ thông dụng nhất do tính tổng quát và trực quan của nó (gần gũi với đồ thị hàm số). Vectơ pháp tuyến $\vec{n}(a; b)$ có thể được dùng để viết một phương trình đường thẳng tương đương ngắn gọn hơn: $\vec{n}.\overrightarrow{OP} = (a; b).(x; y) = ax + by = c$ với mọi $P(x;\ y)$ nằm trên đường thẳng. Dạng tổng quát này còn được kí hiệu là $(\vec{n}, c)$.
- Dạng tham số: $(P, \vec{d})$, trong đó P là một điểm trên đường thẳng còn $\vec{d}$ là vectơ chỉ phương của đường thẳng với toạ độ là: $(-b;\ a)$. Như vậy, một đường thẳng được xác định bởi bộ bốn số x_0, y_0, a và b trong đó $(x_0;\ y_0)$ là toạ độ điểm P còn $(-b;\ a)$ là toạ độ của vectơ $\vec{d}$. Biểu diễn này có ý nghĩa: đường thẳng là tập các điểm có toạ độ chính là toạ độ của vectơ $\overrightarrow{OP} + s\vec{d}$ với mọi $s \in \mathbb{R}$.

Chuyển đổi giữa các biểu diễn được thực hiện dựa trên biến đổi tương đương đại số, cùng với công thức chuyển đổi giữa vectơ chỉ phương $\vec{d}(-b; a)$ và vectơ pháp tuyến $\vec{n}(a; b)$.

d) Góc

Góc thể hiện sự khác nhau về hướng của các vectơ và được biểu diễn bởi một số thực. Góc có nhiều đơn vị khác nhau: °, rad... Trong lập trình, để giảm sai số và tăng tốc độ tính toán, người ta tránh dùng *arcsin*, *arcos* mà thay vào đó bằng các đại lượng có cùng tính thứ tự như góc: *sin*/*cos*/*tang* của một góc có thể được tính nhanh hơn bằng toán tử +, –, *, / có sẵn trong các ngôn ngữ lập trình.

e) Đa giác

Đa giác là một đường gấp khúc khép kín. Trong lập trình, một đa giác được lưu bởi một dãy các đỉnh liên tiếp nhau $A_1, A_2, \ldots, A_n$.

Diện tích đại số của một đa giác không tự cắt có thể được xác định bởi công thức:

$$S = \frac{(x_1 - x_2)(y_1 + y_2) + (x_2 - x_3)(y_2 + y_3) + \cdots + (x_n - x_1)(y_n + y_1)}{2}.$$

$|S|$ chính là diện tích của đa giác.

Dãy đỉnh của đa giác có thể được lưu cùng hoặc ngược chiều kim đồng hồ. Ta có thể biết thứ tự lưu đỉnh nhờ dấu của S: $S > 0$ có nghĩa là đỉnh của đa giác được liệt kê ngược chiều kim đồng hồ (và ngược lại).

Để hình dung công thức trên, ta có thể dựng các đường thẳng đứng xuống trục hoành. Mỗi hạng tử sẽ là diện tích đại số của một hình thang thành phần. Diện tích các hình thang phía trên sẽ ngược dấu với các hình thang bên dưới, từ đó cho ta diện tích đại số của đa giác khi lấy tổng của chúng.

Nếu vẫn phân vân về lời giải thiếu tính xây dựng trên, ta có thể xây dựng lại công thức tính diện tích bằng cách khác: tính tổng diện tích các tam giác $A_1A_2A_3$, $A_1A_3A_4$, …, $A_1A_{n-1}A_n$. Diện tích mỗi tam giác được tính bằng tích có hướng của hai vectơ $\overrightarrow{A_1A_{i-1}}$ và $\overrightarrow{A_1A_i}$, $i = \overline{3;n}$, do đó đây là diện tích có dấu. Tổng của các diện tích này sau khi rút gọn sẽ đưa về biểu thức trên đây và do đó cũng lí giải tại sao S lại là diện tích đại số.

Dưới đây là chương trình tính diện tích đa giác với khuôn dạng Input/Output như sau:

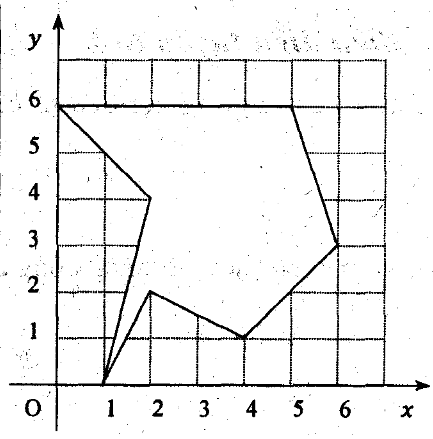
Input

- Dòng đầu tiên chứa số đỉnh n của đa giác ($1 \leq n \leq 10^6$);
- n dòng tiếp theo, dòng thứ i chứa hai số thực x_i, y_i là toạ độ đỉnh i của đa giác.

Output

Chiều của đa giác (CCW/CW) và diện tích đa giác

Sample Input	Sample Output
7 1 0 2 2 4 1 6 3 5 6 0 6 2 4	Polygon direction: CCW Area = 18.5000



Hình 8.5

POLYGONAREA.PAS ✓ Tính diện tích đa giác

```
{$MODE OBJFPC}
program PolygonArea;
uses Math;
const maxN = 1000000;
var
 x, y: array[0..maxN + 1] of Float;
 i, n: Integer; s: Float;
begin
 ReadLn(n);
 for i := 1 to n do ReadLn(x[i], y[i]);
 x[0] := x[n]; x[n + 1] := x[1];
 s := 0;
 for i := 1 to n do
 s := s + y[i] * (x[i - 1] - x[i + 1]);
 if s > 0 then WriteLn('Polygon direction: CCW')
 else WriteLn('Polygon direction: CW');
 WriteLn('Area = ', Abs(s)/2:0:4);
end.
```

f) Đường tròn

Đường tròn $(O; R)$ là tập hợp các điểm cách đều tâm $O(x; y)$ một khoảng cách R. Đường tròn được hoàn toàn xác định bởi bộ ba số (x, y, R) $(R > 0)$.

II. Một số bài toán cơ bản

1. *Biểu diễn tuyến tính*

Bài toán đầu tiên ta xét đến là cho ba vectơ $\vec{a}$, $\vec{b}$ và $\vec{c}$, hãy tìm hai số thực p, q để:

$$\vec{c} = p\vec{a} + q\vec{b}.$$

Hai số p, q có thể tính bằng công thức:

$$p = \frac{\vec{c} \times \vec{b}}{\vec{a} \times \vec{b}} = \frac{Dx}{D};$$

$$q = \frac{\vec{a} \times \vec{c}}{\vec{a} \times \vec{b}} = \frac{Dy}{D}.$$

- Nếu $D = \vec{a} \times \vec{b} \neq 0$ thì có duy nhất một cách biểu diễn tuyến tính vectơ $\vec{c}$ qua hai vectơ $\vec{a}$ và $\vec{b}$ (nghiệm $(p;q)$ là duy nhất).
- Nếu $D = 0$, hai vectơ $\vec{a}$ và $\vec{b}$ song song với nhau, khi đó:
  - $(p;q) = (Nan;Nan)$ nếu $\vec{c}$ song song với cả $\vec{a}$ và $\vec{b}$;
  - $(p;q) = (Inf;Inf)$ nếu $\vec{c}$ không song song với $\vec{a}$ và $\vec{b}$.

Ta viết hàm

```
function SolveSLE(const a, b, c: TVector): TVector;
```

nhận vào ba vectơ $\vec{a}, \vec{b}, \vec{c}$ và trả về một vectơ có toạ độ $(x;y)$ tương ứng với hệ số $(p;,q)$ cần tìm:

```
function SolveSLE(const a, b, c: TVector): TVector;
var
 D: Float;
begin
 D := a >< b;
 Result := Vector((c >< b)/D, (a >< c)/D);
end;
```

2. *Tìm giao điểm của hai đường thẳng*

Trên mặt phẳng với hệ toạ độ Đề-các vuông góc cho hai đường thẳng với phương trình tổng quát:

$$A_1x + B_1y + C_1 = 0;$$
$$A_2x + B_2y + C_2 = 0.$$

Bài toán đặt ra là xác định giao điểm của hai đường thẳng đã cho.

Đặt $\vec{u} = (A_1;A_2)$, $\vec{v} = (B_1;B_2)$ và $\vec{w} = (-C_1;-C_2)$, bài toán trở thành bài toán biểu diễn vectơ $\vec{w}$ qua tổ hợp tuyến tính của hai vectơ $\vec{u}$ và $\vec{w}$

$$\vec{w} = x.\vec{u} + y.\vec{v}.$$

Việc còn lại chỉ là biện luận cho giá trị giao điểm tìm được.

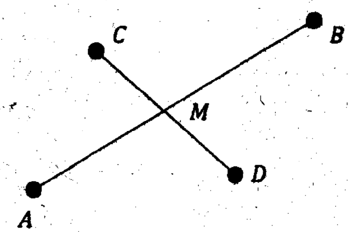
LINEINTERSECT.PAS ✓ Tìm giao điểm của hai đường thẳng

```
{$MODE OBJFPC}
program LineIntersection;
uses math;
type
```

```
TPoint = record
          x, y: Float;
        end;
TVector = TPoint;
var
 a1, b1, c1, a2, b2, c2: Float;
 p: TPoint;
function Vector(x, y: Float): Tvector;
begin
 Result.x := x; Result.y := y;
end;
//Tích chéo của hai vectơ
operator ><(const u, v: TVector): Float;
begin
 Result := u.x * v.y - u.y * v.x;
end;
//Biểu diễn tuyến tính
function SolveSLE(const a, b, c: TVector): TVector;
var
 D: Float;
begin
 D := a >< b;
 Result := Vector((c >< b)/D, (a >< c)/D);
end;
//Chương trình chính
BEGIN
SetExceptionMask([Low(TFPUExceptionMask)..High(TFPUExceptionMask)]);
ReadLn(a1, b1, c1, a2, b2, c2);
//Tìm giao điểm của hai đường thẳng
p := SolveSLE(Vector(a1,a2),Vector(b1,b2),Vector(c1,c2));
 if IsNan(p.x) then //Hai đường thẳng trùng nhau
 WriteLn('Two lines are coincident')
 else if IsInfinite(p.x) then //Hai đường thẳng song song
 WriteLn('Two lines are parallel')
 else //Hai đường thẳng có giao điểm duy nhất
WriteLn('Intersection point(',p.x:0:4,',', p.y:0:4,')');
END.
```

3. Tìm giao điểm của hai đoạn thẳng

Bài toán tiếp theo ta xét là cho bốn điểm A, B, C, D trên mặt phẳng, hãy cho biết hai đoạn thẳng AB và CD có giao điểm duy nhất không, nếu có cho biết toạ độ giao điểm.



Hình 8.6

Để tìm giao điểm của hai đoạn thẳng AB và CD ta có thể:

- Viết phương trình tổng quát của hai đường thẳng AB và CD;
- Tìm giao điểm M của hai đường thẳng;
- Kiểm tra M có nằm trên đoạn thẳng AB và trên đoạn thẳng CD hay không.

Phương trình của đường thẳng đi qua hai điểm phân biệt $A = (x_A; y_A)$ và $B = (x_B; y_B)$ là

$$(x - x_A) * (y_B - y_A) = (y - y_A) * (x_B - x_A).$$

Ta có thể biến đổi phương trình trên về phương trình tổng quát và sử dụng hàm *SolveSLE* để tìm giao điểm M của hai đường thẳng AB và CD, sau đó kiểm tra M có nằm trên hai đoạn thẳng AB và CD hay không.

Cách làm này kéo theo nhiều phép tính không hiệu quả, ta sẽ sử dụng công thức khác.

Nếu M là giao điểm duy nhất của hai đoạn thẳng AB và CD thì sẽ tồn tại duy nhất một cặp số thực $p, q \in [0; 1]$ để

$$\begin{cases} \overrightarrow{AM} = p\,\overrightarrow{AB} \\ \overrightarrow{CM} = q\,\overrightarrow{CD}. \end{cases}$$

Trừ tương ứng hai vế, ta có

$$\overrightarrow{AC} = p\,\overrightarrow{AB} + q\,\overrightarrow{DC}.$$

Như vậy, ta chỉ cần tìm biểu diễn tuyến tính của $\overrightarrow{AC}$ qua $\overrightarrow{AB}$ và $\overrightarrow{DC}$, sau khi có cặp số p, q, ta kiểm tra điều kiện $p, q \in [0; 1]$ và tính toạ độ giao điểm M theo công thức:

$$\overrightarrow{OM} = \overrightarrow{OA} + p\overrightarrow{AB}.$$

Thuật toán như sau:

```
Input → A, B, C, D: TPoint;
r := SolveSLE(B - A, C - D, C - A);
if not InRange(r.x, 0, 1) or not InRange(r.y, 0, 1) then
//Kiểm tra nghiệm ∈[0,1]
Output ← Không có giao điểm duy nhất
else
begin
p := A + (B - A) * r.x;
Output ← Giao điểm duy nhất (p.x, p.y)
end;
```

4. Tìm giao điểm giữa một đoạn thẳng và một tia

Để tìm giao điểm duy nhất giữa đoạn thẳng AB và tia CD, ta có thể sử dụng phương pháp tương tự như trên: Nếu M là giao điểm duy nhất của đoạn thẳng AB và tia CD thì sẽ tồn tại duy nhất một cặp số thực p, q, trong đó $p \in [0; 1]$, $q \in [0; +\infty)$ sao cho

$$\begin{cases} \overrightarrow{AM} = p\,\overrightarrow{AB} \\ \overrightarrow{CM} = q\,\overrightarrow{CD}. \end{cases}$$

Áp dụng thuật toán như trước, chỉ có điều sau khi tìm được cặp số p, q, ta không kiểm tra $p, q \in [0; 1]$ mà kiểm tra $p \in [0; 1]$ và $q > 0$.

Cách tìm giao điểm của đường thẳng AB với đường thẳng CD, tia AB với tia CD, đường thẳng AB với đoạn thẳng CD... có thể thực hiện theo cách tương tự, chỉ cần sửa đổi phạm vi của tham số p, q.

5. Đo góc giữa hai vectơ

Bài toán: Cho hai vectơ khác $\vec{0}$: $\vec{u}$ và $\vec{v}$. Cần tìm số đo góc định hướng từ $\vec{u}$ tới $\vec{v}$. Số đo góc định hướng nằm trong phạm vi $(-\pi; \pi]$.

Thư viện *math* cung cấp hàm *ArcTan2(y, x)* trả về góc định hướng tạo bởi vectơ $(1; 0)$ với vectơ $(x; y)$. Ta có thể sử dụng hàm này để tính giá trị góc định hướng từ $\vec{u}$ tới $\vec{v}$ bằng cách viết:

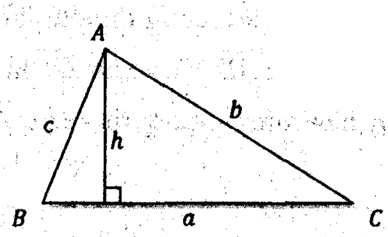
$$\text{ArcTan2}(\vec{u} \times \vec{v}, \vec{u}.\vec{v}).$$

Thủ tục như sau:

```
function Rad(const u, v: TVecto): Float;
begin
 Result := ArcTan2(u >< v, u * v);
end;
```

6. *Tính diện tích*

a) Diện tích tam giác



Hình 8.7

Tuỳ thông tin được cho mà ta sử dụng công thức giải tích thích hợp để tính diện tích tam giác:

- Nếu biết độ dài một cạnh của tam giác là a và chiều cao tương ứng với cạnh đó là h thì diện tích tam giác có thể tính bởi công thức:

$$S = \frac{a + h}{2}.$$

- Nếu biết độ dài hai cạnh tam giác là b và c, đồng thời góc xen giữa hai cạnh đó là A thì diện tích tam giác có thể tính bởi công thức:

$$S = \frac{bc \sin A}{2}.$$

- Nếu ta biết độ dài ba cạnh của tam giác là a, b, c thì diện tích tam giác có thể tính bởi công thức Hê-rông:

$$S = \sqrt{p(p - a)(p - b)(p - c)},$$

trong đó $p = \frac{a+b+c}{2}$ (nửa chu vi).

- Nếu biết hai vectơ $\overrightarrow{AB}$ và $\overrightarrow{AC}$ thì diện tích tam giác có thể tính bằng công thức tích chéo:

$$S = \left|\frac{\overrightarrow{AB} \times \overrightarrow{AC}}{2}\right| \tag{8.9}$$

Công thức này có thể viết lại khi ta biết toạ độ ba điểm $A = (x_A; y_A)$, $B = (x_B; y_B)$ và $C = (x_C; y_C)$:

$$S = \left|\frac{(x_B - x_A)(y_C - y_A) - (x_C - x_A)(y_B - y_A)}{2}\right| \tag{8.10}$$

Chú ý rằng nếu bỏ dấu giá trị tuyệt đối, công thức tính diện tích ở trên sẽ cho giá trị dương nếu hướng quay từ vectơ $\overrightarrow{AB}$ tới vectơ $\overrightarrow{AC}$ là hướng thuận. Ngược lại, công thức sẽ cho giá trị âm nếu hướng quay từ vectơ $\overrightarrow{AB}$ tới vectơ $\overrightarrow{AC}$ là hướng nghịch.

b) Diện tích đa giác

Trên mặt phẳng với hệ toạ độ Đề-các vuông góc, cho đa giác $P = P_1P_2 \dots P_n$, trong đó điểm P_i có toạ độ $(x_i; y_i)$. Bổ sung thêm điểm $P_0 \equiv P_n$ và điểm $P_{n+1} \equiv P_1$, khi đó diện tích đa giác được tính theo công thức (8.11):

$$\begin{aligned} S &= \frac{1}{2}\left|\sum_{i=1}^{n} \overrightarrow{OP_i} \times \overrightarrow{OP_{i+1}}\right| \\ &= \frac{1}{2}\left|\sum_{i=1}^{n} (x_{i-1} - x_{i+1})y_i\right| \end{aligned} \tag{8.11}$$

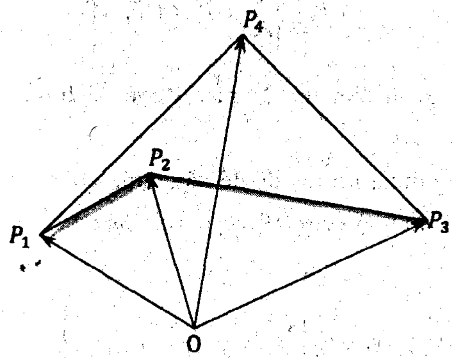
Chứng minh

Giả sử rằng các đỉnh của đa giác được đánh số theo hướng thuận (ngược chiều kim đồng hồ)*, ta nối các điểm $P_1, P_2, \dots, P_n$ với gốc toạ độ O, được n tam giác:

---

* Cụ thể là nếu ta đi dọc theo các cạnh của đa giác theo thứ tự $P_1, P_2, \dots, P_n, P_1$ thì phía tay trái là miền đa giác.

$$P_1OP_2, P_2OP_3, \dots, P_{n-1}OP_n, P_nOP_1.$$



Hình 8.8

Trong hình 8.8, ta xét một tứ giác với bốn đỉnh P_1, P_2, P_3, P_4. Diện tích của tứ giác này có thể tính bằng:

$$S = -S_\Delta(P_1OP_2) - S_\Delta(P_2OP_3) + S_\Delta(P_3OP_4) + S_\Delta(P_4OP_1)$$

tức là diện tích đa giác có thể tính qua các diện tích các tam giác $S_\Delta(P_iOP_{i+1})$. Khi một hạng tử $S_\Delta(P_iOP_{i+1})$ xuất hiện trong tổng, nó sẽ được mang dấu "+" nếu chiều quay từ vectơ $\overrightarrow{OP_i}$ tới vectơ $\overrightarrow{OP_{i+1}}$ là hướng thuận, ngược lại, hạng tử này sẽ mang dấu "-". Thật may mắn, giá trị (có dấu) của các hạng tử $S_\Delta(P_iOP_{i+1})$ có thể tính bằng công thức tích chéo (8.9) nhưng bỏ dấu giá trị tuyệt đối. Tức là diện tích (có dấu) của tam giác P_iOP_{i+1} sẽ được tính bằng:

$$S_\Delta(P_iOP_{i+1}) = \frac{1}{2}\overrightarrow{OP_i} \times \overrightarrow{OP_{i+1}}$$

và như vậy ta có công thức tính diện tích đa giác P:

$$\begin{aligned} S &= \frac{1}{2}\left(\overrightarrow{OP_1} \times \overrightarrow{OP_2} + \overrightarrow{OP_2} \times \overrightarrow{OP_3} + \dots + \overrightarrow{OP_n} \times \overrightarrow{OP_{n+1}}\right) \\ &= \frac{1}{2}\left(\sum_{i=1}^{n} \overrightarrow{OP_i} \times \overrightarrow{OP_{i+1}}\right) \end{aligned} \qquad (8.12)$$

(quy ước $P_{n+1} \equiv P_1$).

Khi các đỉnh của đa giác được đánh số theo chiều nghịch (theo chiều kim đồng hồ), bằng lập luận tương tự, ta có công thức diện tích đa giác là

$$S = -\frac{1}{2}\left(\sum_{i=1}^{n} \overrightarrow{OP_i} \times \overrightarrow{OP_{i+1}}\right). \qquad (8.13)$$

Tổng hợp lại, công thức tính diện tích đa giác có thể viết thành:

$$S = \frac{1}{2}\left|\sum_{i=1}^{n} \overrightarrow{OP_i} \times \overrightarrow{OP_{i+1}}\right| \tag{8.14}$$

mà không cần quan tâm tới đa giác được đánh số theo chiều thuận hay chiều nghịch.

Bây giờ ta sẽ sử dụng tới toạ độ để xây dựng công thức tính diện tích đa giác dựa trên các toạ độ đỉnh. Vectơ $\overrightarrow{OP_i}$ có toạ độ $(x_i; y_i)$, dựa vào công thức toạ độ của tích chéo, ta có:

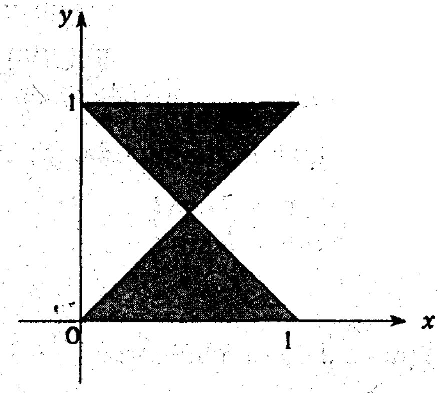
$$\begin{aligned} S &= \frac{1}{2}\left|\sum_{i=1}^{n}(x_i y_{i+1} - x_{i+1} y_i)\right| \\ &= \frac{1}{2}\left|\sum_{i=1}^{n}(x_{i-1} - x_{i+1}) y_i\right|. \end{aligned} \tag{8.15}$$

Việc chứng minh đẳng thức thứ hai có thể sử dụng vài phép biến đổi đại số. Ở đây ta có thể hình dung như sau: Do cơ chế quay vòng số thứ tự (đỉnh 0 trùng với đỉnh n và đỉnh $n+1$ trùng với đỉnh 1), với mỗi đỉnh i sẽ có đúng một hạng tử $+x_{i-1}y_i$ và một hạng tử $-x_{i+1}y_i$ xuất hiện trong tổng $\sum_{i=1}^{n}(x_i y_{i+1} - x_{i+1} y_i)$. Ta có thể ghép hai hạng tử này lại với thừa số chung y_i.

Nếu như việc tính đẳng thức thứ nhất cần $2n$ phép nhân thì ở đẳng thức thứ hai ta chỉ cần n phép nhân mà thôi (số phép cộng và phép trừ không thay đổi). Chính vì vậy, đẳng thức thứ hai hiệu quả hơn khi lập trình tính diện tích đa giác. ■

Chú ý: Khi sử dụng công thức (8.11):

- Ta có thể xác định phép đánh số các đỉnh đa giác là thuận hay nghịch dựa vào công thức tính diện tích: Bỏ dấu giá trị tuyệt đối trong công thức (8.11), khi đó $S > 0$ tương ứng với phép đánh số thuận và $S < 0$ tương ứng với phép đánh số nghịch.
- Đường biên đa giác phải là một đường gấp khúc khép kín, không tự cắt. Công thức (8.11) sẽ sai nếu dữ liệu vào không thoả mãn điều kiện này.
- Hình 8.9 là một ví dụ về một "đa giác" gồm bốn đỉnh $(0; 0)$, $(1; 1)$, $(0; 1)$ và $(1; 0)$.



Hình 8.9

Công thức (8.11) sẽ tính được diện tích miền mặt phẳng giới hạn bởi đường gấp khúc này là 0, tuy nhiên trên thực tế thì diện tích miền này là 0.5.

c) Diện tích đường cong

Trên mặt phẳng cho C là một đường cong khép kín, không tự cắt, trơn từng khúc, có phương trình tham số:

$$\begin{cases} x = x(t) \\ y = y(t) \end{cases}$$

với $t \in [a, b]$, $x(a) = x(b)$ và $y(a) = y(b)$.

Ta có thể hình dung khi t chạy từ a tới b thì các điểm $(x(t); y(t))$ vẽ ra trên mặt phẳng trực chuẩn một đường cong khép kín không tự cắt C. Đường cong C trơn từng khúc tức là ta có thể chia khoảng $[a; b]$ thành một số đếm được các khoảng con mà trên mỗi khoảng con đó, các đạo hàm $x'(t)$ và $y'(t)$ là các hàm liên tục.

Khi đó, diện tích của miền mặt phẳng D giới hạn bởi đường cong C có thể tính qua một hệ quả của công thức Green trong giải tích hàm:

$$\begin{aligned} S &= \frac{1}{2}\int_C xdy - ydx \\ &= \frac{1}{2}\int_a^b \big(x(t)y'(t) - y(t)x'(t)\big)dt. \end{aligned} \tag{8.16}$$

Chú ý: Công thức (8.16) cho kết quả là diện tích có dấu của miền D (là số âm nếu miền D được định hướng nghịch). Có thể thêm vào công thức dấu giá trị tuyệt đối nếu chỉ cần biết diện tích.

Thực ra công thức tính diện tích đa giác mà ta trình bày ở mục trước chỉ là trường hợp riêng của công thức (8.16). Việc chứng minh công thức (8.16) cần phải sử dụng nhiều kiến thức của toán cao cấp, ta chỉ cần nhớ và áp dụng.

Ví dụ: Xét đường tròn tâm O bán kính R, đường tròn này có phương trình tham số

$$\begin{cases} x = R\cos t \\ y = R\sin t \end{cases}$$

với $t \in [0; 2\pi]$.

Áp dụng công thức (8.16), ta có diện tích hình tròn:

$$S = \frac{1}{2}\int_0^{2\pi} (R^2\cos^2 t + R^2\sin^2 t)dt = \frac{1}{2}R^2\int_0^{2\pi} dt = \pi R^2.$$

III. Một số bài toán thông dụng khác

1. Tam giác

Xét một tam giác xác định bởi toạ độ ba đỉnh là ba điểm $A(A.x; A.y)$, $B(B.x; B.y)$, $C(C.x; C.y)$. Kí hiệu a, b, c là độ dài ba cạnh của tam giác ABC.

a) Đường tròn ngoại tiếp tam giác

Tâm đường tròn ngoại tiếp tam giác ABC là giao điểm của hai đường trung trực thuộc hai cạnh của tam giác. Đường trung trực của cạnh AB được xác định thông qua trung điểm $M\left(\frac{A.x+B.x}{2}; \frac{A.y+B.y}{2}\right)$ của cạnh AB và vectơ chỉ phương là vectơ pháp tuyến của đường thẳng AB.

b) Đường phân giác

Đường phân giác của góc $\widehat{ABC}$ được xác định thông qua dạng biểu diễn $(P, \vec{d})$ trong đó P chính là điểm B còn vectơ $\vec{d}$ xác định theo công thức sau:

$$\vec{d} = \frac{\overrightarrow{BA}}{|\overrightarrow{BA}|} + \frac{\overrightarrow{BC}}{|\overrightarrow{BC}|}$$

c) Đường tròn nội tiếp tam giác

Tâm của đường tròn nội tiếp tam giác ABC có thể xác định bằng cách tìm giao điểm của hai đường phân giác. Tuy nhiên, ta có thể sử dụng công thức toán học để xác định toạ độ tâm của đường tròn nội tiếp chính là toạ độ của vectơ:

$$\frac{a.\overrightarrow{OA} + b.\overrightarrow{OB} + c.\overrightarrow{OC}}{a + b + c}$$

trong đó O là gốc toạ độ.

Bán kính đường tròn nội tiếp bằng

$$r = \frac{2S}{a + b + c}$$

với S là diện tích tam giác.

2. Kiểm tra điểm nằm trong đa giác

Bài toán. Trên mặt phẳng với hệ toạ độ Đề-các vuông góc, cho đa giác $P = P_1P_2 \dots P_n$ và một điểm A. Hãy cho biết điểm A có nằm trong đa giác hay không.

Thuật toán

Nếu A thuộc một cạnh đa giác thì kết luận ngay A nằm trong đa giác. Nếu không, từ điểm A, xác định một tia gốc A không đi qua đỉnh nào của đa giác, gọi tia này là tia AB. Có nhiều cách chọn tia này, chẳng hạn như chọn ngẫu nhiên $n + 1$ tia đôi một khác nhau, khi đó chắc chắn sẽ có một tia không đi qua đỉnh nào của đa giác. Tuy nhiên trên thực tế, phương pháp thực dụng tỏ ra hiệu quả hơn: sinh ngẫu nhiên một tia, nếu tia đó đi qua một đỉnh của đa giác thì sinh ngẫu nhiên một tia khác và thử lại...

Nếu tia AB cắt cạnh đa giác một số lẻ lần thì điểm A nằm trong đa giác, ngược lại thì điểm A nằm ngoài đa giác.

Tất cả những kĩ thuật tìm giao điểm, xác định tia đã được nói đến trong các bài trước, ta chỉ tổng hợp lại các kĩ thuật này để lập trình giải bài toán điểm nằm trong đa giác.

Input

- Dòng thứ nhất chứa số nguyên $n \le 10^6$ là số đỉnh của đa giác và hai số thực xA, yA tương ứng là hoành độ và tung độ điểm A;

- n dòng tiếp theo, dòng thứ i chứa hai số thực x_i, y_i là toạ độ đỉnh P_i của đa giác.

Output

Cho biết điểm A có nằm trong đa giác P hay không?

PTINPOLYGON.PAS ✓ Kiểm tra điểm nằm trong đa giác

```
{$MODE OBJFPC}
program PointInPolygon;
uses Math;
const
 maxN = 1000000; epsilon = 1E-6;
type
 TPoint = record
            x, y: Float;
          end;
 TVector = TPoint;
var
 p: array[1..maxN + 1] of TPoint;
 n: Integer; A, B: TPoint;
 Inside: Boolean;
function Vector(x, y: Float): Tvector;
begin
 Result.x := x; Result.y := y;
end;
//Phép trừ vectơ
operator -(const u, v: TVector): TVector;
begin
 Result.x := u.x - v.x;
 Result.y := u.y - v.y;
end;
//Tích chấm của hai vectơ
operator *(const u, v: TVector): Float;
begin
 Result := u.x * v.x + u.y * v.y;
end;
//Tích chéo của hai vectơ
operator ><(const u, v: TVector): Float;
begin
 Result := u.x * v.y - u.y * v.x;
end;
```

```
procedure Enter;
var i: Integer;
begin
 ReadLn(n, A.x, A.y);
 for i := 1 to n do
   with p[i] do
     begin
       ReadLn(x, y);
     end;
   p[n + 1] := p[1];
end;
//Kiểm tra điểm nằm trên đoạn thẳng
function OnSegment(const P, Q: TPoint): Boolean;
begin
 Result := IsZero(P >< Q, epsilon) and (P * Q <= 0)
end;
//Tạo tia
procedure MakeRay;
var
 OK: Boolean;
 i: Integer;
begin
 repeat
   OK := True; B.x := Random; B.y := Random;
   for i := 1 to n do
     if IsZero((B - A) >< p[i], epsilon) then
       begin
         OK := False;
         Break;
       end;
 until OK;
end;
//Biểu diễn tuyến tính
function SolveSLE(const a, b, c: TVecto): TVecto;
var D: Float;
begin
 D := a >< b;
 Result := Vecto((c >< b)/D, (a >< c)/D);
end;
//Kiểm tra tia cắt cạnh đa giác
function Cut(const C, D: TPoint): Boolean;
var r: TVecto;
```

```
begin
 r := SolveSLE(B - A, C - D, C - A);
 Result := (r.x >= 0) and InRange(r.y, 0, 1);
end;
function Check: Boolean;
var i: Integer;
begin
 for i := 1 to n do
   if OnSegment(p[i], p[i + 1]) then Exit(True);
 MakeRay;
 Result := False;
 for i := 1 to n do
   if Cut(p[i], p[i + 1]) then Result := not Result;
end;
//Chương trình chính
BEGIN
SetExceptionMask([Low(TFPUExceptionMask)..High(TFPUExceptionMask)]);
 Enter;
 if Check then WriteLn('Inside the polygon')
 else WriteLn('Outside the polygon');
END.
```

3. *Tìm hai điểm gần nhất*

Cho một tập điểm Q (có thể trùng nhau). Tìm cặp điểm trong Q có khoảng cách nhỏ nhất.

Nếu kiểm tra tất cả các cặp điểm ta sẽ có một thuật toán với độ phức tạp $O(N^2)$. Thuật toán sau đây có độ phức tạp $O(N\log N)$ giải quyết vấn đề bằng cách chia để trị. Với một bài toán kích thước N, ta đưa về thuật toán giải hai bài toán nhỏ kích thước $\frac{N}{2}$, và kết hợp kết quả trong thời gian $O(N)$.

Q được biểu diễn bởi hai mảng đã sắp xếp X (theo hoành độ) và Y (theo tung độ). Thuật toán như sau:

Bước 1. Dùng một đường thẳng đứng d, ta có thể chia đôi Q thành Q_1 và Q_2. Các điểm nằm đúng trên d có thể phân bố tuỳ ý sao cho Q_1 và Q_2 cân bằng.

Bước 2. Gọi đệ quy tìm khoảng cách ngắn nhất d_1 trong Q_1 và d_2 trong Q_2. Đặt $d_0 = \min(d_1,d_2)$.

Bước 3. Kết quả trả về sẽ là d_0, hoặc một giá trị nhỏ hơn d_0 nếu tồn tại một điểm trong Q_1 và một điểm trong Q_2 có khoảng cách nhỏ hơn d_0. Các điểm như vậy chỉ có thể nằm trong khoảng cách d_0 từ đường thẳng d. Tìm tập hợp Q_0 các điểm thoả mãn đường thẳng đó, cùng với Y_0 tương ứng của Q_0.

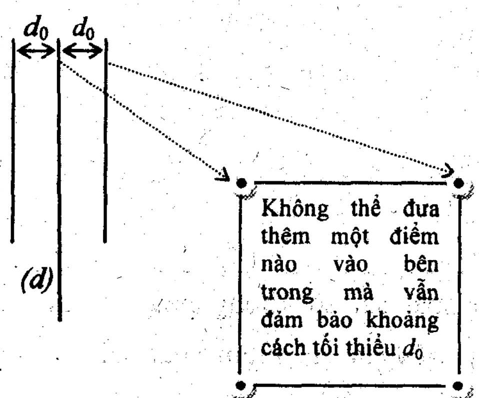
Bước 4. Với mỗi điểm P_i trong Q_0, ta xét khả năng P_i là đầu mút phía trên của cặp điểm gần nhau nhất. Như vậy, điểm còn lại phải nằm trong hình chữ nhật $d_0 \times 2d_0$ có cạnh trên đi qua P_i. Hình chữ nhật này có tối đa bốn điểm trong Q_1 và bốn điểm trong Q_2. Do đó nó có tối đa tám điểm trong Q_0. Như vậy, ta chỉ cần kiểm tra khoảng cách của P_i đến bảy điểm theo sau nó trong Y_0.

Bước 5. Trả về khoảng cách ngắn nhất tìm được.

Tại bước 2, ta cần X_1, X_2, Y_1, Y_2 tương ứng cho Q_1, Q_2. Các mảng này có thể được sinh ra trong thời gian $O(N)$.

Tại bước 3, Y_0 cũng được sinh ra trong $O(N)$ bằng cách đọc tuần tự mảng Y và loại ra các điểm cách xa d.

Các bước 1, 2, 3, 4 đảm bảo chi phí xử lí $O(N)$ tại mỗi lần đệ quy, từ đó cho ta thuật toán có độ phức tạp $O(N\log N)$.



Hình 8.10. Khoảng cách tối thiểu

Minh hoạ. Do các điểm trong Q_1 có khoảng cách tối thiểu d_0, từ P_i trở xuống chỉ có tối đa bốn điểm thuộc về Q_1.

4. *Một số thủ tục cơ bản*

a) Kiểm soát sai số

Khi tính toán các công thức hình học trên máy tính, có rất nhiều phép tính không thể đạt được độ chính xác tuyệt đối như phép chia, phép khai căn bậc 2... Ngay cả những phép cộng và nhân số thực trên máy tính cũng phải chịu sai số ở chữ số có nghĩa cuối cùng.

Sai số trong các phép tính có thể dẫn tới nhiều phiền toái, vì thế phải có phương pháp để kiểm soát sai số này trong quá trình tính toán. Kinh nghiệm chung là:

- Nếu dữ liệu vào là các số nguyên, ta cố gắng sử dụng các phép tính toán số nguyên đến chừng nào còn có thể được.
- Nếu bắt buộc phải tính toán số thực, trong một số trường hợp ta cần phải dự kiến sai số tính toán. Chẳng hạn, chọn một hằng số ϵ đủ nhỏ (chẳng hạn $\epsilon = 10^{-6}$), sau đó thay vì so sánh $p = q$, ta viết $|p - q| < \epsilon$.

Ví dụ, ta sẽ viết hàm $OnSegment(xA, yA, xB, yB, xC, yC)$ để kiểm tra xem điểm $C = (xC; yC)$ có nằm trên đoạn thẳng AB với $A = (xA; yA)$, $B = (xB; yB)$ hay không:

```
Const epsilon = 1E-6;
function OnSegment(xA, yA, xB, yB, xC, yC: Float): Boolean;
var
 x1, y1, x2, y2: Float;
begin
 x1 := xC - xA; y1 := yC - yA; //(x1, y1) = vectơ AC
 x2 := xC - xB; y2 := yC - yB; //(x2, y2) = vectơ BC
 Result :=
          (Abs(x1 * y2 - x2 * y1) < epsilon) and
                                   (x1 * x2 + y1 * y2 ≤ 0)
end;
```

Việc kiểm soát sai số không phải lúc nào cũng cần thực hiện. Cụ thể trong trường hợp trên, ta chỉ kiểm soát sai số trong phép so sánh $\overrightarrow{AC} \times \overrightarrow{BC} = 0$ mà không cần kiểm soát sai số trong phép so sánh $\overrightarrow{AC}.\overrightarrow{BC} \leq 0$. Lí do là bởi vì hàm $OnSegment$ bản thân nó đã có sai số (nếu điểm C nằm rất sát đoạn thẳng AB nhưng không nằm trên đoạn thẳng AB, hàm vẫn có thể trả về giá trị *True*) và sai số của phép so sánh $\overrightarrow{AC}.\overrightarrow{BC} \leq 0$ được tính luôn vào sai số của hàm.

b) Các thủ tục, hàm thường dùng

Một trong những khó khăn gặp phải khi giải quyết các bài toán hình học tính toán là ta phải làm việc với số thực và thường gặp nhiều sai sót trong khi lập trình vì chương trình thường rất dài và cần phải cài đặt tỉ mỉ, chi tiết. Những thủ tục này giúp bạn tránh được những sai sót đáng tiếc và giúp bạn cài đặt nhanh các thuật toán cho bài toán hình học.

```
Type Real=Extended;
 TPoint=Record
         x,y:real;
       end;
Const _Eps: Real=1e-3;
     ZeroPnt: TPoint=(X:0; Y:0);
/* so sánh hai số thực */
Function RealEq(Const a, b:real): Boolean;
Begin
 RealEq:=Abs(a-b)<=_Eps;
End;
Function RealMax(Const a, b:Real):Real;
Begin
 If RealMore(a,b) Then RealMax:=a Else RealMax:=b;
End;
/* Kiểm tra hai điểm trùng nhau */
Function EqPoint(Const A, B:TPoint):Boolean;
Begin
 EqPoint:=RealEq(A.x, B.x) And RealEq(A.y, B.y);
End;
/* Khoảng cách hai điểm */
Function Dist(Const A, B:TPoint):Real;
Begin
 Dist:=Sqrt(Sqr(A.x-B.x)+Sqr(A.y-B.y));
End;
/* Tổng vectơ */
Type TVecCart=TPoint;
Procedure AssVecCart(Const a, b:TVecCart;Var c:TVecCart);
Begin
 c.x:=a.x+b.x;
c.y:=a.y+b.y;
End;
/* Tích vô hướng */
Function SkMulCart(Const a, b:TVecCart):Real;
Begin
 SkMulCart:=a.x*b.x+a.y*b.y;
End;
Type TVecPol=Record
               rst, angle:Real
             End;
```

```
/* Góc với trục Ox, đo bằng radian */
Function GetAngle(Const x, y:Real):Real;
Var rs, c:Real;
Begin
 rs:=Sqrt(Sqr(x)+Sqr(y));
 If RealEq(rs, 0) Then GetAngle:=0
 Else
   Begin
     c:=x/rs;
     If RealEq(c, 0) Then c:=Pi/2
     Else c:=ArcTan(Sqrt(Abs(1-Sqr(c)))/c);
     If RealLess(c, 0) Then c:=Pi+c;
     If RealLess(y, 0) Then c:=2*Pi-c;
     GetAngle:=c;
  End;
End;
/* toạ độ Đề-các chuyển sang toạ độ cực */
Procedure CartToPol(Const a:TVecCart;Var b:TVecPol);
Begin
 b.rs:=Sqrt(Sqr(a.x)+Sqr(a.y));
 b.angle:=GetAngle(a.x, a.y);
End;
/* Toạ độ cực chuyển sang toạ độ Đề-các*/
Procedure PolToCart(Const a:TVecPol;Var b:TVecCart);
Begin
 b.x:=a.rs*cos(a.angle);
 b.y:=a.rs*sin(a.angle);
End;
Type TLine=Record
            A,B,C:Real;
          End;
/* Viết phương trình đường thẳng đi qua hai điểm */
Procedure Point2ToLine(Const v, w:Tpoint;Var L:TLine);
Begin
 L.A:=v.y -w.y;
 L.B:=w.x- v.y;
 L.C:=-(v.x*L.A+v.y*L.B);
End;
```

```
/* Hai đường thẳng có khác phương hay không */
Function CrossLine(Const L1,L2:TLine):Boolean;
Var st:Real;
Begin
 st:=L1.A*L2.B-L2.A*L1.B;
 CrossLine:=Not RealEq(st,0);
End;
/* Giao điểm hai đường thẳng */
Procedure Line2ToPoint(Const L1,L2:TLine;Var P:TPoint);
Var st:Real;
Begin
 st:=L1.A*L2.B-L2.A*L1.B;
 P.X:=(L1.C*L2.B-L2.C*Lq.B)/st;
 P.Y:=(L1.A*L2.C-L2.A*L1.C)/st;
End;
/* Tìm giao điểm hai đường thẳng */
Procedure FindPointCross(Const fL, fR, sL, sR:TPoint;Var rs:TPoint);
Var L1,L2:TLine;
Begin
 Point2ToLine(fL,fR,L1); Point2ToLine(sL,sR,L2);
 If CrossLine(L1,L2) Then Line2ToPoint(L1,L2,rs)
 Else rs:=ZeroPnt;
End;
/* Input: 2 đoạn thẳng nằm trên cùng đường thẳng
Nhiệm vụ: Kiểm tra vị trí tương đối
Output: 0: có 1 điểm chung; 2: có 1 đoạn chung; 1: không giao nhau
*/
Function SegmLineCross(Const fL,fR,sL,sR:Tpoint):Byte;
Var Minf,Maxf,Mins,Maxs:Real;
Begin
 Minf:=RealMin(Dist(fL,ZeroPnt),Dist(fR,ZeroPnt));
 Maxf:=RealMax(Dist(fL,ZeroPnt),Dist(fR,ZeroPnt));
 Mins:=RealMin(Dist(sL,ZeroPnt),Dist(sR,ZeroPnt));
 Maxs:=RealMax(Dist(sL,ZeroPnt),Dist(sR,ZeroPnt));
If RealEq(Minf,Maxs) or RealEq(Maxf,Mins)
 Then SegmLineCross:=0
 Else
 If RealMore(Mins,Maxf) Or RealMore(Minf,Maxs)
 Then SegmLineCross:=1
 Else SegmLineCross:=2;
End;
```

```
/* Kiểm tra đoạn thẳng cắt nhau.
Thuật toán: tìm giao điểm, sau đó kiểm tra giao điểm nằm trên đoạn thẳng
Trả về một số nguyên đánh số các trường hợp vị trí tương đối đoạn thẳng-đoạn thẳng.
Hai đoạn thẳng cắt nhau sẽ trả về 7
*/
Function SegmCross(Const fL,fR,sL,sR :TPoint);
Var rs:TPoint;
 L1,L2:TLine;
Begin
 Point2ToLine(fL,fR,L1);
 Point2ToLine(sL,sR,L2);
 If CrossLine(L1,L2)Then
 Begin
 Line2ToPoint(L1,L2,rs)
 If EqPoint(rs,fL) Or EqPoint(rs,fR)
     Or EqPoint(rs,sL) Or EqPoint(rs,sR)
 Then SegmCross:=5
 Else
 If AtSegm(fL,fR,rs) And AtSegm(sL,sR,rs) Then SegmCross:=7
 Else
 If AtSegm(fL,fR,rs) Or AtSegm(sL,sR,rs) Then SegmCross:=6
 Else SegmCross:=4;
 End
 Else
 If EqPoint(L1.A*L2.B,L2.A*L1.B)
     And Not (EqPoint(L1.C,L2.C)
 Then SegmCross:=3
 Else SegmCross:=SegmLineCross(fL,fR,sL,sR);
End;
/* Đường thẳng vuông góc */
Procedure PerLine(Const n:TLine;Const P:TPoint;Var L:TLine);
Begin
 L.A:=n.B;
L.B:=n.A;
L.C:=L.A*P.X-L.B*P.Y;
End;
/* Khoảng cách từ một điểm đến một đường thẳng */
Function DistPointToLine(Const P:TPoint;Const L:TLine):Real;
Begin
 DistPointToLine :=
        Abs((L.A*P.x+L.B*P.y+L.C))/Sqrt(Sqr(L.A)+Sqr(L.B));
End;
```

```
/* Kiểm tra điều kiện bất đẳng thức tam giác */
Function IsTrian(Const a,b,c:Real):Boolean;
Begin
 IsTrian:=RealMore(a+b,c) And RealMore(a+c,b)
                          And RealMore(b+c,a);
End;
/* Diện tích tam giác bằng công thức Hê-rông */
Function Sq(a,b,c:Real):Real;
Var p:Real;
Begin
 p:=(a+b+c)/2; Sq:=Sqrt(p*(p-a)*(p-b)*(p-c));
End;
/* Diện tích tam giác khi biết tọa độ ba đỉnh */
Function SquareTrian(Const p1,p2,p3:TPoint);
Begin
 SquareTrian := Abs(p1.x*(p2.y-p3.y)-p1.y*(p2.x-p3.x)
                          +(p2.x*p3.y-p3.x*p2.y))/2);
End;
/* Chiều cao tam giác ứng với cạnh a */
Function GetHeight(Const a,b,c:Real):Real;
Var p:Real;
Begin
 p:=(a+b+c)/2; GetHeight:=2*Sqrt(p*(p-a)*(p-b)*(p-c))/a;
End;
/* Tìm độ dài trung tuyến */
Function GetMed(Const a,b,c:Real):Real;
Begin
 GetMed:=Sqrt(2*(b*b+c*c)-a*a)/2;
End;
/* Độ dài đường phân giác xuất phát từ một đỉnh tới cạnh đối diện có độ dài bằng a */
Function GetBis(Const a,b,c:Real):Real;
Var p:Real;
Begin
 p:=(a+b+c)/2; GetBis:=2*Sqrt(b*c*p*(p-a))/(b+c);
End;
/* Bán kính đường tròn nội tiếp */
Function GetRadIns(Const a,b,c:Real):Real;
Var p:Real;
Begin
 p:=(a+b+c)/2; GetRadIns:=Sqrt((p-a)*(p-b)*(p-c)/p);
End;
```

```
/* Bán kính đường tròn ngoại tiếp */
Function GetRadExt(Const a,b,c:Real):Real;
Var p:Real;
Begin
 p:=(a+b+c)/2;
 GetRadExt:=a*b*c/(4*Sqrt(p*(p-a)*(p-b)*(p-c)));
End;
/* Kiểm tra đa giác không tự cắt */
Function IsPoligonSimple
            (Const A:Array Of TPoint;Const N:Word):Boolean;
Var i,j:Integer;
 pp:Boolean;
Begin
 pp:=True;
 i:=1;
 While (i<=N-1) And pp Do
   Begin
     j:=i+1;
     While (j<=N) And pp Do
       Begin
         Case SegmCross(A[i],A[i+1]],A[j],A[j+1]) Mod N
              Of 0,2,6,7: pp:=False;
         End;
         Inc(j);
       End;
     Inc(i);
   End;
  IsPoligonSimple:=pp;
End;
```

c) Thư viện Math và các hằng số không xác định

Free Pascal cung cấp thư viện Math để hỗ trợ cho các phép tính toán học chuyên dụng. Ngoài một hệ thống phong phú các thủ tục và hàm, thư viện Math cung cấp ba hằng số đặc biệt *Nan*, *Infinite* và *NegInfinite*.

- *Nan* tương ứng với một số thực không xác định (Not a Number), trong thư viện Math, hằng *Nan* được khai báo bằng 0.0/0.0.
- *Infinite* tương ứng với hằng số thực $+\infty$, phép chia một số thực dương cho 0 sẽ cho kết quả bằng *Infinite*.

- *NegInfinite* tương ứng với hằng số thực $-\infty$, phép chia một số thực âm cho 0 sẽ cho kết quả bằng *NegInfinite*.

Ví dụ đoạn chương trình sau:

```
program TestDiv0;
uses math;
var
 x, y, z: Float;
 zero: Float = 0;
begin
SetExceptionMask([Low(TFPUExceptionMask)..High(TFPUExceptionMask)]);
 x := 0/zero;
 y := 1/zero;
 z := -2/zero;
 WriteLn(x); //In ra Nan
 WriteLn(y); //In ra +Inf
 WriteLn(z); //In ra -Inf
end.
```

Ba hằng số này rất tiện dụng trong việc lập trình các bài toán hình học, chẳng hạn khi tìm giao điểm của hai đường thẳng, nếu hai đường thẳng đó trùng nhau, ta có thể gán toạ độ giao điểm là $(Nan; Nan)$, còn nếu hai đường thẳng song song với nhau, ta có thể gán toạ độ giao điểm là $(Infinite; Infinite)$. Chương trình chính chỉ việc gọi hàm để lấy toạ độ giao điểm, sau đó kiểm tra các toạ độ có khác *Nan* và khác *Infinite* hay không để xử lí tiếp.

Những chú ý sau rất quan trọng khi sử dụng các hằng *Nan*, *Inifinite* và *NegInfinite*:

Chú ý:

- Việc chia cho 0 để lấy hằng *Nan, Infinite, NegInfinite*, cũng như việc so sánh trực tiếp với các hằng số này, chỉ thực hiện được trong lúc chạy (run-time) nếu ta đặt lại thiết lập trong bộ đồng xử lí toán học bằng lệnh:

```
SetExceptionMask([Low(TFPUExceptionMask)..High(TFPUExceptionMask)]);
```

- Việc chia trực tiếp cho 0, gán, so sánh các hằng *Nan, Infinite, NegInfinite* với một biến số thực chỉ được phép viết khi dịch chương trình ở chế độ {\$*R*-,*Q*-}, tức là tất cả các chế độ kiểm tra tràn phạm vi và tràn số học phải được tắt. Tuy nhiên, việc tắt hoàn toàn chế độ kiểm tra tràn phạm vi và tràn số học có thể rất nguy hiểm và khó gỡ rối, vì vậy ta nên làm theo cách khác như sau:

- Sử một biến $zero$ gán bằng 0 và thực hiện phép chia cho $zero$ thay vì chia cho 0, kĩ thuật này nhằm đánh lừa trình biên dịch.
- Không so sánh trực tiếp một biểu thức số thực với các hằng $Nan, Infinite, NegInfinite$ mà sử dụng các hàm cung cấp sẵn trong thư viện Math:
  - Hàm $IsNan(x)\colon Boolean$: Kiểm tra số thực x có phải bằng Nan hay không.
  - Hàm $IsInifinite(x)\colon Boolean$: Kiểm tra số thực x có phải bằng $Infinite$ hay $NegInifinite$ hay không.
  - Hàm $IsZero(x, \varepsilon)\colon Boolean$: Kiểm tra $x \approx 0$, cụ thể là $|x|$ có nhỏ hơn ε hay không, hàm này có thể dùng để xử lí sai số. ε là sai số chấp nhận do người dùng tuỳ chọn.

d) Thư viện Matrix

Trong các bài toán ví dụ, chúng tôi chỉ sử dụng thư viện Math để hỗ trợ cho việc tính toán được ngắn gọn. Nếu muốn thực hiện các tính toán hình học phức tạp, trên không gian nhiều chiều, bạn có thể sử dụng thêm thư viện Matrix. Thư viện này cung cấp các phép toán thao tác trực tiếp trên các vectơ và ma trận: tính tích ma trận, nghịch đảo ma trận, định thức của ma trận v.v...

Hiện tại, các thuật toán trong thư viện Matrix vẫn là những thuật toán tuần tự. Tuy nhiên trong tương lai, sẽ có những phiên bản với thuật toán song song, sử dụng kĩ thuật đa xử lí để tăng tốc quá trình tính toán. Một số dự án đang viết lại thư viện Matrix, dùng sức mạnh của hàng nghìn GPU trong card đồ hoạ để tính toán thay vì sử dụng CPU, các phép toán ma trận được tăng tốc lên hàng trăm lần và các chương trình sử dụng thư viện này theo đó cũng được tăng lên đáng kể.

Bài tập

8.1. Trên mặt phẳng cho một tam giác và một đoạn thẳng. Tính độ dài của đoạn thẳng nằm trong tam giác.

8.2. Trên mặt phẳng cho hai tam giác. Tính diện tích phần chung của hai tam giác.

8.3. Cho hai hình chữ nhật với các cạnh song song với hệ trục toạ độ. Mỗi hình chữ nhật xác định bởi toạ độ hai đỉnh đối. Tìm diện tích phần mặt phẳng được phủ bởi hai hình chữ nhật và diện tích phần chung của chúng.

8.4. Trên mặt phẳng cho N điểm. Tìm hai điểm xa nhau nhất trong N điểm đó.

Gợi ý: Hai điểm cần tìm là hai đỉnh của đa giác bao lồi.

8.5. Trên mặt phẳng cho N điểm. Hãy phân hoạch N điểm thành K tập $S_1, S_2, \ldots, S_K$ sao cho đường kính lớn nhất của K tập là nhỏ nhất, trong đó đường kính của tập S_i là khoảng cách lớn nhất giữa các cặp đỉnh thuộc S_i.

8.6. Trên mặt phẳng cho N điểm đôi một khác nhau. Hãy tìm hai trong N điểm sao cho đường thẳng đi qua hai điểm đó chia mặt phẳng thành hai phần mà số điểm thuộc hai nửa mặt phẳng lệch nhau ít nhất.

8.7. Trên mặt phẳng cho N điểm. Hãy tìm tập ít nhất các đường thẳng sao cho mỗi điểm trong N điểm đã cho thuộc ít nhất một đường thẳng.

8.8. Trên mặt phẳng cho N điểm. Hãy nối N điểm thành một mạng liên thông bằng các đoạn thẳng nối các cặp điểm sao cho tổng độ dài các đoạn thẳng được nối là nhỏ nhất.

8.9. Trên mặt phẳng cho N hình chữ nhật có các cạnh song song với hai trục toạ độ. Hình thứ i xác định bởi hai đỉnh đối $(x_i; y_i)$ và $(z_i; t_i)$. Toạ độ các điểm là nguyên; có giá trị tuyệt đối không vượt quá 10^4. Khi tô các hình chữ nhật này bằng một màu xanh, ta nhận được một số phần mặt phẳng được tô màu xanh. Tính tổng độ dài các đường bao quanh các vùng màu xanh.

8.10. *Tầm nhìn*

Trên mặt phẳng cho N đoạn thẳng ($1 \le N \le 1000$). Toạ độ các đầu mút của các đoạn thẳng là các số nguyên không âm không vượt quá 20000. Các đường thẳng thu được bằng cách kéo dài các đoạn thẳng đã cho luôn cắt hai trục toạ độ và hai giao điểm cùng gốc toạ độ tạo thành một tam giác vuông cân. Không có hai đoạn thẳng nào giao nhau.

Ta nói một đoạn thẳng là nhìn thấy được từ gốc toạ độ O, nếu tìm được điểm X trên nó sao cho đoạn thẳng OX không cắt bất cứ đoạn nào khác trong số các đoạn thẳng đã cho.

Hãy viết chương trình đếm số đoạn thẳng nhìn thấy được từ gốc toạ độ.

Input: Vào từ tệp VPOINT.INP:

- Dòng đầu tiên chứa số đoạn thẳng N;
- Mỗi một trong số N dòng tiếp theo chứa bốn số nguyên không âm X_1, Y_1, X_2 và Y_2, phân cách nhau bởi dấu cách, trong đó (X_1, Y_1) là toạ độ của đầu mút thứ nhất còn (X_2, Y_2) là toạ độ của đầu mút thứ hai của đoạn thẳng tương ứng.

Output: Ghi ra trên một dòng của tệp VPOINT.OUT số đoạn thẳng nhìn thấy được từ gốc toạ độ.

Ví dụ:

VPOINT.INP	VPOINT.OUT
4 3 13 11 5 14 1 10 5 10 14 20 4 5 6 10 1	3

8.11. *Khôi phục đa giác*

Bờm vẽ trên mặt phẳng một hình đa giác tổng quát (đường gấp khúc khép kín không tự cắt) với các cạnh song song với các trục toạ độ và các đỉnh có toạ độ nguyên. Sau đó, vì vô ý Bờm đã xoá mất tất cả các cạnh thẳng đứng (cạnh song song với trục tung) của đa giác. Bạn hãy tìm cách từ những thông tin còn lại trên hình vẽ giúp Bờm tính diện tích của đa giác đã vẽ ban đầu.

Input: Vào từ tệp văn bản POLYGON.INP:

- Dòng đầu tiên chứa N là số cạnh nằm ngang (cạnh song song với trục hoành) của đa giác đã cho ($N \le 1000$) ;
- Mỗi dòng trong số N dòng tiếp theo chứa thông tin về một cạnh nằm ngang của đa giác bao gồm bốn số nguyên x, y, u, v được ghi cách nhau bởi dấu cách, trong đó (x, y) và (u, v) là hai cặp toạ độ của hai đầu mút

của cạnh nằm ngang. Giả thiết rằng các toạ độ là các số nguyên có giá trị tuyệt đối không vượt quá 100.

Output: Ghi ra tệp văn bản POLYGON.OUT:

- Dòng đầu tiên ghi diện tích của đa giác.
- Dòng thứ *i* trong số 2*N dòng tiếp theo chứa toạ độ đỉnh thứ *i* của đa giác được liệt kê theo thứ tự đi vòng quanh đa giác theo chiều kim đồng hồ (đỉnh bắt đầu được chọn tuỳ ý).

Ví dụ:

POLYGON.INP	POLYGON.OUT
2 1 1 3 1 1 3 3 3	4 1 1 1 3 3 3 3 1

8.12. *Đỉnh núi*

Đỉnh núi có thể nhìn thấy được ở chân trời chỉ khi nó không bị che khuất bởi một ngọn núi khác hay là đường viền của nó không bị ẩn trên nền một ngọn núi khác. Giả thiết rằng tất cả các dốc núi đều có góc nghiêng là 45^0 đồng thời cho biết toạ độ và độ cao đỉnh của tất cả các ngọn núi. Cần xác định số lượng đỉnh núi nhìn thấy được.

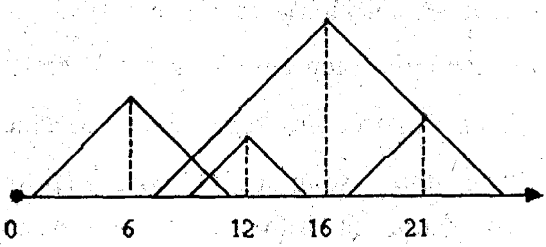
Input: Vào từ tệp văn bản MOUNT.INP:

- Dòng đầu tiên chứa số nguyên N ($1 \le N \le 10000$).
- Mỗi dòng trong N dòng tiếp theo chứa hai số nguyên được ghi cách nhau bởi dấu cách: đó là toạ độ x của đỉnh núi ($0 \le x \le 30000$) và độ cao của đỉnh núi y ($1 \le y \le 10000$).

Output: Ghi ra tệp văn bản MOUNT.OUT số lượng đỉnh núi nhìn thấy được.

Ví dụ:

MOUNT.INP	MOUNT.OUT
4 6 5 12 3 16 9 21 4	2



Hình 8.11

8.13. *Đài phun nước*

Để làm đẹp cảnh quan, Ban giám đốc một Công ti quyết định xây dựng ở sân tiền sảnh trụ sở công ti một đài phun nước. Đài phun nước phải có dạng một hình tròn với kích thước lớn nhất có thể được. Nhà thiết kế được biết là sân tiền sảnh của công ti có dạng một hình chữ nhật kích thước $X \times Y$. Tuy nhiên, khi lựa chọn vị trí cho đài phun nước nhà thiết kế gặp phải một vấn đề phức tạp: Trong sân tiền sảnh có N cột hình trụ tròn xoay không được phép di chuyển. Vì vậy vấn đề đặt ra cho nhà thiết kế là: Cần đặt đài phun nước ở vị trí nào để nó có bán kính lớn nhất có thể được đồng thời không được có diện tích chung khác 0 với các cột. Bạn hãy lập trình giúp nhà thiết kế giải quyết vấn đề trên.

Input: Vào từ tệp văn bản FONTAN.INP:

- Dòng đầu tiên chứa hai số thực X, Y, $1 \le X, Y \le 10^4$. Giả thiết rằng sân tiền sảnh là hình chữ nhật trên mặt phẳng toạ độ có toạ độ các đỉnh là $(0, 0)$, $(X, 0)$, (X, Y) và $(0, Y)$;
- Dòng thứ hai chứa số nguyên N $(0 \le N \le 10)$ là số lượng cột trong sân tiền sảnh;
- Dòng thứ i trong số N dòng tiếp theo chứa ba số thực X_i, Y_i và R_i cho biết toạ độ của tâm và bán kính của cột thứ i ($R_i \le X_i \le X - R_i$, $R_i \le Y_i \le Y - R_i$, $0.1 \le R_i \le \min\left(\dfrac{X}{2}, \dfrac{Y}{2}\right)$, $\sqrt{\left(X_i - X_j\right)^2 + \left(Y_i - Y_j\right)^2} \ge R_i + R_j$ với mọi $i \ne j$).

  Các số trên một dòng trong tệp dữ liệu được ghi cách nhau bởi dấu cách.

Output: Ghi ra tệp văn bản FONTAN.OUT ba số thực X_F, Y_F, R_F là toạ độ tâm và bán kính của đài phun nước.

Chú ý: Đài phun nước phải được đặt trong sân, được phép tiếp xúc với tường của sân hoặc cột, nhưng không được có diện tích chung khác không với các cột. Nếu có nhiều vị trí cùng cho bán kính lớn nhất chỉ cần đưa ra một trong số chúng.

Ví dụ:

FONTAN.INP	FONTAN.OUT
10 20 0	5.000 5.000 5.000

FONTAN.INP	FONTAN.OUT
20 20 4 2 2 2 18 2 2 2 18 2 18 18 2	10.000 10.000 9.314

FONTAN.INP	FONTAN.OUT
20 20 4 2 2 2 18 2 2 3 17 2 16 16 4	9.510 7.054 7.053

Chuyên đề 9

LÍ THUYẾT TRÒ CHƠI

I. Một số khái niệm

1. Vị trí

Là tập thông tin về trò chơi tại từng thời điểm. Những vị trí chứa đựng nhiều khả năng dẫn tới chiến thắng cho người nào đi tới được nó gọi là những vị trí có lợi (cho người chơi chiếm được lợi thế). Ngược lại có những vị trí chứa đựng nhiều khả năng dẫn tới thất bại cho người nào đi tới nó gọi là những vị trí không có lợi (người chơi bị vào thế bất lợi).

2. Luật chơi

Là những quy định cho phép người chơi thực hiện các phép biến đổi chuyển trò chơi từ một vị trí hiện tại tới một vị trí khác. Một bước đi hợp lệ là một phép biến đổi theo đúng luật chơi. Vị trí kết thúc là vị trí từ nó không thể di chuyển tiếp.

3. Trò chơi đối kháng

Là trò chơi hai người, khi một người chơi được lợi thế thì người chơi kia sẽ gặp bất lợi. Những trò chơi này chia làm hai loại: trò chơi có thông tin đầy đủ và trò chơi có thông tin không đầy đủ. Sự phân chia này dựa trên cơ sở người chơi có biết đầy đủ mọi thông tin về vị trí hiện tại của đối phương hay không. Chẳng hạn, các trò chơi với thông tin đầy đủ như cờ carô, cờ tướng, cờ quốc tế; các trò chơi với thông tin không đầy đủ như trò chơi đánh bài (người chơi che dấu những quân bài không cho đối thủ biết).

4. Trong mọi trò chơi, quá trình chơi có thể được biểu diễn dưới dạng cấu trúc cây có hướng (đôi khi là đồ thị có hướng) gọi là cây trò chơi. Nút của các cây này biểu diễn một vị trí trò chơi (nút gốc là *vị trí khởi đầu*, nút lá là *vị trí kết thúc*). Cung $(i; j)$ trên cây thể hiện một bước đi hợp lệ chuyển từ một vị trí i đến vị trí j kế tiếp.

II. Trò chơi tổ hợp cân bằng

1. Mô tả

Trò chơi tổ hợp cân bằng là trò chơi đối kháng thoả mãn những điều kiện sau:

- Có hai người chơi (người đi đầu kí hiệu là A, người kia là B);
- Có một tập hữu hạn các vị trí có thể xảy ra X (các trạng thái) của trò chơi.
- Có quy luật chơi Q. Quy luật chơi áp dụng cho hai người chơi là cân bằng nghĩa là mỗi người chơi đến lượt mình đều có quyền chọn một phép di chuyển hợp lệ tuỳ ý.
- Hai người chơi lần lượt, mỗi lần thực hiện một phép di chuyển hợp lệ.
- Trò chơi kết thúc khi đạt tới vị trí kết thúc. Thông thường quy định người di chuyển được cuối cùng là người thắng, người nào đến lượt mà không thể di chuyển được nữa thì thua (một số trò chơi cũng có thể quy định ngược lại).
- Nếu trò chơi không bao giờ kết thúc sẽ có một thông báo rút thăm. Có thể bổ sung thêm điều kiện nào đó để trò chơi kết thúc không cần thông báo rút thăm như: Trò chơi sẽ kết thúc khi đã có đủ một số lần di chuyển nhất định (kể cả trường hợp còn bước di chuyển tiếp, ...) mà không đấu thủ nào thắng thì hai đấu thủ là hoà.

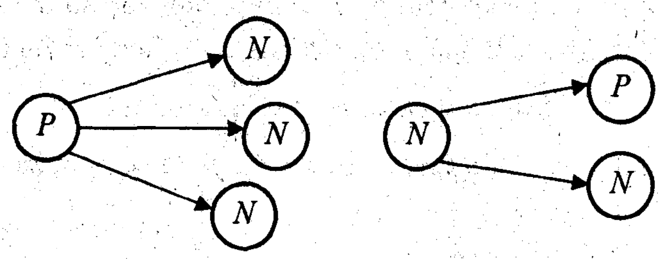
a) Tập P, tập N và cách tìm

Để thuận tiện cho việc xây dựng thuật toán (giành thắng) của trò chơi, người ta đưa ra khái niệm tập P và tập N. Đó là hai tập thoả mãn các tính chất sau:

(*a*) *Tất cả các vị trí kết thúc đều thuộc P.*

(*b*) *Từ mỗi vị trí thuộc N luôn có ít nhất một di chuyển tới vị trí thuộc P.*

(*c*) *Từ mỗi vị trí thuộc P, mọi di chuyển đều tới vị trí thuộc N.*



Hình 9.1. Tập P và tập N

Từ định nghĩa trên, dễ dàng suy ra thuật toán **giành** thắng như sau:

> *Thuật toán giành chiến thắng cho đấu thủ A khi **ban đầu A nhận vị trí thuộc** N là: đấu thủ **A luôn di chuyển tới các vị trí thuộc P** buộc đấu thủ B chỉ có thể đi tới vị trí thuộc N.*

Nếu ban đầu đấu thủ A bị nhận vị trí thuộc P thì A cần kéo dài trò chơi chờ đến một thời điểm nào đó trong quá trình chơi do B vô ý sao cho A có được vị trí thuộc N thì dẫn đến A luôn đi vào vị trí thuộc P (vậy trong trường hợp này A thắng phụ thuộc vào sự vô ý của B).

Ta có thể tìm tập các vị trí thuộc P và thuộc N bằng *đệ quy* như sau:

Bước 1. Gán $P = \varnothing$. Gán $N = \varnothing$. Nạp các vị trí kết thúc vào P.

Bước 2. Tìm các vị trí có thể di chuyển đến một vị trí nào đó thuộc tập P, nạp chúng vào N.

Bước 3. Tìm những vị trí *chỉ có thể* chuyển đến các vị trí thuộc N, nạp chúng vào P.

Bước 4. Dừng tìm kiếm.

b) Một số ví dụ

Ví dụ 1. Trò chơi *Làm rỗng và phân chia.*

Phát biểu: Có hai chiếc hộp, một hộp đựng m quân cờ, hộp còn lại đựng n quân cờ ($m, n \in \mathbb{N}^*$). Hai đấu thủ lần lượt chơi. Mỗi lần đến lượt chơi của mình, đấu thủ đó lấy hết quẩn cờ trong một hộp rồi đem các quân cờ trong hộp còn lại chia vào hai hộp sao cho mỗi hộp phải có ít nhất một quân. Rõ ràng trò chơi kết thúc khi mỗi hộp còn một quân. Đấu thủ nào được chơi lượt cuối sẽ giành chiến thắng. Hãy tìm các vị trí thuộc tập P, các vị trí thuộc tập N.

Phân tích. Trong trò chơi này, mỗi vị trí chơi là một cặp số thể hiện số quân cờ trong hai hộp. Vị trí khởi đầu là (m, n). Chúng ta sẽ chứng minh tập

$$\{(m', n') | \ m' \text{ và } n' \text{ là số nguyên lẻ}\}$$

là tập các vị trí thuộc P và tập $\{(m', n') | \ m' \text{ hoặc } n' \text{ là số nguyên chẵn}\}$ là tập các vị trí thuộc N. Thật vậy:

Trước hết, thấy vị trí kết thúc là $(1, 1) \in P$.

Bây giờ giả sử $(m', n') \in N$ thì hoặc m' chẵn hoặc n' chẵn. Không mất tính chất tổng quát, giả sử m' chẵn. Từ vị trí này, chúng ta chọn phép chuyển là: làm rỗng hộp thứ hai (chứa n') và chia m' quân có trong hộp thứ nhất thành hai phần với số quân cờ trong cả hai phần đều lẻ cho vào hai hộp, đi tới vị trí $(s, m' - s)$ với s' là số lẻ $(1 \le s \le m - 1)$, nghĩa là $(s, m' - s) \in P$. Vậy từ một vị trí thuộc N, luôn tìm được ít nhất một phép chuyển tới vị trí thuộc P.

Tiếp theo, giả sử $(m', n') \in P$ nghĩa là m' và n' đều lẻ. Nếu làm rỗng hộp thứ nhất (chứa m') và lấy n' quân của hộp thứ hai chia vào hai hộp và chuyển tới vị trí $(s, n' - s)$ với $1 \le s \le n - 1$, thì s và $n' - s$ không thể cùng lẻ (do tổng của chúng bằng n' là số lẻ), do đó $(s, n' - s) \in N$. Tình trạng tương tự khi làm rỗng hộp thứ hai (chứa n'). Vậy mọi phép chuyển từ vị trí thuộc P đều chuyển tới vị trí thuộc N.

Ví dụ 2. Trò chơi *Lấy bớt quân cờ.*

Phát biểu. Trên bàn có C quân cờ. Có hai đấu thủ chơi lần lượt. Mỗi lần, người chơi đến lượt sẽ lấy ra khỏi bàn từ 1 đến M quân cờ. Người thắng là người lấy được quân cờ cuối cùng. Chương trình đọc hai số nguyên dương C và M từ tệp văn bản *vidu2.txt* (C trong khoảng từ 1 đến 10^4, $1 \le M \le C$), lập trình với giao diện trên màn hình để người chơi với máy (ưu tiên cho *người đi trước*) với thuật toán giành thắng cho máy.

Phân tích. Chúng ta dùng phương pháp phân tích ngược, xét từ vị trí kết thúc đến vị trí khởi đầu. Vị trí kết thúc là vị trí ứng với số quân cờ bằng 0, đây là vị trí thuộc P. Nếu trên bàn còn 1 hoặc 2... hoặc M quân cờ thì đến lượt đấu thủ nào lấy hết số quân cờ trên bàn cờ thì sẽ tới vị trí kết thúc và thắng cuộc; vậy các vị trí này thuộc N. Nếu còn lại $M + 1$ quân cờ, người đến lượt dù lấy kiểu nào cũng phải để lại trên bàn số quân cờ là 1 hoặc 2 ... hoặc M, nghĩa là từ vị trí $M + 1$ quân cờ chỉ có thể đi tới các vị trí thuộc N, vậy vị trí $M + 1$ quân cờ là vị trí thuộc P. Bằng cách lí giải tương tự, đi đến kết luận chung là: các vị trí gồm 0, $M + 1$, $2(M + 1)$, $3(M + 1)$ quân cờ, ... là các vị trí thuộc P, các vị trí còn lại (phần dư của phép chia số quân cờ hiện tại cho $M + 1$ là khác 0) thuộc tập N.

Chiến thuật giành chiến thắng. Nếu số quân cờ C ban đầu không chia hết cho $M + 1$ (nghĩa là đấu thủ A ban đầu nhận vị trí thuộc tập N), thì A thắng: mỗi lần A cần lấy sao cho luôn để lại trên bàn số quân chia hết cho $M + 1$ (nghĩa là A

luôn buộc đấu thủ *B* phải nhận vị trí thuộc *P* để *B* chỉ có thể đi tiếp đến vị trí thuộc *N*). Ngược lại, nếu số quân *C* ban đầu chia hết cho *M* + 1 (*A* ban đầu ở vị trí thuộc *P*) thì *A* chỉ nên lấy 1 quân cờ để kéo dài trò chơi, đợi vô ý (nếu có) của *B* sao cho *A* vào được vị trí thuộc *N*, khi đó *A* phải di chuyển tới vị trí thuộc *P*.

LAYCO.PAS ✓ Trò chơi *Lấy bớt quân cờ* với thuật giành thắng cho máy

```
uses crt;
const fi = 'vidu2.txt';
var c, {số quân ban đầu}
 m, {bước đi cực đại}
 move, {số quân được chuyển của một lượt}
 m1 : integer; {m+1}
 OK : boolean;
procedure nhap;
var f : text;
begin
 assign(f, fi); reset(f);
 read(f, c, m); m1 := m+1;
 writeln('Tong so quan:',c,'.Quan chuyen toi da la:',m);
 close(f);
end;
procedure thuatthang;
begin
 move := c mod m1;
 dec(c, move);
 writeln('May chuyen so quan:',move,' so quan con:', c);
 OK := not OK;
end;
procedure keodai;
begin
 dec(c,1);
 writeln('May chuyen so quan: 1, so quan con: ', c);
 OK := not Ok;
end;
procedure nguoidi;
begin
 write('Ban chuyen so quan: '); readln(move);
 dec(c, move);
 writeln('So quan con lai la: ', c);
 OK := not OK;
end;
```

```
BEGIN
 clrscr;
 nhap;
 OK := True;
 while c>0 do
  begin
   nguoidi;
   if c=0 then break;
   if c mod m1 <>0 then thuatthang else keodai;
  end;
 if OK then writeln('May thang ')
 else writeln('Ban thang ');
 readln;
END.
```

Sau đây là thuật toán lập trình trò chơi đối kháng giành thắng cho máy khi tiến hành trò chơi giữa người và máy tính:

1. Thể hiện giao diện chọn đấu thủ đi trước;

   {biến OK = True là người đi trước; OK = False là máy đi trước}

2. OK := False;
3. Tạo vị trí ban đầu;
4. Thể hiện vị trí ban đầu;
5. **While** (*chưa là vị trí kết thúc*) **do**

   begin

   If OK **then** *Người đi;*

   If (*kết thúc*) **then** *Thoát khỏi while;*

   If (*là vị trí thuộc P*) **then** *Máy đi kéo dài trò chơi*

   else *Máy đi theo thuật thắng;*

   End;

6. **If** OK **then** *thông tin Người thắng* **else** *thông tin Máy thắng;*

Trong mỗi chương trình con *Người đi; Máy đi kéo dài trò chơi, Máy đi theo thuật thắng* đều có chung nội dung là: Thể hiện bước đi và vị trí sau khi đi, đổi giá trị biến OK. Riêng trong chương trình con *Máy đi theo thuật thắng* còn tuỳ thuộc trò chơi, đầu tiên cần tìm bước đi giành thắng cho thích hợp.

Ví dụ 3. Trò chơi *Trừ dần.*

Phát biểu. Cho tập S gồm hữu hạn các số nguyên dương, gọi là *tập trừ*. Trên một cọc có N quân, hai đấu thủ lần lượt chơi, mỗi lần chuyển đi s quân khỏi cọc ($s \in S$). Đấu thủ nào được chuyển lần cuối cùng thì thắng. Viết chương trình thông báo cho biết đấu thủ đi đầu có chắc thắng không?

Input: Tệp *vidu3.inp* gồm hai dòng:

- Dòng thứ nhất ghi số nguyên dương N;
- Dòng thứ hai ghi các số nguyên dương trong tập trừ S (sắp tăng dần).

Output: Tệp *vidu3.out* như sau: Nếu đấu thủ đi đầu chắc thắng thì ghi số 1, nếu không chắc thắng thì ghi số 0.

Phân tích. Xét trò chơi trừ dần với tập trừ là $S = \{1, 3, 4\}$:

Trong trò chơi này, mỗi vị trí của trò chơi là một số quân còn trên cọc tại từng thời điểm. Chúng ta tìm tập P của trò chơi này: Có đúng một vị trí kết thúc là 0 đó là vị trí P. Do đó các vị trí 1, 3, 4 sẽ thuộc N vì từ chúng có thể chuyển tới 0. Nhưng vị trí 2 thuộc P vì từ 2 có duy nhất một phép chuyển đến 1 (là vị trí của N). Còn 5 và 6 phải thuộc N vì nó có thể chuyển đến 2... Chúng ta sẽ mở rộng kết luận bằng quy nạp và tìm được $P = \{0, 2, 7, 9, 14, 16,...\}$ là tập các số nguyên không âm khi chia cho 7 còn dư 0 hoặc 2. Còn N chứa các số còn lại $N = \{1, 3, 4, 5, 6, 8, 10, 11, 12, 13, 15,...\}$

Nhận xét. Khi lập trình, để tìm được các vị trí P và N trong trò chơi này chúng ta có thể dùng phương pháp đánh dấu trên mảng một chiều A.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
	P	N	P	N	N	N	N	P	N	P	N	N	N	N	P	N	P	...
$A[i]$	1	0	1	0	0	0	0	1	0	1	0	0	0	0	1	0	1	...

Với vị trí i thuộc tập P thì ghi nhận $A[i] = 1$, nếu i thuộc tập N thì ghi nhận $A[i] = 0$. Ngoài ra các số thuộc tập trừ S được lưu vào mảng một chiều S (giả sử có m số thuộc tập trừ). Ban đầu khởi trị $A[0] = 1$; sau đó xét các vị trí i từ 1 đến n. Với mỗi vị trí i, chúng ta duyệt tất cả các vị trí có thể đi tiếp theo từ i, đó là các vị trí $i - S[j]$ (j từ 1 đến m), nếu thấy tất cả các vị trí $i - S[j]$ đều thuộc tập N (nghĩa là $A[i - s[j]] = 0$) thì vị trí i chính là vị trí thuộc tập P. Ngược lại nếu có một vị trí trong các vị trí $i - S[j]$ là vị trí thuộc P thì kết luận ngay vị trí i là vị trí thuộc N.

TRUDAN.PAS ✓ Trò chơi *Trừ dần* với thuật thắng cho đấu thủ đi trước

```
const fi = 'vidu3.inp';
 fo = 'vidu3.out';
 max = 100000;
var n,m : longint;
 a : array[0..max] of longint;
 s : array[0..max] of longint;
 f,g : text;
 i : longint;
function la_P(i : longint): boolean;
var j: longint;
begin
 la_P := false;
 for j:=1 to m do
 if i-s[j]>=0 then
 if a[i-s[j]]=1 then exit;
 la_P := true;
end;
BEGIN
 assign(f,fi); reset(f);
 readln(f,n);
 m := 0;
 while not eoln(f) do
  begin
   inc(m);
   read(f,s[m]);
  end;
 close(f);
 for i:=0 to n do a[i] := 0;
 a[0] := 1;
 for i:=1 to n do
 if a[i]=0 then
 if la_P(i) then a[i]:=1;
 assign(g,fo);
 rewrite(g);
 if a[n]=1 then write(g,0) else write(g,1);
 close(g);
END.
```

2. *Tổng Nim và trò chơi Nim*

Để có thuật chiến thắng trong những trò chơi đối kháng ta cần xác định được tập *P* và tập *N*. Công việc này có thể được giải quyết dễ dàng hơn nhờ khái niệm tổng Nim.

a) Tổng Nim

Tổng Nim của hai số nguyên không âm là kết quả phép cộng *không nhớ* của hai số đó trong hệ cơ số 2 (còn gọi là cộng theo môđun 2).

Ví dụ 1

Số 14 hệ thập phân viết dưới dạng cơ số 2 là **1110**$_{(2)}$

Số 55 hệ thập phân viết dưới dạng cơ số 2 là **110111**$_{(2)}$

Tổng Nim của chúng là: **111001**$_{(2)}$ (là 57 trong hệ thập phân)

Vậy $14 \oplus 55 = 57$. Thường kí hiệu phép toán cộng theo môđun 2 là $\oplus$.

Ví dụ 2

Số 3 hệ thập phân viết dưới dạng cơ số 2 là **11**$_{(2)}$

Số 5 hệ thập phân viết dưới dạng cơ số 2 là **101**$_{(2)}$

Số 8 hệ thập phân viết dưới dạng cơ số 2 là **1000**$_{(2)}$

Tổng Nim của chúng là: **1110**$_{(2)}$ (là 14 trong hệ thập phân)

Vậy $3 \oplus 5 \oplus 8 = 14$.

Trong ngôn ngữ lập trình Pascal kí hiệu phép toán $\oplus$ (tính tổng Nim) là *xor*, trong C/C++ là ^.

Phép toán $\oplus$ có tính chất kết hợp, giao hoán. Đặc biệt:

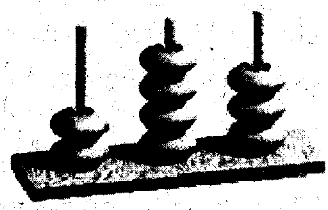
$$0 \oplus a = a,\ a \oplus a = 0, \text{ do đó nếu } a \oplus b = a \oplus c \text{ thì } b = c.$$

Có thể tạo ra tổng Nim của hai số nguyên không âm x và y bằng cách lập bảng như sau: Tại ô giao điểm của cột x và dòng y ghi $x \oplus y$; đó là số nguyên không âm nhỏ nhất chưa có trong tập các số đã có tại các ô (x', y) với $1 \le x' < x$ và các số đã có tại các ô (x, y') với $1 \le y' < y$. Chú ý $0 \oplus y = y$ và $x \oplus 0 = x$.

y \ x	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	3	2	5	4	7	6	9	8	11	10	13
2	2	3	0	1	6	7	4	5	10	11	8	9	14
3	3	2	1	0	7	6	5	4	11	10	9	8	15
4	4	5	6	7	0	1	2	3	12	13	14	15	8
5	5	4	7	6	1	0	3	2	13	12	15	14	9
6	6	7	4	5	2	3	0	1	14	15	12	13	10
7	7	6	5	4	3	2	1	0	15	14	13	12	11
8	8	9	10	11	12	13	14	15	0	1	2	3	4
9	9	8	11	10	13	12	15	14	1	0	3	2	5
10	10	11	8	9	14	15	12	13	2	3	0	1	6
11	11	10	9	8	15	14	13	12	3	2	1	0	7
12	12	13	14	15	8	9	10	11	4	5	6	7	0

b) Trò chơi Nim chuẩn

Trò chơi Nim chuẩn như sau: Có ba cọc lần lượt chứa x_1, x_2, x_3 quân. Hai người chơi lần lượt tạo ra các bước di chuyển quân. Mỗi bước di chuyển là: chọn một cọc tuỳ ý còn quân và lấy bớt một số quân ở cọc này (từ một quân cho đến toàn bộ quân). Người chiến thắng là người lấy được quân cuối cùng. Trường hợp tổng quát có số cọc tuỳ ý.



Hình 9.2.
Trò chơi Nim chuẩn

c) Định lí Bouton

> *Mỗi vị trí (x_1, x_2, x_3) trong trò chơi Nim ba cọc là vị trí P khi và chỉ khi tổng Nim $x_1 \oplus x_2 \oplus x_3 = 0$.*

Ví dụ. Vị trí (5, 7, 9) trong Nim ba cọc ứng với tổng Nim $5 \oplus 7 \oplus 9 = 11 > 0$ nên nó là vị trí N. Có thể kiểm tra thấy vị trí (4, 12, 8) là vị trí P.

Định lí còn áp dụng với trò chơi Nim có số cọc nhiều hơn.

Từ định lí suy ra chiến thuật giành thắng trong trò chơi Nim chuẩn như sau:

Giả sử vị trí hiện tại là $(x_1, x_2, ..., x_n)$ tương ứng với tổng Nim là

$$g = x_1 \oplus x_2 \oplus ... \oplus x_n > 0.$$

Có thể chứng minh sẽ tồn tại thành phần x_i mà $x_i' = g \oplus x_i \le x_i$. Cách đi để giành chiến thắng là giảm cọc x_i thành x_i'.

Vị trí mới $(x_1, x_2, ..., x_{i-1}, x_i', x_{i+1}, ..., x_n)$ có tổng Nim là $g' = 0$ vì:

$$g' = x_1 \oplus x_2 \oplus ... \oplus x_{i-1} \oplus x_i' \oplus x_{i+1} \oplus ... \oplus x_n$$

$$= x_1 \oplus x_2 \oplus ... \oplus x_{i-1} \oplus [g \oplus x_i] \oplus x_{i+1} \oplus ... \oplus x_n$$

$$= (x_1 \oplus x_2 \oplus ... \oplus x_{i-1} \oplus x_i \oplus x_{i+1} ... \oplus x_n) \oplus g = g \oplus g = 0.$$

Ngược lại cũng có thể chứng minh: nếu vị trí hiện tại là $(x_1, x_2, ..., x_n)$ tương ứng với tổng Nim là $g = x_1 \oplus x_2 \oplus ... \oplus x_n = 0$ thì sau phép di chuyển bất kì đều dẫn tới vị trí mới có tổng Nim dương.

Do đó nếu vị trí ban đầu $(x_1, x_2, ..., x_n)$ có tổng Nim dương thì người A luôn thắng. Trong trường hợp nếu tổng Nim bằng 0 (vị trí bất lợi) thì người A chơi cầm chừng kéo dài trò chơi chẳng hạn mỗi lần giảm đi 1 (hoặc số ngẫu nhiên) ở một cột bất kì để chờ sai lầm của đối thủ B khiến A nhận được vị trí có tổng Nim dương.

3. *Trò chơi trên đồ thị*

Một số trò chơi có thể mô tả bằng đồ thị có hướng. Mỗi vị trí của trò chơi được coi như một đỉnh của đồ thị và mỗi phép di chuyển hợp lệ được xem như một cung dẫn từ đỉnh này sang đỉnh khác. Chúng ta sẽ định nghĩa hàm Sprague-Grundy (SG) để xác định được các vị trí thuộc tập P hoặc N của trò chơi trên đồ thị.

a) Đồ thị trò chơi

Đồ thị của một trò chơi là đồ thị có hướng $G = (X, F)$ với X là tập đỉnh không rỗng - đó là tập các vị trí của trò chơi và F là hàm trên tập X sao cho với mỗi $x \in X$ thì $F(x)$ là một tập con thuộc X tức là $x \in X \to F(x) \subset X$. Các đỉnh thuộc $F(x)$ được gọi là những đỉnh có thể tới từ đỉnh x. Nếu $F(x)$ rỗng thì x là đỉnh kết thúc. Một phép di chuyển hợp lệ là đi từ đỉnh x tới một đỉnh thuộc $F(x)$.

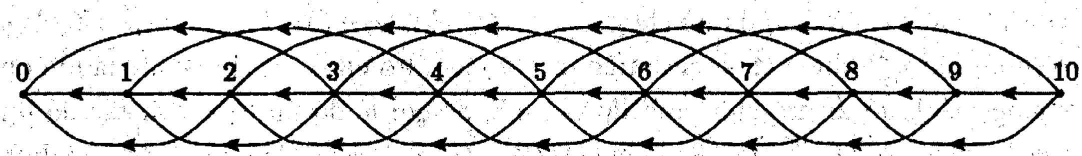
Trò chơi tổ hợp cân bằng có hai người chơi được mô phỏng trên đồ thị $G = (X, F)$ với đỉnh xuất phát $x_0 \in X$ và tuân theo các quy tắc sau:

(1) Người chơi thứ nhất đi trước và bắt đầu từ đỉnh x_0;

(2) Hai người chơi lần lượt thực hiện phép di chuyển trên đồ thị;

(3) Tại vị trí x, người chơi được di chuyển đến bất kì $y \in F(x)$;

(4) Người chơi nào đến lượt phải nhận đỉnh kết thúc là người thua.



Hình 9.3

Ví dụ trò chơi trừ số với tập trừ là {1, 2, 3} bắt đầu với cọc có $n = 10$, có thể mô tả như một đồ thị có hướng (X, F) như sau:

Tập $X = \{0, 1, \ldots, 10\}$. Hàm F có các giá trị cụ thể là:

$$F(0) = \phi, F(1) = \{0\}, F(2) = \{0,1\} \text{ và } F(x) = \{x-1, x-2, x-3 \mid n \geq x > 2\}.$$

b) Hàm Sprague-Grundy

Hàm Sprague-Grundy (SG) của một đồ thị $G = (X, F)$ được định nghĩa như sau: mỗi đỉnh x của đồ thị được gắn một số $g(x)$ là số nguyên không âm nhỏ nhất không nằm trong tập các giá trị $g(y)$ của các đỉnh y tới được từ x: $g(x) = \min\{n \geq 0 : \forall y \in F(x), n \neq g(y)\}$. Đỉnh x nếu là đỉnh kết thúc thì $g(x) = 0$. Đôi khi còn dùng kí hiệu: $g(x) = \max\{g(y) | \forall y \in F(x)\}$.

Khi trên đồ thị trò chơi xác định được hàm Sprague-Grundy chúng ta sẽ xác định đỉnh x là bất lợi nếu $g(x) = 0$ và là đỉnh có lợi nếu $g(x) > 0$. Thật vậy, vì:

- Người nhận đỉnh kết thúc (có $g(x) = 0$) thì không có phép di chuyển tiếp theo nữa nên thua (thường quy định là thua).
- Nếu nhận đỉnh x có $g(x) = 0$ và không là đỉnh kết thúc thì mọi cách đi đều đến đỉnh y có $g(y) > 0$ chưa là đỉnh kết thúc; nên người tiếp theo vẫn còn đi tiếp được. Vậy nhận đỉnh x có $g(x) = 0$ và không là đỉnh kết thúc thì đối thủ chưa thua.
- Nếu nhận đỉnh $g(x) > 0$ thì luôn tồn tại một đỉnh $y \in F(x)$ mà $g(y) = 0$ buộc đối thủ phải nhận, cứ tiếp tục như thế cuối cùng buộc đối thủ phải nhận đỉnh kết thúc và thua.

Ví dụ. Trong trò chơi *Trừ dần* với tập trừ là $S = \{1, 2, 3, ..., a\}$ thì đồ thị trò chơi có hàm Sprague-Grundy là $g(x) = x \bmod (a + 1)$.

c) Tổng trò chơi

Giả sử chúng ta có n đồ thị trò chơi là $G_1 = (X_1, F_1)$, $G_2 = (X_2, F_2)$, …, $G_n = (X_n, F_n)$. Nếu mỗi lượt đi được chọn chơi trên một trong n trò chơi này thì chúng ta tổ hợp chúng thành đồ thị trò chơi mới là: $G = (X, F)$ gọi là đồ thị tổng của các đồ thị thành phần: $G_1, G_2, \ldots, G_n$. Kí hiệu $G = G_1 + G_2 + \ldots + G_n$. Đồ thị G có: Tập đỉnh X là một tích Đề-các $X = X_1 \times X_2 \times \ldots \times X_n$ gồm tất cả các bộ $x = (x_1, x_2, \ldots, x_n)$ với $x_i \in X_i$, $i = 1, 2, \ldots, n$. Với mỗi đỉnh $x \in X$, tập các đỉnh theo sau x là:

$$F(x) = F(x_1, x_2, \ldots, x_n)$$

$$= F_1(x_1) \times \{x_2\} \times \ldots \times \{x_n\} \cup \{x_1\} \times F_2(x_2) \times \ldots \times \{x_n\} \cup \ldots \cup \{x_1\} \times \{x_2\} \times \ldots \times F_n(x_n)$$

Do một phép di chuyển từ x chứa đúng một phép di chuyển từ x_i tới một điểm thuộc $F_i(x_i)$ nên có thể viết: $F(\mathrm{x}) = \{(x_1, ..., x_{i-1}, x'_i, x_{i+1}, ..., x_n) : x'_i \in F(x_i)\}$.

Ví dụ. Trò chơi Nim ba cọc, có thể coi như tổng của ba trò chơi Nim một cọc.

Chúng ta cần một phương pháp tính hàm Sprague-Grundy trên đồ thị tổng các trò chơi khi biết hàm Sprague-Grundy trên từng đồ thị trò chơi thành phần.

d) Định lí Sprague-Grundy

> *Nếu g_i là các hàm Sprague-Grundy trên đồ thị của trò chơi thành phần G_i, với $i = 1, 2, ..., n$, thì đồ thị tổng của chúng là*
>
> $$G = G_1 + G_2 + \ldots + G_n$$
>
> *sẽ có hàm SG là:* $g(x_1, x_2, ..., x_n) = g_1(x_1) \oplus g_2(x_2) \oplus \ldots \oplus g_n(x_n)$.

Ví dụ 1. Kí hiệu $G(m)$ là trò chơi *trừ dần một cọc* với tập trừ là $S_m = \{1, 2, \ldots, m\}$ có thể trừ bớt từ 1 đến m quân trên cọc, ta kí hiệu hàm Sprague-Grundy của trò chơi này là $g_m(x)$ thì $g_m(x) = x \bmod (m + 1)$ và $0 \le g_m(x) \le m$.

Cho ba trò chơi trừ dần trên một cọc cụ thể là: trò chơi $G(3)$ với số quân trên một cọc là 9; trò chơi $G(5)$ với số quân trên một cọc là 10 và trò chơi $G(7)$với số quân trên một cọc là 14.

Xét trò chơi tổng $G = G(3) + G(5) + G(7)$ có vị trí khởi đầu là (9, 10, 14). Hàm Sprague-Grundy của vị trí khởi đầu là:

$$g(9, 10, 14) = g_3(9) \oplus g_5(10) \oplus g_7(14) = 1 \oplus 4 \oplus 6 = 3.$$

Một phép chuyển tối ưu là: trong trò chơi $G(7)$ chuyển đến vị trí 13 vì $g_7(13) = 5$ nên $g(9, 10, 13) = g_3(9) \oplus g_5(10) \oplus g_7(13) = 1 \oplus 4 \oplus 5 = 0$.

Sau đây là một số ví dụ về những trò chơi có tên là *Lấy và phân chia* là những trò chơi trên các cọc mà phép di chuyển là lấy bớt quân khỏi cọc hoặc phân chia một cọc thành nhiều cọc khác rỗng theo những điều kiện nào đó.

Ví dụ 2. Trò chơi *Kayles*

Phát biểu. Hai người chơi trò lăn bóng (bowing). Trước hai người là một hàng quả bowing mà quả thứ hai đã bị đánh đổ. Giả sử cả hai người chơi thành thạo đến mức có thể lăn đổ bất kì một quả bowing nào hoặc lăn đổ hai quả bowing bất kì cạnh nhau. Hai người lần lượt lăn bóng. Quy định người lăn đổ quả bowing cuối cùng sẽ thắng.

Phân tích. Khi đánh đổ một hoặc hai quả bowing ở các đầu hàng chính là phép lấy đi khỏi cọc (hàng) một hoặc hai quân. Khi đánh đổ một hoặc hai quân (ở giữa hàng) chia cọc thành hai thì đó là phép phân chia cọc.

Chúng ta hãy tìm hàm Sprague-Grundy cho trò chơi này. Chỉ có một vị trí kết thúc là cọc rỗng, vậy $g(0) = 0$.

x	Các đỉnh y kế tiếp x	Các giá trị $g(y)$	$g(x)$
0			0
1	0	0	1
2	1, 0	1, 0	2
3	2, 1, (1, 1)	$2, 1, 1 \oplus 1 = 0$	3
4	3, 2, (1, 2), (1, 1)	$3, 2, 1 \oplus 2 = 3, 1 \oplus 1 = 0$	1
5	4, 3, (3, 1), (2, 2), (1, 2)	$1, 3, 3 \oplus 1 = 2, 2 \oplus 2 = 0, 1 \oplus 2 = 3$	4

Khi $x \geq 72$, các giá trị $g(x)$ bắt đầu tuần hoàn với chu kì 12, dãy giá trị $g(x)$ sau được lặp mãi:

x	...	72	73	74	75	76	77	78	79	80	81	82	83	...
$g(x)$	...	4	1	2	8	1	4	7	2	1	8	2	7	...

Ví dụ 3. Trò chơi *Treblecross*

Phát biểu. Trên một hàng có n ô vuông để trống. Hai người chơi lần lượt ghi X vào một ô trống không kề với ô đã có X. Người ghi chữ X cuối cùng là người thắng. Tính hàm Sprague- Grundy của trò chơi này.

Phân tích. Trò chơi này có thể mô tả như trò chơi trên một cọc mà có thể bỏ bớt các quân trên cọc hoặc chia cọc thành hai cọc.

Nếu $n = 1$ có đúng một phép chuyển tới $n = 0$.

Với $n > 1$. Một phép ghi X vào ô ở một trong hai đầu dòng là tương ứng loại trừ hai quân ở đầu đó khỏi cọc. Đặt X tại ô cách ô ở một trong hai đầu với khoảng cách là một ô thì tương đương loại trừ ba quân ở đầu đó khỏi cọc. Đặt X tại ô không phải ô ở hai đầu (và không phải tại ô kề ô X đã có) tương đương loại trừ ba quân ở đầu đó khỏi cọc và chia cọc thành hai cọc mới. Vậy quy luật di chuyển trong trò chơi này là:

(1) Nếu cọc chỉ có một quân thì loại bỏ quân này;

(2) Có thể loại bỏ hai quân;

(3) Có thể loại bỏ ba quân;

(4) Chia cọc đó thành hai cọc có tổng số quân giảm đi 3.

Giả sử đã tính được giá trị hàm Sprague-Grundy của các cọc 0, 1, 2, ..., 9. Bây giờ cần tính $g(11)$. Với quy luật trên, từ cọc 11 sẽ đi tới cọc 9, cọc 8, hai cọc (1, 7), hai cọc (2, 6), hai cọc (3, 5) và hai cọc (4, 4). Cọc 9 cũng có thể coi như hai cọc (−1, 9), cọc 8 coi như hai cọc (0, 8) và quy ước thêm $g(-1) = 0$ thì ta có thể dễ viết công thức tính $g(11)$ là số nguyên không âm nhỏ nhất chưa có trong các số $g(a) \oplus g(b)$ mà $a + b = 11 - 3 = 8$.

a	−1	0	1	2	3	4	5	6	7	8	9
$g(a)$	0	0	1	1	2	0	3	1	1	0	3
$g(b)$	3	0	1	1	3	0	2	1	1		
b	9	8	7	6	5	4	3	2	1		

Vậy các giá trị mà $g(a) \oplus g(b)$ (với $a + b = 8$) là: 3, 0, 0, 0, 1, 0, 1, 0, 0 nên $g(11) = 2$.

💻 TREBLECROSS.PAS ✓ Tính giá trị hàm Sprague- Grundy

```
uses crt;
var g,c : array[-1..1000] of integer;
 n,k : integer;
procedure tinh(m: integer);
var i, j, k : integer;
begin
 i:=-1; j:= m-i-3;
 while i<=j do
  begin
   c[g[i] xor g[j]] := 1;
   inc(i); dec(j);
  end;
 i:= 0;
 while c[i]=1 do inc(i);
 g[m] := i;
end;
BEGIN
 clrscr; n := 100;
 g[-1] := 0; write(g[-1]:4); g[0] := 0; write(g[0]:4);
 g[1] := 1; write(g[1]:4); g[2] := 1; write(g[2]:4);
 for k:=3 to n do
  begin
   fillchar(c, sizeof(c),0);
   tinh(k); write(g[k]:4);
  end;
 readln;
END.
```

Ví dụ 4. Xét trò chơi Nim một cọc với quy luật chuyển ít nhất là một quân và nhiều nhất là nửa số quân trên cọc (*At-Most-Half*). Viết chương trình xây dựng hàm Sprague-Grundy với trò chơi này.

Input. Tệp *vidu4.inp* gồm 10 dòng mỗi dòng là một giá trị của n ($1 \le n \le 10000$).

Output. Tệp *vidu4.out* gồm 10 dòng cho biết giá trị hàm SG tại vị trí ban đầu n của dòng tương ứng trong tệp *vidu4.inp*.

💻 ATMOSTHALF.PAS ✓ Tính giá trị hàm Sprague- Grundy

```
uses crt;
const max = 10000;
f1 = 'vidu3.inp'; f2 = 'vidu3.out';
```

```
type mang = array[0..max] of longint;
var v,g : mang;
n : longint;
fi,fo : text;
procedure sg;
var i,j,k : longint;
begin
for i:=0 to max do g[i] := 0;
g[0] := 0;
for i:=2 to max do
begin
for j:=0 to max do v[j] := 0;
for k:=1 to (i div 2) do v[g[i-k]] := 1;
k:= 0;
while v[k]=1 do inc(k);
g[i] := k;
end;
end;
BEGIN
sg;
assign(fi, f1); reset(fi);
assign(fo, f2); rewrite(fo);
while not eof(fi) do
begin
readln(fi, n);
writeln(fo, g[n]);
end;
close(fi); close(fo);
END.
```

4. *Các trò chơi lật xu*

a) Lật xu một chiều

Phát biểu. Cho một hàng gồm hữu hạn đồng xu đặt ngửa hoặc sấp. Một lượt đi là một thao tác đổi trạng thái ngửa thành sấp hoặc ngược lại của toàn bộ các đồng xu thuộc một tập các đồng xu thoả mãn các luật quy định của từng trò chơi. Trò chơi kết thúc khi các đồng xu đều sấp. Ai được thực hiện bước lật cuối cùng thì người đó chiến thắng. Trò chơi lật rùa (Turning Turtles) cũng là một dạng của trò chơi lật xu.

Chú ý rằng một trạng thái với k đồng xu ngửa tại vị trí $x_1,...,x_k$ có thể coi là tổng của k trò chơi j ($j=\overline{1..k}$). Mỗi trò chơi j được mô tả là một dãy x_j đồng xu trong đó các đồng xu từ 1 đến x_{j-1} đặt sấp và đồng xu thứ x_j đặt ngửa. Ví dụ với trạng thái *SNNSSN* (*S* kí hiệu cho sấp và *N* kí hiệu cho ngửa) là tổng của ba trò chơi *SN*, *SSN* và *SSSSSN*. Vì thế:

$$g(SNNSSN) = g(SN) \oplus g(SSN) \oplus g(SSSSSN).$$

Ví dụ 1. Trò chơi *Twins*

Quy luật chơi là phải lật đúng hai đồng xu trong đó đồng xu bên phải đang ngửa bị lật thành sấp, đồng xu thứ hai là một trong các đồng xu bên trái của đồng xu thứ nhất và phải lật thành ngửa. Nếu ta đánh số các đồng xu từ 1 thì hàm $g(x) = x$ vì từ vị trí x có thể đi đến các vị trí từ 0 đến $x-1$ (vị trí 0 quy ước là trạng thái các đồng xu đều sấp).

Ví dụ 2. Trò chơi *Mock Turtles*

Phát biểu. Có một hàng đồng xu, phép di chuyển là lật một nhóm tối đa ba đồng xu với điều kiện đồng xu bên phải nhất của nhóm đang ngửa. Khi mọi đồng xu đều sấp là kết thúc.

Phân tích. Vị trí kết thúc là mọi đồng xu đều sấp. Kí hiệu vị trí này là -1, thì $g(-1) = 0$. Để thuận tiện cho tính toán ta đánh dấu các đồng xu từ 0. Với đồng xu ngửa tại 0 thì rõ ràng với $g(0) = 1$ vì lật sấp nó sẽ đến vị trí kết thúc. Đồng xu ngửa tại vị trí 1 có thể đi đến vị trí ngửa tại 0 (úp xu 1, lật ngửa xu 0) hoặc vị trí kết thúc (úp xu 1) nên giá trị $g(1) = 2$. Vị trí ngửa 2 có thể đi đến vị trí ngửa 1, hoặc vị trí ngửa 0, hoặc các vị trí có 2 xu ngửa tại 0 và 1 ($g(0, 1) = g(0) \oplus g(1) = 3$) nên $g(2) = 4$. Tương tự ta có được bảng giá trị:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
$g(x)$	1	2	4	7	8	11	13	14	16	19	21	22	25	26	28	...

Khi nhìn vào bảng trên, dự đoán giá trị $g(x)$ thấy $g(x) = 2x$ hoặc $2x+1$. Người ta phát hiện rằng giá trị đó phụ thuộc vào số lượng số 1 trong khai triển nhị phân của $2x$. Một số tự nhiên được gọi là bit lẻ nếu số bit 1 trong khai triển nhị phân là lẻ và gọi là bit chẵn trong trường hợp ngược lại. Chẳng hạn, số 1, 2, 4, 7 là bit lẻ do khai triển nhị phân tương ứng là 1, 10, 100, và 111. Các giá trị 0, 3, 5 là bit chẵn vì khai triển nhị phân tương ứng là 0, 11, 101. Có thể chứng minh $g(x) = 2x$ nếu $2x$ là bit lẻ và $g(x) = 2x + 1$ trong trường hợp $2x$ là bit chẵn.

Ví dụ 3. Trò chơi *Ruler*

Cho một hàng gồm n đồng xu. Một lượt đi có thể lật một nhóm gồm số đồng xu liên tiếp nếu đồng xu bên phải nhất của nhóm đang ngửa. Nếu chúng ta đánh số các đồng xu từ 1, dễ thấy hàm SG là:

$$g(n) = \max\{0, g(n-1), g(n-1) \oplus g(n-2), ..., g(n-1) \oplus g(n-2)... \oplus g(1)\}.$$

Từ đó có thể tính được hàm g như sau:

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
$g(x)$	1	2	1	4	1	2	1	8	1	2	1	4	1	2	...

b) Trò chơi tích

Trò chơi G được gọi là trò chơi tích của các trò chơi thành phần $G_1, G_2, \ldots, G_n$ nếu mỗi bước di chuyển trong G phải chơi đồng thời tất cả các trò chơi thành phần.

Giả sử chúng ta có n đồ thị trò chơi là $G_1 = (X_1, F_1)$, $G_2 = (X_2, F_2)$, …, $G_n = (X_n, F_n)$. Nếu chơi đồng thời cả n trò chơi này thì chúng ta tổ hợp chúng thành đồ thị trò chơi mới là: $G = (X, F)$. Kí hiệu $G = G_1 \times G_2 \times \ldots \times G_n$. Đồ thị G có: Tập đỉnh X là một tích Đề-các $X = X_1 \times X_2 \times \ldots \times X_n$ gồm tất cả các bộ $x = (x_1, x_2, \ldots, x_n)$ với $x_i \in X_i$, $i = 1, 2, \ldots, n$. Với mỗi đỉnh $x \in X$, tập các đỉnh theo sau x là:

$$F(x) = F(x_1, x_2, \ldots, x_n) = \{(F_1(x_1), F_2(x_2), \ldots, F_n(x_n))\}.$$

c) Lật xu hai chiều

Đây là trò chơi thực hiện trên một bảng hai chiều (hàng và cột được đánh số từ 0). Có một số đồng xu được đặt tại một số ô của bảng. Một lượt đi là lật đồng xu tại ô (x, y) từ ngửa thành sấp và lật một hoặc nhiều đồng xu khác trong hình chữ nhật $\{(0, 0), (x, y)\}$ có góc trái trên là ô $(0, 0)$ và góc phải dưới là ô (x, y).

Ví dụ. Trò chơi lật xu bốn góc *Turning Corners*

Một lần đi là lật bốn đồng xu tại bốn góc của một hình chữ nhật $\{(a,b),(x,y)\}$ thoả mãn $0 \le a < x$, $0 \le b < y$. Đồng xu tại vị trí (x, y) đang ngửa được lật thành sấp, ba đồng xu kia đang sấp được lật thành ngửa. Hàm Sprague-Grundy:

$$g(x,y) = \max\{g(x,b) \oplus g(y,a) \oplus g(a,b) : 0 \le a < x,\ 0 \le b < y\},$$

Sở dĩ có công thức trên vì trên bảng hai chiều ứng với mỗi ô (x, y) có xu ngửa là một trò chơi lật xu hai chiều chỉ có xu tại (x, y) là ngửa, còn các xu khác đều sấp.

Với điều kiện hình chữ nhật không rỗng thì $g(x,0) = g(0,y) = 0$, $g(1,1) = 1$ và $g(x,1) = \max\{g(a,1) : 0 \le a < x\} = x$; $g(1,y) = \max\{g(1,b) : 0 \le b < y\} = y$.

Bảng sau đây cho các giá trị SG của các vị trí (x, y) trong trò chơi này.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	$\boxed{3}$	4	5	6	7	8	9	10	11	12	13	14	15
2	0	2	3	1	8	10	11	9	12	14	15	13	4	6	7	5
3	0	3	$\boxed{1}$	2	12	15	13	14	4	7	5	6	8	11	9	10
4	0	4	8	12	$\boxed{6}$	2	14	10	11	15	3	7	13	9	5	1
5	0	5	10	15	2	7	8	13	3	6	9	12	1	4	11	14
6	0	6	11	13	14	8	5	3	7	1	12	10	9	15	2	4
7	0	7	9	14	10	13	3	4	15	8	6	1	5	2	12	11
8	0	8	12	4	11	3	7	15	13	5	1	9	6	14	10	2
9	0	9	14	7	15	6	1	8	5	12	11	2	10	3	4	13
10	0	10	15	5	3	9	12	6	1	11	14	4	2	8	13	7
11	0	11	13	6	7	12	10	1	9	2	4	15	14	5	3	8
12	0	12	4	8	13	1	9	5	6	10	2	14	11	7	15	3
13	0	13	6	11	9	4	15	2	14	3	8	5	7	10	1	12
14	0	14	7	9	5	11	2	12	10	4	13	3	15	1	8	6
15	0	15	5	10	1	14	4	11	2	13	7	8	3	12	6	9

Xét trò chơi lật xu bốn góc với trạng thái ban đầu như bảng a) dưới đây:

S	*S*	*S*	*S*	*S*
S	*S*	*S*	*N*	*S*
S	*S*	*S*	*S*	*S*
S	*S*	*N*	*S*	*S*
S	*S*	*S*	*S*	*N*

Bảng a) Vị trí ban đầu : $SG = 4$

→

S	*S*	*S*	*S*	*S*
S	*S*	*S*	*N*	*S*
S	*S*	*S*	*S*	*S*
S	*S*	*N*	*N*	*N*
S	*S*	*S*	*N*	*S*

Bảng b) Vị trí sau khi lật : $SG = 0$

Giá trị hàm *SG* ứng với trạng thái ban đầu của bảng là $3 \oplus 1 \oplus 6 = 4$, đây là vị trí *N*; có thể di chuyển đến vị trí *P* bằng cách lật bốn đồng xu tại bốn góc của hình chữ nhật {(3, 3), (4, 4)}. Giá trị hàm *SG* của vị trí mới là $3 \oplus 1 \oplus 2 \oplus 12 \oplus 12 = 0$.

d) Tích Nim

Tích Nim là phép toán có kí hiệu là $\otimes$ thoả mãn những tính chất sau:

$$x \otimes 0 = 0 \otimes x = 0,$$
$$x \otimes 1 = 1 \otimes x = x,$$
$$x \otimes y = y \otimes x,$$
$$(x \otimes y) \otimes z = x \otimes (y \otimes z),$$
$$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z).$$

Tích Nim liên quan đến số có dạng 2^{2^n} gọi là số Ferma dạng luỹ thừa 2. Giá trị Ferma với $n = 0, 1, 2, \ldots$ tương ứng là 2, 4, 16, 256, 65536. Dựa vào số Ferma, tìm kết quả phép tích Nim như sau:

$$2^{2^n} \otimes x = 2^{2^n} \times x \text{ với mọi } x < 2^{2^n}$$

$$2^{2^n} \otimes 2^{2^n} = \frac{3}{2} \times 2^{2^n}.$$

Mọi giá trị x lớn hơn 0 đều có giá trị nghịch đảo. Ví dụ nghịch đảo của 2 là 3 vì $2 \otimes 3 = 2 \otimes (2 \oplus 1) = (2 \otimes 2) \oplus (2 \otimes 1) = 3 \oplus 2 = 1$.

Ví dụ 1

$2 \otimes 16 = 32;$

$16 \otimes 16 = \frac{3}{2} \times 16 = 24;$

$$24 \otimes 17 = (16 \oplus 8) \otimes (16 \oplus 1) = (16 \otimes 16) \oplus (16 \otimes 1) \oplus (8 \otimes 16) \oplus (8 \otimes 1)$$
$$= 24 \oplus 16 \oplus 128 \oplus 8 = 128;$$

$$5 \otimes 6 = 5 \otimes (2 \oplus 4) = (2 \otimes 5) \oplus (4 \otimes 5) = (2 \otimes (1 \oplus 4)) \oplus (4 \otimes (1 \oplus 4))$$
$$= (1 \otimes 2) \oplus (2 \otimes 4) \oplus (1 \otimes 4) \oplus (4 \otimes 4) = 2 \oplus 8 \oplus 4 \oplus 6 = 8.$$

Trong trường hợp chúng ta không thể phân tích x thành tổng Nim của các số nhỏ hơn thì có thể phân tích x thành tích Nim của các số nhỏ hơn.

Ví dụ 2

$$8 \otimes 8 = (2 \otimes 4) \otimes (2 \otimes 4) = 2 \otimes 2 \otimes 4 \otimes 4 = 3 \otimes 6 = (2 \oplus 1) \otimes (4 \oplus 2)$$
$$= (2 \otimes 4) \oplus (2 \otimes 2) \oplus (1 \otimes 4) \oplus (1 \otimes 2) = 8 \oplus 3 \oplus 4 \oplus 2 = 13;$$

$$3 \otimes (5 \otimes 6) = 3 \otimes 8 = 3 \otimes (2 \otimes 4) = (3 \otimes 2) \otimes 4 = 1 \otimes 4 = 4.$$

Có nhiều trò chơi có thể giải dựa vào tích Nim.

Ví dụ 3. Trò chơi *Tartan*

Cho hai trò chơi lật xu một chiều là G_1 và G_2. Trò chơi Tartan là trò chơi lật xu trên bảng hai chiều hay là trò chơi tích $G_1 \times G_2$ của các trò chơi G_1 và G_2 được mô tả như sau: nếu một lượt đi đúng luật trong G_1 là cách lật các đồng xu tại $x_1,...,x_m$ và một lượt đi đúng luật trong G_2 là cách lật các đồng xu $y_1,...,y_n$ thì cách lật các đồng xu tại các vị trí (x_i, y_j), $i = \overline{1..m}, j = \overline{1..n}$ là một cách đi hợp lệ trong trò chơi tích $G_1 \times G_2$. Giải trò chơi này, dựa vào định lí sau:

Định lí Tartan

> *Nếu $g_1(x)$ và $g_2(y)$ là hàm Sprague-Grundy của trò chơi G_1 và G_2 thì hàm $g(x,y)$ của trò chơi $G_1 \times G_2$ có giá trị là:* $g(x,y) = g_1(x) \otimes g_2(y)$.

Ví dụ 4. Trò chơi tích $G = G_1 \times G_2$ với G_1 và G_2 là các trò lật xu một chiều *Twins* (mỗi lần lật đúng hai xu). Lật **một đồng** xu bất kì tại (x, y) là một phép di chuyển trong trò chơi tích $G_1 \times G_2$: **lật đồng** xu x trong G_1 và lật đồng xu y trong G_2.

Ta có hàm g của trò chơi Twins là: $\mathbf{g(x) = x}$, $x \geq 1$ (*chú ý bắt đầu từ 1*).

Với vị trí như bảng a), ta tính giá trị ***SG* của** vị trí này bằng tích Nim như sau:

Ô (1, 3) trong $G_1 \times G_2 =$ (trò chơi *Twins*) × (trò chơi *Twins*) tương ứng với vị trí 2 trong G_1 và vị trí 4 trong G_2: nên có $g(1, 3) = g_1(2) \otimes g_2(4) = 2 \otimes 4 = 8$, tương tự $g(3, 2) = g_1(4) \otimes g_2(3) = 4 \otimes 3 = 12$, $g(4, 4) = g_1(5) \otimes g_2(5) = 5 \otimes 5 = 7$. Do đó vị trí của bảng a) có giá trị là $8 \oplus 12 \oplus 7 = 3$ đó là vị trí *N*. Một cách di chuyển đến vị trí *P* là lật liên tiếp bốn đồng xu tại vị trí (4, 4), (1, 2), (1, 4) và (4, 2) sẽ có hàm $SG = 1 \oplus 10 \oplus 15 = 4$ khi đó bảng a) thành bảng b) và vị trí bảng b) có $SG = 8 \oplus 12 \oplus 4 = 0$.

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	$\boxed{3}$	4	5
2	0	2	3	1	8	10
3	0	3	$\boxed{1}$	2	12	15
4	0	4	8	12	$\boxed{6}$	2
5	0	5	10	15	2	7

Bảng tích Nim

	0	1	2	3	4
0	S	S	S	S	S
1	S	S	$\boxed{S}$	N	$\boxed{S}$
2	S	S	S	S	S
3	S	S	N	S	S
4	S	S	$\boxed{S}$	S	$\boxed{N}$

Vị trí bảng a) có $SG = 3$

	0	1	2	3	4
0	S	S	S	S	S
1	S	S	$\boxed{N}$	N	$\boxed{N}$
2	S	S	S	S	S
3	S	S	N	S	S
4	S	S	$\boxed{N}$	S	$\boxed{S}$

Vị trí bảng b) có $SG = 0$

Chú ý. Với bài toán tổng quát, không dễ dàng thực hiện được cách đi chiến thắng vì số lượng vị trí *y* đi đến được từ mỗi vị trí *x* có thể rất lớn.

Ví dụ 5. Trò chơi $G = G_1 \times G_2$ với G_1 và G_2 đều là trò chơi *Mock Turtles*. Vị trí ban đầu chỉ có hai ô có xu ngửa (*N*) còn các ô khác là sấp (*S*). Tìm di chuyển tối ưu.

	0	1	2	3	4	5
0	*S*	*N*	*S*	*S*	*S*	*S*
1	*S*	*S*	*S*	*S*	*S*	*S*
2	*S*	*S*	*S*	*S*	*S*	*S*
3	*S*	*S*	*S*	*S*	*S*	*S*
4	*S*	*S*	*S*	*S*	*S*	*N*

Bước 1. Tính giá trị *SG* của vị trí ban đầu:

Dựa vào giá trị *SG* của trò chơi *Mock Turtles*:

$$g(x) = \begin{cases} 2x & \text{nếu } 2x \text{ là bit lẻ} \\ 2x+1 & \text{nếu } 2x \text{ là bit chẵn.} \end{cases}$$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
$g(x)$	1	2	4	7	$\boxed{8}$	$\boxed{11}$	13	14	16	19	21	22	25	26	28	...

và bảng tích Nim, có thể tạo bảng SG cho trò chơi $G = G_1 \times G_2$ này như sau:

	1	2	4	7	8	11
1	1	2	4	7	8	11
2	2	3	8	9	12	13
4	4	8	6	10	11	7
7	7	9	10	4	15	1
8	8	12	11	15	13	[redacted]

Có thể thấy rằng hai ô ngửa là ô (0, 1) có $SG = g_1(0) \otimes g_2(1) = 1 \otimes 2 = 2$ và ô (4, 5) có $SG = g_1(4) \otimes g_2(5) = 8 \otimes 11 = 9$, suy ra SG của bảng là $2 \oplus 9 = 11$. Suy ra, bước đi chiến thắng là biến đổi ô (4, 5) sao cho SG từ 9 thành 2. Tuy nhiên không dễ để tìm ra cách biến đổi này (có 176 cách khác nhau).

Bước 2. Biến đổi ô $(x, y) = (4, 5)$ sao cho SG từ 9 thành 2: Tại ô (4, 5) có $SG = v_1 \otimes v_2$ mà $(v_1, v_2) = (8, 11)$ trong đó $v_1 = g_1(x)$, $v_2 = g_2(y)$. Ta cần tìm (u_1, u_2) mà trong G_1 có u_1 là giá trị của vị trí mới sau di chuyển từ vị trí x, trong G_2 có u_2 là giá trị của vị trí mới sau di chuyển từ vị trí y với $u_1 < v_1$ và $u_2 < v_2$, đồng thời thoả mãn:

$$(u_1 \otimes u_2) \oplus (u_1 \otimes v_2) \oplus (v_1 \otimes u_2) = 2.$$

Một khả năng là $(u_1, u_2) = (3, 10)$ do

$$(3 \otimes 10) \oplus (3 \otimes 11) \oplus (8 \otimes 10) = 5 \oplus 6 \oplus 1 = 2$$

(cũng có thể là (1, 1) nhưng do cách đi này không mô tả được cho phương pháp cần tìm nên không chọn).

Bước 3. Tìm cách di chuyển cụ thể để chuyển giá trị hàm g_1 từ 8 thành 3 (trong G_1) và chuyển giá trị hàm g_2 từ 11 thành 10 (trong G_2). Để chuyển giá trị hàm g_1 từ 8 thành 3 (trong G_1) có thể lật xu tại 0, 1 và 4 vì

$g_1(0) \oplus g_2(1) = 1 \oplus 2 = 3$. Để chuyển g_2 từ 11 về thành 10 (trong G_2), ta lật xu tại 1, 4 và 5 vì $g_2(1) \oplus g_2(4) = 2 \oplus 8 = 10$.

Do đó trong trò chơi $G = G_1 \times G_2$ cần di chuyển 9 đồng xu tại các vị trí thuộc tích $\{0, 1, 4\} \times \{1, 4, 5\}$. Sau di chuyển sẽ tới vị trí như bảng sau:

	0	1	2	3	4	5
0	S	S	S	S	N	N
1	S	N	S	S	N	N
2	S	S	S	S	S	S
3	S	S	S	S	S	S
4	S	N	S	S	N	S

Kiểm tra lại ta sẽ thấy hàm SG tại vị trí này bằng 0.

Tổng quát giải bài toán Tartan. Giả sử chúng ta đang đứng tại vị trí (x,y) của trò chơi tích $G_1 \times G_2$ với SG là $g(x,y) = g_1(x) \otimes g_2(y) = v$. Kí hiệu $v_1 = g_1(x)$, $v_2 = g_2(y)$ và $v = (v_1, v_2)$. Cần di chuyển đến vị trí mới có SG là $u = (u_1, u_2)$ trong đó $u_1 = g_1(x')$ mà x' là vị trí sau x trong G_1, $u_2 = g_2(y')$ mà y' là vị trí sau y trong G_2. Thực hiện các bước sau:

Bước 1. Tìm cặp (u_1, u_2) sao cho: $(u_1 \otimes u_2) \oplus (u_1 \otimes v_2) \oplus (v_1 \otimes u_2) = u$.

Bước 2. Tìm tập M_1 trong G_1 biến giá trị $g_1(x) = v_1$ thành giá trị u_1 với $u_1 < v_1$.

Bước 3. Tìm tập M_2 trong G_2 biến giá trị $g_2(y) = v_2$ thành giá trị u_2 với $u_2 < v_2$.

Lúc đó $M_1 \times M_2$ là tập cách đi trong $G_1 \times G_2$ để tới vị trí có giá trị SG là u.

Bài tập

9.1. Xem xét trò chơi *Lấy bớt quân cờ* nhưng với quy định người có lượt lấy cuối cùng thì thua. Phân tích xem tập P là tập nào?

9.2. Tìm tập hợp các vị trí P đối với trò chơi trừ dần, nếu tập trừ là:

a) $S = \{1, 3, 5, 7\}$;

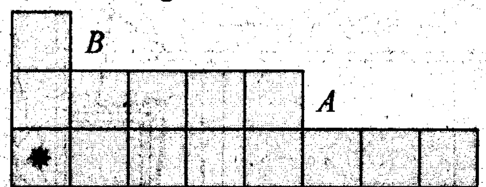
b) $S = \{1, 3, 6\}$;

c) $S = \{1, 2, 4, 8, 16, \ldots\}$ là tập các luỹ thừa của 2.

9.3. ***Trò chơi Chomp!*** (Bẻ miếng sôcôla)

Cần di chuyển những ô vuông trong bảng hình chữ nhật gồm $m \times n$ ô vuông. Mỗi phép di chuyển là lấy đi một ô vuông và các ô vuông ở bên phải nó cùng các ô vuông phía trên chúng. Hai người chơi lần lượt di chuyển, người nào chọn phải ô (1, 1) thì thua cuộc (ô này coi như bị nhiễm độc!).

Ví dụ chơi với bảng 3×8. Người A lấy đi ô (6, 2) cùng các ô bên phải và các ô phía trên (lấy đi một hình chữ nhật có tất cả sáu ô). Sau đó người B lấy ô (2, 3) và các ô bên phải (lấy đi hình chữ nhật có tất cả bốn ô). Khi đó các ô vuông còn lại trên bảng như hình dưới:



a) Tìm ra một phép chuyển tiếp theo giành thắng cho A.

b) Có thể chứng minh A có thuật thắng trong bảng tổng quát $m \times n$ ô vuông hay không?

9.4. ***Trò chơi Phép trừ động***

Có thể mở rộng thêm lớp trò chơi trừ dần bằng cách tạo ra tập trừ S phụ thuộc vào lần di chuyển trước của đối thủ.

Có một cọc N thẻ, người chơi đầu tiên có thể lấy bao nhiêu thẻ tuỳ ý nhưng ít nhất là một thẻ và không được lấy tất cả. Sau đó, đến lượt người kia và hai người thay nhau lần lượt chơi, mỗi người chơi không được phép lấy số thẻ nhiều hơn số thẻ mà đối thủ vừa lấy.

a) Người thứ nhất di chuyển như thế nào để thắng nếu số thẻ là $N = 44$.

b) Với giá trị nào của N thì người chơi sau thắng?

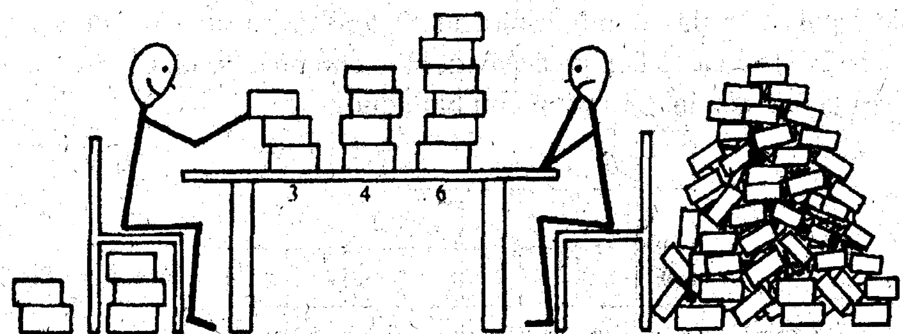
9.5. ***Trò chơi Fibonacci Nim***

Có một cọc N thẻ, người chơi đầu tiên có thể lấy bao nhiêu thẻ tuỳ ý nhưng ít nhất là một thẻ và không được lấy tất cả. Hai người chơi lần lượt, mỗi người chơi có thể lấy số thẻ nhiều nhất bằng hai lần số thẻ mà đối thủ lấy trong lượt đi trước. Hạn chế: $1 < N \leq 100$.

Lập trình tìm chiến thuật thắng.

9.6. *Trò chơi Porker Nim*

Đây là trò chơi với các cọc là các chồng hộp gỗ. Cũng như trò NIM chuẩn, ngoại trừ một điều là có thể giảm kích thước của một vài cọc và phép di chuyển là có thể giảm hoặc tăng thêm kích thước của một vài cọc bằng các hộp gỗ đã chiếm được trong các bước di chuyển trước. Trong hình vẽ dưới đây là trò chơi ở thời điểm ba chồng hộp gỗ có kích thước là 3, 4 và 6 sau khi hai người chơi đã di chuyển một số bước và thu đã được một số hộp. Bây giờ đến lượt đấu thủ bên trái (gọi là A). Vậy chiến thuật của A như thế nào để thắng (nếu trò chơi không cho phép kéo dài mãi)?



9.7. *Trò chơi Lật Rùa (Turning Turtles)*

Trên đường thẳng nằm ngang có n con rùa ngẫu nhiên đang bò hoặc đang nằm ngửa. Một phép chuyển là lật một con rùa đang bò thành nằm ngửa và nếu muốn còn có thể lật một con rùa khác ở bên trái nó (đang bò thành nằm ngửa hoặc đang nằm ngửa thành đang bò). Người chơi nào có phép di chuyển cuối cùng thì thắng.

a) Chứng tỏ rằng trò chơi này là Nim nếu mỗi con rùa đang bò ở vị trí n là một cọc có n quân.

b) Nếu a) đúng, tìm chiến thuật thắng cho trạng thái nêu ở hình vẽ sau:

9.8. *Trò chơi Moore's Nim$_k$*

Có n cọc với số quân tương ứng là $x_1, x_2, \ldots, x_n$ và quá trình chơi như trong Nim nhưng một phép chuyển là người chơi có thể chuyển số quân tuỳ ý từ k cọc bất kì (k là số cố định cho trước) và ít nhất phải có một quân chuyển đi từ một cọc nào đó.

Trò chơi Nim_k với $k = 1$ chính là trò chơi Nim chuẩn với n cọc. Để giải trò chơi này chúng ta áp dụng định lí sau:

Định lí Moore

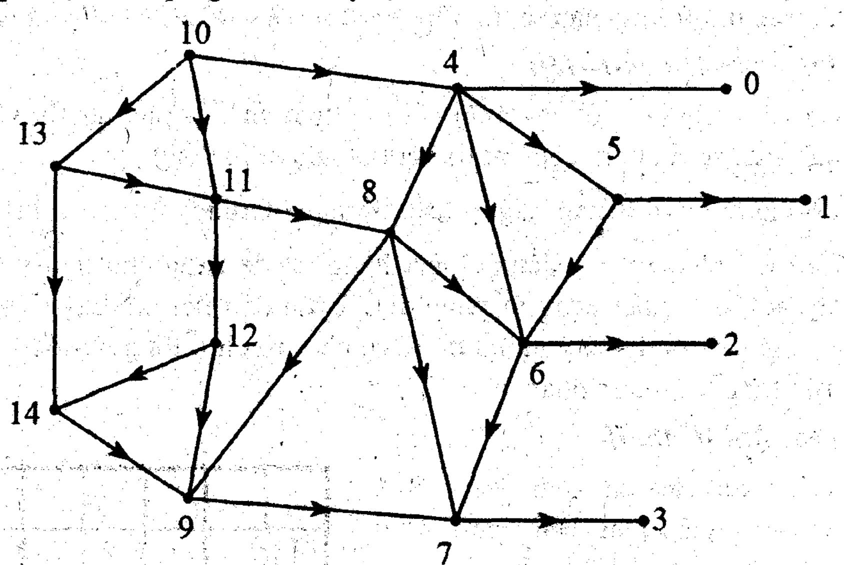
> *Vị trí $(x_1, x_2, ..., x_n)$ là vị trí P trong Nim_k khi và chỉ khi $x_1, x_2, ..., x_n$ viết dưới dạng cơ số 2 trong phép cộng không nhớ theo môđun $(k+1)$ cho kết quả bằng 0. Nói cách khác số các số 1 ở mỗi cột trong phép cộng này đều chia hết cho $k+1$.*

Hãy lập trình trò chơi Nim_k giữa máy và người, cho máy đi trước.

Input: Tệp *bai9_8.inp*. Mỗi dòng cho một cấu hình của trò chơi: đầu dòng là số k nguyên dương sau đó là n số nguyên dương thể hiện số quân trên n cọc $(1 \le k \le n)$.

Output: Tệp *bai9_8.out* có số dòng tương ứng. Đầu mỗi dòng ghi số 1 hoặc 0 thể hiện máy có thể đạt chiến thắng hoặc không chắc thắng, nếu ghi số 1 thì sau đó ghi ra n số thể hiện số quân còn lại trên các cọc 1, 2, ..., n sau bước đi thứ nhất của máy.

9.9. Tính giá trị hàm Sprague-Grundy tại các đỉnh của đồ thị sau:



9.10. Tìm hàm Sprague-Grundy cho trò chơi trừ dần với tập trừ

a) $S = \{1, 3, 4\}$;

b) S là tập các số nguyên tố lẻ;

c) S là tập các số nguyên tố;

d) S là tập các số Fibonacci;

e) S là tập các số Lucas.

Số Lucas được định nghĩa đệ quy:

$$L_1 = 1, L_2 = 3, L_{i+2} = L_{i+1} + L_i \ (i = 1, 2, \ldots).$$

9.11. Trò chơi *At-Most-Haft* là trò chơi trên một cọc với quy luật phải chuyển ít nhất là một quân và nhiều nhất là nửa số quân trên cọc (tất nhiên vị trí kết thúc là 0 hoặc 1). Xây dựng hàm Sprague-Grundy với trò chơi này.

9.12. ***Trò chơi Dim⁺***

Giả sử có một cọc n quân. Mỗi lần loại bỏ đi c quân từ cọc với điều kiện c là ước của n (kể cả 1 và n). Chỉ có một vị trí kết thúc là 0.

Tính hàm Sprague-Grundy.

9.13. ***Trò chơi Chẵn-Lẻ***

Cho một cọc chứa n quân. Quy tắc chuyển quân khỏi cọc là: có thể bớt đi một số chẵn quân nhưng không phải là toàn bộ cọc hoặc có thể bớt đi toàn bộ cọc có số quân lẻ. Trò chơi có hai vị trí kết thúc là 0 và 2.

Chứng minh (quy nạp) công thức: $g(2k) = k - 1$; $g(2k - 1) = k$, với $k \geq 1$.

9.14. ***Trò chơi At-Least-Haft***

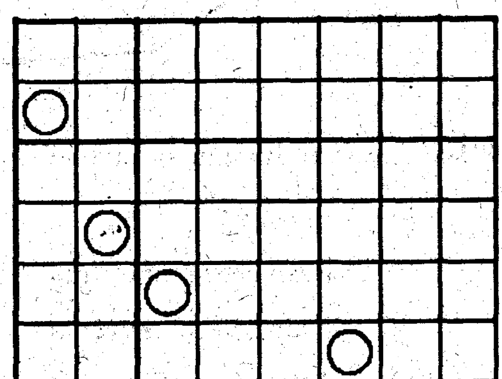
Trò chơi chỉ có một cọc mẫu với quy luật chơi là phải di chuyển ít nhất là một nửa số thẻ trên cọc. Vị trí kết thúc duy nhất là 0.

Chứng minh (quy nạp) giá trị hàm Sprague-Grundy $g(x) = \min\{k : 2^k > x\}$.

Lưu ý. Trò chơi này dành chiến thắng rất dễ dàng cho người thứ nhất (là lấy hết mọi quân ngay từ phép di chuyển đầu tiên). Nhưng việc tính hàm Sprague-Grundy có ý nghĩa phục vụ cho trò chơi tổng mà At-Least-Haft là một trò chơi thành phần.

9.15. ***Trò chơi Wythoff***

Trên bàn cờ vua kích thước 8×8 đặt một quân hậu. Hai người chơi với mỗi lượt đi là di chuyển quân hậu với số bước tuỳ ý theo hướng thẳng xuống, ngang sang trái, hoặc đi chéo xuống góc trái-dưới. Khi quân hậu đi đến góc trái-dưới thì trò chơi kết thúc.



Tìm hàm Sprague-Grundy của trò chơi.

9.16. Tìm phép chuyển tối ưu từ vị trí ban đầu cho trò chơi tổng của ba trò chơi:

- Trò chơi chẵn-lẻ với cọc 18 quân;
- Trò chơi *At-Least-Haft* với cọc 17 quân;
- Trò chơi Nim chuẩn một cọc 7 quân.

9.17. ***Trò chơi Grundy***

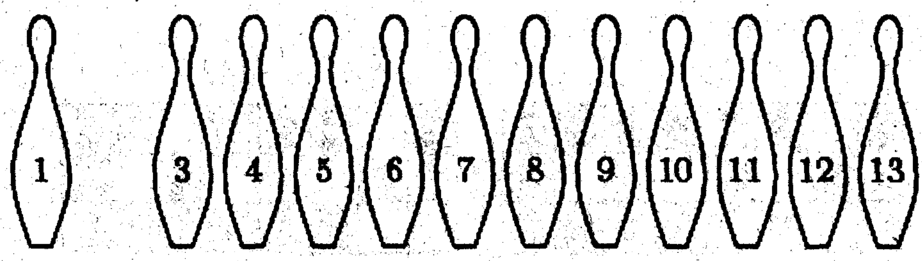
Trò chơi này có phép chuyển hợp lệ là chia một cọc thành hai cọc có kích thước khác nhau. Người nào có phép chuyển cuối cùng thì thắng (rõ ràng có hai vị trí kết thúc là cọc 1 quân và cọc 2 quân).

a) Tính giá trị hàm Sprague-Grundy cho trò chơi này khi cọc ban đầu có kích thước n quân (với $n = 1, 2, 3, \ldots, 13$).

b) Xét trò chơi *Grundy* với ba cọc ban đầu có số quân là 5, 8 và 13. Tìm các phép chuyển tối ưu đầu tiên nếu có.

9.18. ***Trò chơi Kayler***

Có 13 quân bowing sắp thành một hàng, nhưng quân thứ hai đã bị hạ đổ (xem hình vẽ).

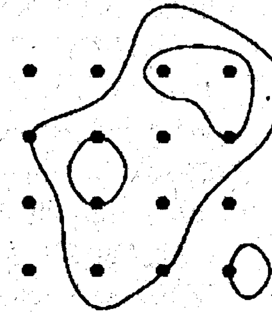


a) Chứng tỏ vị trí này là N.

b) Tìm phép chuyển chiến thắng. Khi đó quân bowing nào sẽ bị hạ?

9.19. ***Trò chơi Rims***

Một vị trí của trò chơi Rims là một tập hữu hạn chấm tròn trên mặt phẳng có thể bị ngăn cách bởi một vài đường vòng kín không cắt nhau. Một phép di chuyển là vẽ một vòng kín đi qua một vài chấm tròn nào đó (ít nhất là một điểm) nhưng không cắt các vòng khác. Hai người chơi lần lượt thực hiện di chuyển, người có phép chuyển cuối cùng là người chiến thắng.

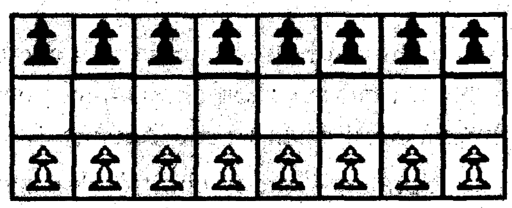


a) Chứng tỏ trò chơi này là một dạng của Nim;

b) Với vị trí cho như hình vẽ, tìm phép di chuyển chiến thắng nếu có.

9.20. *Trò chơi cờ Dawson*

Có hai hàng quân tốt đen và trắng đối diện nhau, cách nhau một hàng trống. Một quân tốt chỉ có thể đi lên một ộ trống phía trước hoặc buộc phải bắt một quân của đối phương đã ở ô kề chéo bên trái hoặc bên phải của ô chứa quân tốt này.

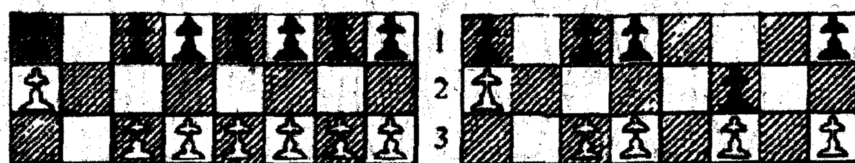


Tìm cách chơi để người chơi sau sẽ thắng.

Ví dụ. Nếu Trắng đi từ ô a3 đến ô a2 (kí hiệu a3-a2), Đen đi b1-a2 bắt quân Trắng ở a2, Trắng đi b3-a2 bắt quân đen tại a2. Có hình ảnh như hình (a): một cặp quân tốt bị loại và *đổi lượt đi*.

Bây giờ đến lượt Đen đi. Giả sử Đen đi f1-f2, Trắng e3-f2, Đen e1-f2, Trắng g3-f2, Đen g1-f2. Có kết quả như hình (b): hai cặp quân tốt bị loại và *đổi lượt đi*.

(a) (b)



a b c d e f g h a b c d e f g h

Do vậy trò chơi này có thể coi là trò chơi trên một hàng có n ô vuông để trống. Hai người chơi lần lượt ghi X vào một ô trống không kề với ô đã có X. Người ghi chữ X cuối cùng thắng.

Trò chơi này có thể mô tả như trò chơi trên một cọc có thể bỏ bớt các quân trên cọc hoặc chia cọc thành hai cọc. Nếu $n = 1$ có đúng một phép chuyển tới $n = 0$. Với $n > 1$. Một phép ghi X vào ô ở một trong hai đầu dòng là tương ứng loại trừ hai quân ở đầu đó của cọc. Đặt X tại ô cách ô ở một trong hai đầu khoảng cách một ô tương đương loại trừ ba quân khỏi cọc. Đặt X tại ô không phải như hai loại trên (và không phải tại ô kề ô X đã có) tương đương loại trừ ba quân khỏi cọc và chia cọc thành hai cọc mới. Vậy quy luật di chuyển trong trò chơi này là:

(1) Nếu cọc chỉ có một quân thì loại bỏ quân này;

(2) Có thể loại bỏ hai quân;

(3) Có thể loại bỏ ba quân;

(4) Chia cọc đó thành hai cọc có tổng số quân giảm đi 3.

Hãy tìm bước chuyển chiến thắng khi $n = 18$.

9.21. Lập chương trình tính tích Nim của hai số x và y.

9.22. ***Trò chơi lật xu Twins***

Cho dãy $n + 1$ đồng xu đánh số từ 0 đến n, đồng xu n ngửa, các đồng xu khác sấp. Một lượt đi là lật đồng xu ngửa thành sấp và một đồng xu khác tuỳ ý ở bên trái đang sấp thành ngửa.

Xây dựng hàm *SG* cho các vị trí của trò chơi này.

9.23. ***Trò chơi lật xu Ruler***

Cho một hàng gồm n đồng xu, đánh số từ 1 đến n, đồng xu n đang ngửa, còn các đồng xu khác thì sấp. Một lượt đi là lật đồng thời một số đồng xu với điều kiện các đồng xu là liên tiếp và đồng xu bên phải nhất đang ngửa bị lật thành sấp. Xây dựng hàm *SG* cho các vị trí trong trò chơi này.

9.24. ***Trò chơi Grunt***

Cho một hàng gồm $n + 1$ đồng xu, đánh số từ 0 đến n, đồng xu n đang ngửa, còn các đồng xu khác thì sấp. Một lượt đi là lật bốn đồng xu có vị trí đối xứng qua vị trí chính giữa của đồng xu bên trái nhất và đồng xu bên phải nhất. Xây dựng hàm *SG* cho các vị trí trong trò chơi này.

9.25. ***Trò chơi Turning Corners***

Bàn cờ vuông kích thước $n \times n$ được chia thành các ô vuông và đánh toạ độ như sau: ô góc trên-phải có toạ độ (0; 0), hoành độ tăng dần từ trái qua phải, tung độ tăng dần từ trên xuống dưới. Trên mỗi ô vuông có thể đặt một đồng xu (ngửa hoặc sấp). Hai người chơi lần lượt. Một lượt đi là chọn một hình chữ nhật nhỏ trong bàn cờ có góc phải-dưới là (x; y) mà đồng xu tại ô này đang ngửa và ba đồng xu ở ba góc còn lại của hình chữ nhật nhỏ này đang sấp. Sau đó người chơi cần lật đồng xu đang ngửa thành sấp và ba đồng xu ở ba góc còn lại của hình chữ nhật cần lật thành ngửa. Người chơi nào sấp được đồng xu cuối cùng thì thắng cuộc.

Lập trình cho biết trạng thái ban đầu của bàn cờ là N hay P? Nếu trạng thái là N, hãy tìm một bước đi chiến thắng.

Input: Tệp *Bai9_25.inp* chứa trạng thái ban đầu của bàn cờ:

- Dòng đầu tiên là số nguyên dương n;

- Tiếp theo là $n + 1$ dòng thể hiện bàn cờ, mỗi dòng có $n + 1$ số 0 hoặc 1 (số 0 thể hiện đồng xu sấp, số 1 thể hiện đồng xu ngửa).

Output: Tệp *Bai9_25.out*

- Dòng đầu ghi số 1 nếu trạng thái ban đầu của bàn cờ là N, ghi số 0 nếu trạng thái ban đầu là P.
- Nếu trạng thái ban đầu là N thì dòng thứ hai ghi toạ độ (x, y) ở góc phải-dưới của hình chữ nhật nhỏ trong bước đi chiến thắng. Hạn chế: $n < 21$.

III. Trò chơi hai người có tổng điểm bằng 0

1. Mô tả

a) Dạng chiến thuật

Mô hình toán học đơn giản của trò chơi *Hai người chơi có tổng điểm bằng 0* là bộ ba (X, Y, A). Trong đó:

- X là tập không rỗng, gọi là tập chiến thuật cho người thứ nhất;
- Y là tập không rỗng, gọi là tập chiến thuật cho người thứ hai;
- A là hàm lượng giá có giá trị thực, xác định trên tập $X \times Y$ (mỗi phần tử của tập là cặp (x, y) mà $x \in X$ và $y \in Y$); với mỗi cặp (x, y) xác định duy nhất một giá trị $A(x, y)$ là một số thực.

Lưu ý. Trong trò chơi này, với mỗi cặp $(x; y)$ hàm lượng giá điểm cho người thứ hai là giá trị đối của hàm lượng giá điểm cho người thứ nhất nên chỉ cần quan tâm tới hàm lượng giá tính điểm cho người thứ nhất (là A).

Hai người chơi phải *chọn đồng thời* chiến thuật của mình: người thứ nhất chọn $x \in X$ và người thứ hai chọn $y \in Y$. Sau sự lựa chọn của hai người, nếu $A(x, y) > 0$ thì người thứ nhất thắng $A(x, y)$ điểm từ người thứ hai, ngược lại nếu $A(x, y) < 0$ thì người thứ nhất phải trả $|A(x, y)|$ điểm cho người thứ hai.

b) Chiến thuật cân bằng

Để minh hoạ cho chiến thuật cân bằng, chúng ta xét trò chơi *Chẵn-Lẻ* như sau:

Quy ước gọi người thứ nhất là Lẻ, người thứ hai là Chẵn. Giả sử tập chiến thuật của người thứ nhất là $X = \{1, 2\}$, tập chiến thuật của người thứ hai là $Y = \{1, 2\}$. Hai người đồng thời chọn: người thứ nhất chọn $x \in X$, người thứ hai chọn $y \in Y$;

nếu $x + y$ lẻ thì người thứ nhất thắng $A(x, y) = x + y$ điểm, ngược lại nếu $x + y$ chẵn thì người thứ nhất nhận số điểm là $-(x + y)$ nghĩa là mất $(x + y)$ điểm. Vậy khi tính điểm cho người thứ nhất thì bảng điểm $A(x, y)$ như sau:

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Người thứ hai chọn y bằng</td></tr>
<tr><td>1</td><td>2</td></tr>
<tr><td rowspan="2">Người thứ nhất chọn x bằng</td><td>1</td><td>–2</td><td>+3</td></tr>
<tr><td>2</td><td>+3</td><td>–4</td></tr>
</table>

Giả sử người thứ nhất chọn chiến thuật “1” với tần suất $\frac{3}{5}$, chọn chiến thuật “2” với tần suất $\frac{2}{5}$ (chẳng hạn trong 1000 lần chơi, người thứ nhất có 600 lần chọn “1” và 400 lần chọn”2”).

a) Nếu người thứ hai chọn “1” thì người thứ nhất sẽ mất $2\times\frac{3}{5}$ điểm và sẽ được $3\times\frac{2}{5}$ điểm trong một lần. Vậy trung bình một lần chọn người thứ nhất thu được $3\times\frac{2}{5}-2\times\frac{3}{5}=0$ (điểm).

b) Nếu người thứ hai chọn 2 thì người thứ nhất được $3\times\frac{3}{5}$ điểm và mất $4\times\frac{2}{5}$ điểm trong một lần. Vậy trung bình một lần chọn người thứ nhất thu được $3\times\frac{3}{5}-4\times\frac{2}{5}=\frac{1}{5}=0.2$ (điểm).

Người thứ nhất với sự lựa chọn như trên, sẽ hoà khi người thứ hai chọn “1” và sẽ thắng 0.2 điểm khi người thứ hai chọn “2”. Vậy bằng chiến thuật người thứ nhất chọn “1” với tần suất $\frac{3}{5}$, chọn “2” với tần suất $\frac{2}{5}$, người thứ nhất luôn thắng, bất kể người thứ hai chọn “1” hay “2”. Xét trường hợp tổng quát:

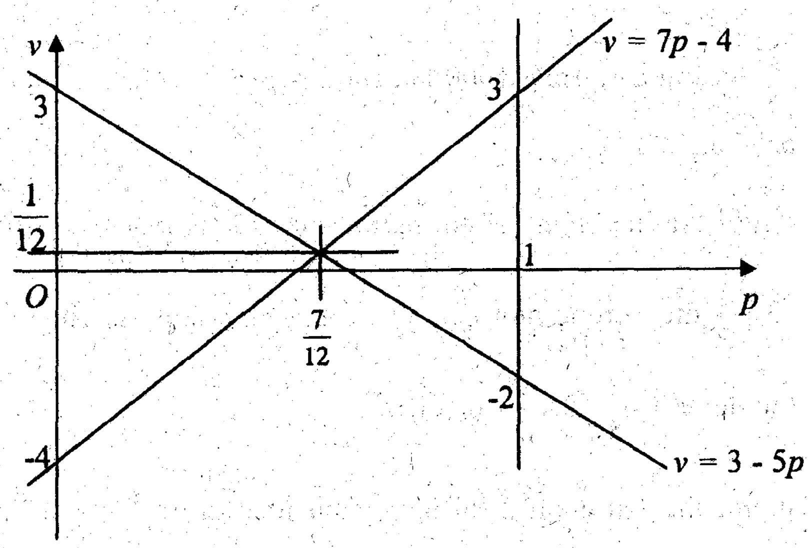
Giả sử tần suất chọn "1" của người thứ nhất là p, chọn "2" là $1 - p$ thì giá trị trung bình trả về cho người thứ nhất khi người thứ hai chọn "1" là

$$3(1 - p) - 2p = 3 - 5p$$

và khi người thứ hai chọn "2" là $3p - 4(1 - p) = 7p - 4$.

Vậy cần chọn p bằng một giá trị nào đó sao cho người thứ nhất đạt được điểm cao nhất mà không phụ thuộc vào cách chọn của người thứ hai.

Ta xét các đồ thị $v = 3 - 5p$ và $v = 7p - 4$ trên đoạn $[0; 1]$. Giá trị của v thể hiện giá trị trả về người thứ nhất. Hai đồ thị này giao nhau tại điểm $(p; v) = \left(\frac{7}{12}; \frac{1}{12}\right)$.



Hình 9.4. Mô phỏng trò chơi cân bằng

Điểm $\left(\frac{7}{12}; \frac{1}{12}\right)$ là điểm thể hiện người thứ nhất thắng $\frac{1}{12}$ điểm không phụ thuộc vào cách chọn của người thứ hai. Cách chơi của người thứ nhất chọn "1" với xác suất $\frac{7}{12}$, chọn "2" với xác suất $\frac{5}{12}$ như thế gọi là chiến thuật cân bằng.

> *Cách chơi đạt được chiến thắng (tính trung bình) không phụ thuộc vào đối phương chơi như thế nào được gọi là* ***chiến thuật cân bằng*** *hay chiến thuật MiniMax.*

Một số khái niệm khác trong trò chơi hai người có tổng điểm bằng 0:

- Mỗi chiến thuật trong các tập chiến thuật X hoặc Y là những chiến thuật thuần tuý. Tổ hợp các chiến thuật thuần tuý với các tần suất khác nhau ta có một chiến thuật hỗn hợp. Tất nhiên mỗi chiến thuật thuần tuý $x \in X$ cũng có thể xem là một chiến thuật hỗn hợp nếu coi như chiến thuật thuần tuý này được chọn với tần suất bằng 1, các chiến thuật thuần tuý còn lại được chọn với tần suất bằng 0.
- Trò chơi hai người có tổng điểm bằng 0 với mô hình toán học (X, Y, A) gọi là có giới hạn khi và chỉ khi X và Y là các tập hữu hạn và tồn tại hàm lượng giá A.
- Trong các định lí cơ bản về lí thuyết *Trò chơi hai người có tổng điểm bằng 0* Von Neumann có định lí nổi tiếng sau đây:

Định lí Minimax

> *Với trò chơi hai người có tổng điểm bằng 0 và có giới hạn thì:*
>
> *(1) Tồn tại một số V hữu hạn (gọi là giá trị của trò chơi).*
>
> *(2) Có một chiến thuật hỗn hợp cho người thứ nhất mà giá trị trung bình đạt được của người thứ nhất ít nhất là V, bất kể người thứ hai đi như thế nào.*
>
> *(3) Có một chiến thuật hỗn hợp cho người thứ hai mà giá trị trung bình người thứ hai phải trả nhiều nhất là V, bất kể người thứ nhất đi như thế nào.*

Nếu $V = 0$ thì ta nói trò chơi dễ hoà. Nếu $V > 0$ thì ta nói trò chơi dành ưu tiên (có lợi) cho người thứ nhất, $V < 0$ thì trò chơi dành ưu tiên cho người thứ hai.

Ví dụ. An và Bình dùng một số quân bài Tú lơ khơ để tổ chức trò chơi. An có quân Át đen (số của quân Át coi như là 1) và quân 8 đỏ, Bình có quân 2 đỏ và quân 7 đen. Hai người đồng thời chọn một trong hai quân bài của mình. Nếu hai quân bài được chọn của hai người là cùng màu thì An thắng, ngược lại thì Bình thắng. Số điểm đạt được là tổng các số ghi trên hai quân bài. Tìm hàm A, giá trị trò chơi và chiến thuật hỗn hợp tối ưu của trò chơi.

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Bình chọn</td></tr>
<tr><td>2 đỏ</td><td>7 đen</td></tr>
<tr><td rowspan="2">An chọn</td><td>Át (1) đen</td><td>−3</td><td>8</td></tr>
<tr><td>8 đỏ</td><td>10</td><td>−15</td></tr>
</table>

Phân tích trò chơi trên quan điểm của An. Giả sử An chọn "Át (1) đen" với tần suất p và chọn "8 đỏ" với tần suất $1 - p$. Khi đó: Nếu Bình chọn "2 đỏ" thì giá trị trung bình trả về cho An là $-3p + 10(1 - p) = 10 - 13p$. Nếu Bình chọn "7 đen" thì giá trị trung bình trả về cho An là $8p - 15(1 - p) = 23p - 15$. Thuật tối ưu cho An là chọn p sao cho: $10 - 13p = 23p - 15 \Leftrightarrow p = \frac{23}{36}$. Khi đó An nhận được trung bình ít nhất cũng là $\frac{35}{36}$ điểm.

Phân tích trò chơi theo quan điểm của Bình. Giả sử Bình chọn "2 đỏ" với tần suất p và chọn "7 đen" với tần suất $1 - p$. Khi đó: Nếu An chọn "At (1) đen" thì giá trị trung bình trả về cho Bình là $3p - 8(1 - p) = 11p - 8$. Nếu An chọn "8 đỏ" thì giá trị trung bình trả về cho Bình là $-10p + 15(1 - p) = 15 - 25p$. Thuật tối ưu cho Bình là chọn p sao cho: $11p - 8 = 15 - 25p \Leftrightarrow p = \frac{23}{36}$. Khi đó giá trị trả cho Bình là $-\frac{35}{36}$, nghĩa là Bình phải trả trung bình nhiều nhất là $\frac{35}{36}$ điểm cho An. Trò chơi này luôn ưu tiên cho đấu thủ An. Giá trị trò chơi là $\frac{35}{36}$.

2. *Trò chơi ma trận*

a) Ma trận lượng giá

Trò chơi hai người có tổng điểm bằng 0 và có giới hạn với dạng chiến thuật (X, Y, A) còn được gọi là *trò chơi ma trận.*

Nếu $X = \{x_1, x_2, ..., x_m\}$ và $Y = \{y_1, y_2, ..., y_n\}$. Hàm lượng giá A có thể biểu diễn bằng một ma trận dạng:

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn} \end{pmatrix},$$

trong đó $A(x_i ; y_j) = a_{ij}$ là giá trị hàm lượng giá tại dòng i, cột j (người thứ nhất chọn chiến thuật nguyên thuỷ x_i và người thứ hai chọn chiến thuật nguyên thuỷ y_j).

Đôi khi ma trận A được viết đơn giản là $A(a_{ij})_{m\times n}$ với $i = \overline{1..m}; j = \overline{1..n}$.

Quy định: Người thứ nhất chọn hàng i ($i = 1, 2, ..., m$), người thứ hai chọn cột j ($j = 1, 2, ..., n$) thì người thứ hai phải trả cho người thứ nhất số điểm a_{ij} ghi tại giao của hàng và cột đã chọn. Vậy giá trị này là điểm thắng cho người chọn hàng, là điểm thua cho người chọn cột. Một chiến thuật hỗn hợp cho người thứ nhất là vectơ $p = (p_1, p_2, ..., p_m)$ gồm các tần suất p_i ($i = 1, 2, ..., m$) có tổng bằng 1. Nếu người thứ nhất sử dụng chiến thuật hỗn hợp này và người thứ hai chọn cột j thì giá trung bình trả cho người thứ nhất là $\sum_{i=1}^{m} p_i a_{ij}$. Tương tự chiến thuật hỗn hợp cho người thứ hai là vectơ $q = (q_1, q_2, ..., q_n)$ gồm các tần suất q_j ($j = 1, 2, \ldots, n$) có tổng bằng 1. Nếu người thứ hai sử dụng q và người thứ nhất chọn hàng i thì giá trị trung bình trả cho người thứ nhất là $\sum_{j=1}^{n} q_j a_{ij}$.

Lưu ý. Trong tính toán, thay việc biểu diễn vectơ p bằng biểu diễn một ma trận có kích thước $m \times 1$, đó là ma trận $p = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_m \end{pmatrix}$. *Ma trận chuyển vị* của p là ma trận $p^T = \begin{pmatrix} p_1 & p_2 & \cdots & p_m \end{pmatrix}$ và cũng có thể biểu diễn vectơ $q = (q_1, q_2, ..., q_n)$ thay bằng ma trận $q = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix}$ có kích thước $n \times 1$. *Ma trận chuyển vị* của nó là ma trận có kích thước $1 \times n$, đó là ma trận $q^T = \begin{pmatrix} q_1 & q_2 & \cdots & q_n \end{pmatrix}$.

b) Điểm yên ngựa

Phần tử a_{ij} của ma trận A được gọi là **phần** tử yên ngựa (điểm yên ngựa) nếu thoả mãn hai điều kiện:

- a_{ij} là phần tử nhỏ nhất trong **hàng** i;
- a_{ij} là phần tử lớn nhất trong **cột** j.

Rõ ràng, nếu a_{ij} là điểm yên ngựa **thì người** thứ nhất sẽ luôn thắng ít nhất là a_{ij} bằng cách chọn hàng i, người thứ **hai có thể** giữ cho sự thiệt hại của mình không quá a_{ij} nếu chọn cột j. Do đó a_{ij} là **giá trị của** trò chơi.

Ví dụ: $A = \begin{pmatrix} 4 & -3 & 1 \\ 3 & 5 & 2 \\ 0 & 6 & 1 \end{pmatrix}$.

Ma trận A có điểm yên ngựa là $a_{23} = 2$. Do đó chiến thuật tối ưu cho người thứ nhất là chọn hàng 2 và cho người thứ hai là chọn cột 3. Giá trị của trò chơi bằng 2 và vectơ $p = (0, 1, 0)$ là chiến thuật hỗn hợp tối ưu cho người thứ nhất và chiến thuật tối ưu cho người thứ hai là vectơ $q = (0, 0, 1)$.

c) Giải trò chơi trên ma trận 2×2

Cho trò chơi có ma trận lượng giá $A = \begin{pmatrix} a & b \\ d & c \end{pmatrix}$. Để giải trò chơi này chúng ta thực hiện như sau:

- Tìm điểm yên ngựa.
- Nếu có điểm yên ngựa thì dùng điểm yên ngựa để xây dựng chiến thuật tối ưu. Nếu không có điểm yên ngựa thì dùng chiến thuật cân bằng.

Giả sử A không có điểm yên ngựa. Khi đó có thể chứng minh:

$$(*)\ \left[\begin{array}{llll} a > b, & b < c, & c > d, & d < a \\ a < b, & b > c, & c < d, & d > a. \end{array}\right.$$

Nếu người thứ nhất chọn dòng đầu với tần suất p (nghĩa là dùng chiến thuật hỗn hợp $(p, 1 - p)$, chúng ta cân bằng các giá trị trung bình trả về cho người thứ nhất khi người thứ hai chọn cột 1 hoặc 2:

$$ap + d(1 - p) = bp + c(1 - p) \Leftrightarrow p = \frac{c - d}{(a - b) + (c - d)} \text{ dễ thấy } 0 < p < 1 \text{ (do (*))}.$$

Giá trị trung bình trả cho người thứ nhất là $v = ap + d(1 - p) = \dfrac{ac - bd}{(a - b) + (c - d)}$.

Tương tự nếu người thứ hai chọn cột 1 với tần suất q, cột 2 với tần suất $1 - q$ thì

$$q = \frac{c - b}{(a - b) + (c - d)} \ (0 < q < 1)$$

và giá trị trung bình người thứ hai trả cho người thứ nhất là $v = \dfrac{ac - bd}{(a - b) + (c - d)}$.

Ta kí hiệu giá trị trò chơi là: $v = Val(A) = Val\begin{pmatrix} a & b \\ d & c \end{pmatrix} = \dfrac{ac-bd}{(a-b)+(c-d)}$.

d) Chiến thuật loại bỏ tính chi phối

Với các trò chơi ma trận kích thước lớn ta có thể giảm kích thước bằng cách xoá đi các hàng hoặc cột không có lợi khi sử dụng chúng. Nếu có thể được, chúng ta cố gắng đưa về ma trận kích thước 2×2.

Định nghĩa 1

Ta nói hàng thứ i của ma trận $A = (a_{ij})$ chi phối hàng thứ k nếu $a_{ij} \geq a_{kj}$ với mọi $j = 1, 2, ..., n$ (khi đó ta cũng nói hàng thứ k bị chi phối bởi hàng thứ i). Ta nói hàng thứ i của ma trận $A = (a_{ij})$ chi phối *nghiêm ngặt* hàng thứ k nếu $a_{ij} > a_{kj}$ với mọi $j = 1, 2, ..., n$. Tương tự, cột thứ j của A chi phối (hoặc chi phối nghiêm ngặt) cột thứ k nếu $a_{ij} \leq a_{ik}$ (hoặc $a_{ij} < a_{ik}$) với mọi $i = 1, 2, ..., m$.

Từ định nghĩa trên suy ra: Những gì người thứ nhất đạt được khi sử dụng một hàng k bị chi phối bởi hàng i đều có thể đạt được khi sử dụng chính hàng i (chi phối k). Do đó những hàng bị chi phối có thể xoá khỏi ma trận trò chơi. Lập luận tương tự, các cột bị chi phối bởi cột khác cũng có thể xoá được. Chiến thuật tối ưu xây dựng trên những hàng và cột còn lại sau khi xoá đi những hàng và cột bị chi phối vẫn là chiến thuật tối ưu của trò chơi. Tuy nhiên, vẫn có thể có những chiến thuật tối ưu sử dụng cả những hàng hoặc cột bị chi phối (không nghiêm ngặt). Do đó trong một số trường hợp, việc xoá hàng hoặc cột có thể làm thay đổi số lượng chiến thuật tối ưu (song vẫn còn ít nhất một chiến thuật tối ưu) và giá trị trò chơi không thay đổi.

Chúng ta còn có thể làm lặp lại công việc xoá hàng hoặc cột nhiều lần (nếu còn hàng hoặc cột bị chi phối) để ma trận có kích thước nhỏ dần.

Ví dụ. $A = \begin{pmatrix} 2 & 0 & 4 \\ 1 & 2 & 3 \\ 4 & 1 & 2 \end{pmatrix}$.

Cột 2 chi phối cột 3, nên có thể xoá cột 3 vì thế $A = \begin{pmatrix} 2 & 0 \\ 1 & 2 \\ 4 & 1 \end{pmatrix}$.

Tiếp theo, hàng 3 chi phối hàng 1 nên có thể xoá hàng 1 vì thế $A = \begin{pmatrix} 1 & 2 \\ 4 & 1 \end{pmatrix}$.

Trong ma trận 2×2 vừa tìm được không có điểm yên ngựa, nên dùng chiến thuật cân bằng thu được: $p = \frac{3}{4}$, $q = \frac{1}{4}$, $v = \frac{7}{4}$. Suy ra chiến thuật tối ưu cho người thứ nhất trong trò chơi ma trận ban đầu là: $(0, \frac{3}{4}, \frac{1}{4})$ và cho người thứ hai là $(\frac{1}{4}, \frac{3}{4}, 0)$. Lưu ý: hàng hoặc cột bị xoá được gán tần suất bằng 0.

Định nghĩa 2

Một hàng k bị chi phối bởi một tổ hợp (theo tần suất) của hàng i_1 và hàng i_2 nếu tồn tại số p $(0 < p < 1)$ mà $p.a_{i_1 j} + (1-p).a_{i_2 j} \geq a_{kj} \quad \forall j = \overline{1..n}$.

Các hàng (hoặc cột) cũng có thể xoá nếu nó bị chi phối bởi một tổ hợp (theo tần suất) các hàng (hoặc cột) khác. Nếu hàng k bị chi phối bởi một tổ hợp theo tần suất của hàng i_1 và hàng i_2 thì có thể xoá hàng k. Với người thứ nhất, chiến thuật hỗn hợp chọn hàng i_1 với tần suất p và chọn hàng i_2 với tần suất $1 - p$, ít nhất cũng tốt hơn dùng chiến thuật chọn hàng k. Khi đó, mọi chiến thuật hỗn hợp có sử dụng hàng k với tần suất p_k sẽ được thay bằng chọn các hàng i_1 và i_2 với các tần suất tương ứng thích hợp được chia ra từ p_k : tần suất của hàng i_1 là $p.p_k$, tần suất của hàng i_2 là $(1-p).p_k$.

Lập luận tương tự áp dụng cho cột.

Ví dụ. Xét ma trận: $A = \begin{pmatrix} 0 & 4 & 6 \\ 5 & 7 & 4 \\ 9 & 6 & 3 \end{pmatrix}$.

Cột 2 bị chi phối bởi tổ hợp các cột 1 và 3 với tần suất $\frac{1}{2}$ cho mỗi cột nên có thể xoá cột 2 vì thế $A = \begin{pmatrix} 0 & 6 \\ 5 & 4 \\ 9 & 3 \end{pmatrix}$. Tiếp theo, hàng 2 bị chi phối bởi tổ hợp hàng 1 với tần suất $\frac{1}{3}$ và hàng 3 với tần suất $\frac{2}{3}$ nên có thể xoá hàng 2. Ma trận được giản ước về dạng $A=\begin{pmatrix}0 & 6\\ 9 & 3\end{pmatrix}$. Đến đây dễ dàng tìm được giá trị trò chơi là $v=\frac{9}{2}$. Tương tự có thể tổ hợp của hơn 2 hàng (hoặc cột) sẽ chi phối một hàng (hoặc cột) khác.

Ví dụ. Xét ma trận $A=\begin{pmatrix}1 & 3 & 5 & 3\\ 4 & 0 & 2 & 2\\ 3 & 7 & 3 & 5\end{pmatrix}$.

Tổ hợp các cột 1, 2 và 3 với các tần suất $\frac{1}{3}$ cho mỗi cột là chi phối cột 4, do đó có thể xoá cột 4.

Lưu ý. Không phải mọi trò chơi ma trận đều có thể rút gọn bằng luật chi phối. Ví dụ, ma trận có điểm yên ngựa thì không có hàng hoặc cột nào bị chi phối.

e) Giải các trò chơi ma trận 2×n và m×2

Lời giải các trò chơi ma trận dạng $2\times n$ *và* $m\times 2$ có thể minh hoạ bằng đồ thị.

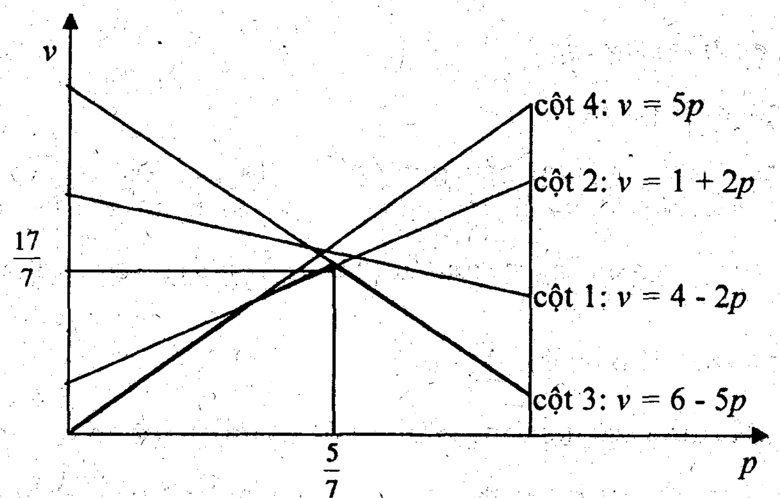
Xét trò chơi với ma trận $A=\begin{pmatrix}2 & 3 & 1 & 5\\ 4 & 1 & 6 & 0\end{pmatrix}$.

Giả sử người thứ nhất chọn hàng 1 với tần suất p và chọn hàng 2 với tần suất $1-p$. Khi đó: nếu người thứ hai chọn cột 1 thì giá trị trả cho người thứ nhất là

$$2p+4(1-p)=4-2p.$$

Nếu người thứ hai lần lượt chọn cột 2, 3, 4 thì giá trị trả cho người thứ nhất tương ứng là: $3p+(1-p)=1+2p$, $p+6(1-p)=6-5p$ và $5p$.

Xét các đồ thị của $v=4-2p$, $v=1+2p$, $v=6-5p$ và $v=5p$ trên $[0;1]$.



Hình 9.5

Với mỗi giá trị xác định của p, người thứ nhất có thể nhận được một giá trị là giá trị nhỏ nhất trong bốn giá trị của các hàm v nêu trên tương ứng với p, khi đó không phụ thuộc vào cách chọn của người thứ hai. Giá trị nhỏ nhất này là giá trị của một hàm có đồ thị là đường đậm chạy trên các đường thấp nhất trong bốn đường trên. Điểm cao nhất trên đường đậm này có hoành độ $p = \frac{5}{7}$. Điều đó có nghĩa là khi chọn hàng 1 với tần suất $p = \frac{5}{7}$ thì người thứ nhất nhận được giá trị lớn nhất trong các giá trị tối thiểu có thể phải nhận, giá trị này không phụ thuộc vào cách chọn của người thứ hai. Giá trị lớn nhất đó là $v_{max} = \frac{17}{7}$.

Qua đồ thị, chúng ta cũng nhận thấy giá trị $p = \frac{5}{7}$ chỉ là hoành độ giao điểm của cột 2 và 3, nghĩa là cột 1 không tham gia vào tìm p. Điều này có thể lí giải vì cột 1 bị chi phối bởi các cột 2 và 3 với cùng tỉ lệ $\frac{1}{2}$. Ngoài ra còn thấy $p = \frac{5}{7}$ chỉ là hoành độ giao điểm của hai đường ứng với cột 2 và 3 nên chỉ cần xét ma trận: $\begin{pmatrix} 3 & 1 \\ 1 & 6 \end{pmatrix}$ (mặc dù đồ thị của cột 4 có tham gia vào đường đậm, từ đó cũng có thể tính được $p = \frac{5}{7}$, $q = \frac{5}{7}$ và $v = \frac{17}{7}$. Vậy chiến thuật tối ưu của người thứ nhất là $\left(\frac{5}{7}; \frac{2}{7}\right)$ và chiến thuật tối ưu của người thứ hai là $\left(0; \frac{5}{7}; \frac{2}{7}; 0\right)$.

3. *Nguyên lí trung lập*

a) Tính giá trị trò chơi

Cho trò chơi ma trận với ma trận lượng giá là $A(a_{ij})_{m\times n}$. Nếu người thứ hai dùng chiến thuật hỗn hợp tối ưu là $p = (p_1, p_2, ..., p_m)$ và người thứ hai sử dụng cột j thì giá trị trung bình trả cho người thứ nhất là $\sum_{i=1}^{m} p_i a_{ij}$. Nếu V là giá trị trò chơi trong chiến thuật tối ưu p thì: $\sum_{i=1}^{m} p_i a_{ij} \geq V \quad \forall j = 1, 2, ..., n$ (1).

Tương tự khi người thứ hai dùng chiến thuật tối ưu $q = (q_1, q_2, ..., q_n)$, người thứ nhất chọn dòng i thì giá trị trò chơi mà người thứ hai phải trả là $\sum_{j=1}^{n} a_{ij} q_j$ thoả mãn:

$$\sum_{j=1}^{n} a_{ij}q_j \le V \qquad \forall i = 1,2,\ldots,m. \tag{2}$$

Khi cả hai người cùng dùng chiến thuật tối ưu thì :

$$V = V\sum_{j=1}^{n} q_j = \sum_{j=1}^{n} Vq_j \le \sum_{j=1}^{n}\left(\sum_{i=1}^{m} p_i a_{ij}\right) q_j = \sum_{i=1}^{m}\sum_{j=1}^{n} p_i a_{ij} q_j$$

$$= \sum_{i=1}^{m} p_i \left(\sum_{j=1}^{n} a_{ij} q_j\right) \le \sum_{i=1}^{m} p_i V = V.$$

Suy ra $V = \sum_{i=1}^{m}\sum_{j=1}^{n} p_i a_{ij} q_j$ (3) do đó V là phần tử duy nhất của ma trận 1×1: $p^T Aq$.

Trong nhiều công thức tính toán ma trận, để đơn giản kí hiệu, quy ước phần tử duy nhất của ma trận 1×1 thay cho chính ma trận. Do đó :

$$V = p^T Aq = \begin{pmatrix} p_1 & p_2 & \cdots & p_m \end{pmatrix} \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix}$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{n} p_i a_{ij} q_j \qquad a_{ij} = A(x_i, y_j).$$

b) Định lí cân bằng

Nguyên lí trung lập được thể hiện qua định lí sau đây gọi là định lí cân bằng:

Định lí

Cho trò chơi ma trận với ma trận lượng giá là $A(a_{ij})_{m\times n}$ nếu người thứ nhất dùng chiến thuật hỗn hợp tối ưu $p = (p_1, p_2, \ldots, p_m)$ và người thứ hai dùng chiến thuật hỗn hợp tối ưu $q = (q_1, q_2, \ldots, q_n)$, giá trị trò chơi đạt được là V thì:

$$\sum_{i=1}^{m} p_i a_{ij} = V \text{ với mọi cột } j \text{ có } q_j>0 \tag{4}$$

$$\text{và } \sum_{j=1}^{n} a_{ij} q_j = V \text{ với mọi dòng } i \text{ có } p_i>0 \tag{5}$$

Chứng minh

Ta chứng minh (5): Giả sử có dòng k với $p_k > 0$ mà $V \neq \sum_{j=1}^{n} a_{kj}q_j$ thì theo (2) có:

$V > \sum_{j=1}^{n} a_{kj}q_j$ (6). Suy ra :

$$V = \sum_{i=1}^{m}\sum_{j=1}^{n} p_i a_{ij} q_j = \sum_{i=1}^{m} p_i\left(\sum_{j=1}^{n} a_{ij}q_j\right) = p_1\sum_{j=1}^{n} a_{1j}q_j + p_k\sum_{j=1}^{n} a_{kj}q_j + p_m\sum_{j=1}^{n} a_{mj}q_j$$

$$< p_1V + \ldots + p_kV + \ldots + p_mV = V\sum_{i=1}^{m} p_i = V.$$

Dẫn tới $V < V$ là điều vô lí. Vậy (5) đã được chứng minh.

Chứng minh (4) tương tự.

Nhờ định lí cân bằng, khi biết chiến thuật tối ưu của một người thì việc tìm chiến thuật tối ưu cho người kia sẽ nhanh chóng hơn.

Ví dụ 1. Xét trò chơi Chẵn-Lẻ, hai người chơi đồng thời gọi ra một trong các số 0, 1 hoặc 2. Ma trận trò chơi là: $A = \begin{pmatrix} 0 & 1 & -2 \\ 1 & -2 & 3 \\ -2 & 3 & -4 \end{pmatrix}$ và bảng thể hiện điểm dành cho người Lẻ:

		Người Chẵn		
		0	1	2
Người Lẻ	0	0	1	–2
	1	1	–2	3
	2	–2	3	–4

Áp dụng định lí cân bằng, gọi $p = (p_1, p_2, p_3)$ là chiến thuật tối ưu của người Lẻ, $q = (q_1, q_2, \ldots, q_3)$ là chiến thuật tối ưu của người Chẵn và V là giá trị của bài toán, giả sử các q_i đều dương thì có hệ :

$$\text{(I)}\ \begin{cases} p_2 - 2p_3 = V \\ p_1 - 2p_2 + 3p_3 = V \\ -2p_1 + 3p_2 - 4p_3 = V \\ p_1 + p_2 + p_3 = 1. \end{cases}$$

Giải hệ này tìm được vectơ $p = \left(\frac{1}{4}; \frac{1}{2}; \frac{1}{4}\right)$ và $V = 0$.

Tìm chiến thuật tối ưu $q = (q_1, q_2, q_3)$ cho người Chẵn (người thứ hai): do $p_i > 0$ với mọi $i = 1, 2, 3$ nên có hệ gồm ba phương trình $\sum_{j=1}^{n} a_{ij} q_j = V$ và $q_1 + q_2 + q_3 = 1$. Nhưng do ma trận A là ma trận đối xứng nên hệ này tương tự hệ (I) trong đó thay p_i bởi q_i suy ra $q = \left(\frac{1}{4}; \frac{1}{2}; \frac{1}{4}\right)$. Nhận thấy các q_i đều dương nên chấp nhận được hệ (I) và suy ra các kết quả tìm thấy là chấp nhận được.

Ví dụ 2. Trò chơi trên ma trận vuông $m \times m$ có các phần tử trên đường chéo chính là các số dương $d_1, d_2, ..., d_m$, các phần tử còn lại là 0 :

$$A = \begin{pmatrix} d_1 & 0 & \dots & 0 \\ 0 & d_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & d_m \end{pmatrix}.$$

Áp dụng định lí cân bằng có thể giải trò chơi này như sau : Giả sử chiến thuật tối ưu hỗn hợp của người thứ nhất là $p = (p_1, p_2, \dots, p_m)$, của người thứ hai là $q = (q_1, q_2, ..., q_m)$. Nếu q_i đều dương, chúng ta có hệ : $p_i d_i = V$ với mọi $i = 1, ..., m$. Suy ra $p_i = \frac{V}{d_i}$ (a) với mọi $i = 1, ..., m$. Cộng từng vế các đẳng thức của hệ (a) ta có:

$$\sum_{i=1}^{m} p_i = \sum_{i=1}^{m} \frac{V}{d_i} \Rightarrow 1 = V \sum_{i=1}^{m} \frac{1}{d_i} \Rightarrow V = \left(\sum_{i=1}^{m} \frac{1}{d_i}\right)^{-1}.$$

Do ma trận đối xứng nên suy ra $q_j = \frac{V}{d_j}$ (b) với $j = 1, ..., m$.

Do $V > 0$, $p_i > 0$ với mọi $i = 1, ..., m$ và $q_j > 0$ với mọi $j = 1, ..., m$ nên chấp nhận được các kết quả này.

Ví dụ 3. Trò chơi ma trận tam giác. Trong trò chơi này ma trận trò chơi có dạng: các phần tử ở phía trên (hoặc dưới) đường chéo chính đều bằng 0. Xét một dạng cụ thể:

$$T = \begin{pmatrix} 1 & -2 & 3 & -4 \\ 0 & 1 & -2 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Nếu q_i đều dương, ta có hệ (4) $\sum_{i=1}^{m} p_i a_{ij} = V \qquad j = 1, ..., m.$

Cụ thể là : $\begin{cases} p_1 & = V \\ -2p_1 + p_2 & = V \\ 3p_1 - 2p_2 + p_3 & = V \\ -4p_1 + 3p_2 - 2p_3 + p_4 & = V. \end{cases}$

Suy ra $p_1 = V; p_2 = 3V; p_3 = 4V; p_4 = 4V$. Do đó: $1 = p_1 + p_2 + p_3 + p_4 = 12V$ và từ đó dẫn tới: $V = \frac{1}{12}$ và $p = \left(\frac{1}{12}; \frac{1}{4}; \frac{1}{3}; \frac{1}{3}\right)$ có thể là chiến thuật tối ưu cho người thứ nhất. Do các p_i đều dương nên tương tự từ hệ $\sum_{j=1}^{n} a_{ij} q_j = V$ (5) cũng suy ra $V = \frac{1}{12}$; $q = \left(\frac{1}{3}; \frac{1}{3}; \frac{1}{4}; \frac{1}{12}\right)$ có thể là chiến thuật tối ưu cho người thứ hai. Do trong kết quả các p_i và q_i đều dương nên kết quả này được chấp nhận.

Ví dụ 4. Trò chơi trên ma trận đối xứng bù. Đó là trò chơi hạn chế, có ma trận lượng giá là ma trận vuông $m \times m$ và bảo đảm $a_{ij} = -a_{ji}$ với mọi $i, j = 1, 2, ..., m$. Trò chơi *Giấy-Kéo-Búa* là điển hình của loại này.

Trong trò chơi Giấy-Kéo-Búa, người thứ nhất và người thứ hai đồng thời chọn và hiện ra một trong ba vật: giấy, kéo hoặc búa. Nếu hai người chọn vật như nhau thì giá trị trả về là 0. Nếu hai vật chọn là khác nhau thì : Kéo thắng Giấy, Búa thắng Kéo, Giấy thắng Búa. Nếu giá trị trả về cho người thắng là 1, có ma trận lượng giá như sau:

		Người thứ hai		
		Giấy	Kéo	Búa
Người thứ nhất	Giấy	0	–1	1
	Kéo	1	0	–1
	Búa	–1	1	0

Ma trận này đối xứng bù, nên trò chơi thuộc loại đối xứng. Các phần tử trên đường chéo chính bằng 0 (điều này cũng đúng cho một ma trận đối xứng bù bất kì vì $a_{ii} = -a_{ii}$ nên $a_{ii} = 0$ với mọi $i = 1, ..., m$).

Dễ dàng giải các hệ (4) và (5) được kết quả: $p = \left(\frac{1}{3}; \frac{1}{3}; \frac{1}{3}\right) = q$ và giá trị trò chơi là $V = 0$.

Có thể chứng minh tổng quát: Trong trò chơi hạn chế có ma trận đối xứng bù thì giá trị trò chơi bằng 0. Một chiến thuật tối ưu của người này cũng là chiến thuật tối ưu của người kia.

c) Các bất biến

Định nghĩa 1

Giả sử $G = (X, Y, A)$ là một trò chơi có giới hạn và g là một phép biến đổi 1-1 của Y lên Y. Trò chơi G là bất biến dưới tác động của g nếu với mỗi $x \in X$, thì có duy nhất một $x' \in X$ sao cho: $A(x, y) = A(x', g(y))$ với mọi $y \in Y$.

Ví dụ. Xét trò chơi $G = (X, Y, A)$ trong đó mỗi chiến thuật nguyên thuỷ trong X và Y là chọn một cặp số như trong bảng sau :

	Các chiến thuật nguyên thuỷ Y của người thứ hai				
		(3,0)	(2,1)	(1,2)	(0,3)
Chiến thuật nguyên thuỷ X của người thứ nhất	(4,0)	4	2	1	0
	(3,1)	1	3	0	–1
	(2,2)	–2	2	2	–2
	(1,3)	–1	0	3	1
	(0,4)	0	1	2	4

Tập $Y = \{(3, 0), (2, 1), (1, 2), (0, 3)\}$, phép biến đổi g được định nghĩa như sau:

$$g((3, 0)) = (0, 3);\ g((0, 3)) = (3, 0);\ g((2, 1)) = (1, 2);\ g((1, 2)) = (2, 1).$$

Dễ thấy g là ánh xạ 1-1 của Y lên Y và trò chơi G là bất biến dưới tác động của g. Ta có thể kiểm tra điều này, chẳng hạn $x = (4, 0)$, $y = (0, 3)$ thì $A(x, y) = 0$ và có duy nhất $x' = (0, 4)$, $g(y) = (3, 0)$ mà $A(x', g(y)) = A(x, y)$. Tương tự có thể kiểm tra với mọi giá trị khác của x và y.

Bổ đề 1

> *Nếu trò chơi hữu hạn $G=(X, Y, A)$ là bất biến dưới hai phép biến đổi 1-1 là g_1 và g_2 thì G cũng bất biến dưới phép biến đổi $g_2.g_1$ được xác định sao cho $g_2.g_1(y)=g_2(g_1(y))$.*

Bổ đề 2

> *Nếu trò chơi hữu hạn $G=(X, Y, A)$ bất biến dưới phép biến đổi 1-1 g thì G cũng bất biến dưới phép biến đổi g^{-1} (g^{-1} là phép nghịch đảo của g được định nghĩa: $g.g^{-1}= g^{-1}.g=e$ ($e(y)=y$ với mọi $y\in Y$).*

Do đó các phép biến đổi g lên Y làm trò chơi bất biến đã tạo thành một nhóm $\wp$ gồm tổ hợp nhân các phép biến đổi. Phần tử đơn vị của nhóm là phép biến đổi đồng nhất e.

Tập $\overline{\wp}$ gồm các phép biến đổi $\overline{g}$ tương ứng trên X cũng tạo thành một nhóm với phần tử đồng nhất là $\overline{e}$ mà $\overline{e}(x) = x$ với mọi $x\in X$.

Định nghĩa 2

Một trò chơi hữu hạn $G = (X, Y, A)$ là bất biến trên nhóm các phép biến đổi $\wp$ nếu nó bất biến với mọi phép biến đổi g của $\wp$, nghĩa là :

$A(x, y) = A(\overline{g}(x), g(y))$ với mọi $x \in X$ và với mọi $y \in Y$ xảy ra với mọi $g\in \wp$.

Định nghĩa 3

Trong trò chơi hữu hạn $G = (X, Y, A)$ bất biến trên nhóm $\wp$ (nhóm các phép biến đổi 1-1 của Y lên Y), một chiến thuật hỗn hợp $q = (q(1), q(2), ..., q(n))$ cho người thứ hai được gọi là *chiến thuật bất biến* dưới nhóm $\wp$ nếu: $q(g(y)) = q(y)$ với mọi $y \in Y$ và với mọi $g \in \wp$.

Định lí

> *Nếu trò chơi hữu hạn $G=(X, Y, A)$ bất biến dưới nhóm $\wp$ thì tồn tại một chiến thuật bất biến tối ưu cho các người chơi.*

Ví dụ 1. Trò chơi ghép xu

Hai người chơi đồng thời chọn và hiện một mặt của một đồng xu đang cầm (đồng xu có hai mặt: sấp và ngửa). Người thứ nhất thắng nếu hai mặt xu của hai người giống nhau, ngược lại thì người thứ hai thắng. Ma trận lượng giá là $A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$. Trong trò chơi ghép xu, $X = Y = \{1, 2\}$ và $A(1, 1) = A(2, 2) = 1$ còn $A(1, 2) = A(2, 1) = -1$.

	Người thứ hai		
Người thứ nhất		Ngửa	Sấp
	Ngửa	1	−1
	Sấp	−1	1

Trò chơi ghép xu $G(X, Y, A)$ là bất biến dưới nhóm $\wp = \{e, g\}$, trong đó e là phép đồng nhất còn g là phép biến đổi sao cho $g(1) = 2$; $g(2) = 1$. Chiến thuật hỗn hợp $(q(1), q(2))$ là bất biến dưới nhóm $\wp$ nếu $q(1) = q(g(1)) = q(2)$. Do $q(1) + q(2) = 1$ nên đã gợi ý rằng $q(1) = q(2) = \frac{1}{2}$ là một chiến thuật cho người thứ hai.

Tương tự $p(1) = p(2) = \frac{1}{2}$ là một bất biến duy nhất và do đó cũng là chiến thuật cho người thứ nhất.

Ví dụ 2. Trò chơi Giấy-Kéo-Búa là bất biến dưới nhóm $\wp = \{e, g, g^2\}$ với g(giấy) = kéo, g(kéo) = búa, g(búa) = giấy.

Do đó: q(giấy) = $q(g$(giấy)) = q(kéo) = $q(g$(kéo)) = q(búa). Vậy chiến thuật tối ưu có tần suất $\frac{1}{3}$ cho mỗi loại giấy, búa và kéo.

Ví dụ 3. Trò chơi Colonel Blotto

Phát biểu. Người chơi thứ nhất là đại tá Colonel Blotto có bốn trung đoàn cần chiếm hai trạm. Người chơi thứ hai là đại tá Lieutenant Kije có ba trung đoàn cũng muốn chiếm hai trạm này. Mỗi bên đều muốn gửi nhiều trung đoàn nhất để chiếm một trạm và các trung đoàn còn lại dùng để chiếm trạm kia. Giá trị trả về cho người chơi như sau: sẽ được một điểm khi chiếm được một trạm và một điểm khi bắt được một trung đoàn của đối phương (khi số quân gửi đến đông hơn sẽ bắt được toàn bộ các trung đoàn của đối phương). Nếu hai bên gửi cùng một số lượng trung đoàn tới một trạm thì cả hai cùng phải rút lui và không được điểm nào. Colonel Blotto cần quyết định phân bổ lực lượng của mình như thế nào tới chiếm hai trạm?

Phân tích. Có năm chiến thuật thuần tuý mà Colonel Blotto có thể thực hiện là $X = \{(4, 0), (3, 1), (2, 2), (1, 3), (0, 4)\}$, ở đây (n_1, n_2) tương ứng với chiến thuật gửi n_1 trung đoàn đi chiếm trạm 1, gửi n_2 trung đoàn đi chiếm trạm 2.

Lieutenant Kije có bốn chiến thuật nguyên thuỷ, $Y = \{(3, 0), (2, 1), (1, 2), (0, 3)\}$

Ma trận giá trị trả về được thể hiện qua bảng sau:

	Chiến thuật nguyên thuỷ của Kije				
Chiến		(3,0)	(2,1)	(1,2)	(0,3)
thuật	(4,0)	4	2	1	0
nguyên	(3,1)	1	3	0	-1
thuỷ	(2,2)	-2	2	2	-2
của	(1,3)	-1	0	3	1
Blotto	(0,4)	0	1	2	4

Bảng a)

Tiếc rằng ma trận 5×4 này không có các dòng và cột bị chi phối. Tuy nhiên, có một bất biến trong bài toán này làm cho việc giải nó đơn giản đi khá nhiều. Điều này liên quan tới tính đối xứng giữa hai trạm cần chiếm. Xét nhóm $\wp=\{e, g\}$ mà $g((3, 0)) = (0, 3)$; $g((0, 3)) = (3, 0)$; $g((2, 1)) = (1, 2)$; $g((1, 2)) = (2, 1)$ và nhóm tương ứng $\bar{\wp}=\{\bar{e},\bar{g}\}$, trong đó $\bar{g}((4, 0)) = (0, 4)$; $\bar{g}((0, 4)) = (4, 0)$; $\bar{g}((3, 1)) = (1, 3)$; $\bar{g}((1, 3)) = (3, 1)$; $\bar{g}((2, 2)) = (2, 2)$.

Các nhóm tương đương cho Kije là $\{(3, 0), (0, 3)\}$ và $\{(2, 1), (1, 2)\}$. Do đó chiến thuật q là bất biến nếu $q((3, 0)) = q((0, 3))$ và $q((2, 1)) = q((1, 2))$.

Tương tự các nhóm tương đương của Blotto là $\{(4, 0), (0, 4)\}$, $\{(3, 1), (1, 3)\}$ và $\{2, 2\}$. Do đó chiến thuật p cho Blotto là bất biến nếu $p((4, 0)) = p((0, 4))$ và $p((3, 1)) = p((1, 3))$.

Chúng ta có thể giảm các chiến thuật của Kije thành hai phần tử :

- (3, 0)*: sử dụng các chiến thuật (3, 0) và (0, 3) với tần suất $\frac{1}{2}$ cho mỗi chiến thuật;
- (2, 1)*: sử dụng các chiến thuật (2, 1) và (1, 2) với tần suất $\frac{1}{2}$ cho mỗi chiến thuật;

Tương tự chúng ta có thể giảm các chiến thuật của Bloto thành ba phần tử :

- (4, 0)* : sử dụng các chiến thuật (4, 0) và (0, 4) với tần suất $\frac{1}{2}$ cho mỗi chiến thuật;
- (3, 1)*: sử dụng các chiến thuật (3, 1) và (1, 3) với tần suất $\frac{1}{2}$ cho mỗi chiến thuật;
- (2, 2) : khi sử dụng chiến thuật (2, 2).

Với các chiến thuật này, ma trận giá trị trả về được thể hiện qua bảng sau:

	(3, 0)*	(2, 1)*
(4, 0)*	2	1.5
(3, 1)*	0	1.5
(2, 2)	-2	2

Bảng b)

Chúng ta giải thích về giá trị của ô ở góc trên-trái : Nếu Blotto dùng các chiến thuật (4, 0) và (0, 4) với tần suất $\frac{1}{2}$ cho mỗi chiến thuật và Kije dùng các chiến thuật (3, 0) và (0, 3) cũng với tần suất $\frac{1}{2}$ cho mỗi chiến thuật thì bốn ô ở bốn góc của bảng a) (bảng giá trị trả về) xuất hiện với tần suất $\frac{1}{4}$ cho mỗi ô, do đó giá trị trả về cho ô ở góc trái-trên của bảng b) là trung bình cộng của bốn số: 4, 0, 0, 4 là 2. Giá trị tại các ô khác cũng tính tương tự.

Bây giờ chúng ta giải bài toán với bảng b). Chú ý dòng 2 có thể được giản ước bởi dòng 1. Sau đó giải trò chơi trên ma trận 2×2 dễ dàng suy ra kết quả. Chiến thuật hỗn hợp tối ưu cho Blotto là $\left(\frac{8}{9};0;\frac{1}{9}\right)$. Chiến thuật hỗn hợp tối ưu cho Kije là $\left(\frac{1}{9};\frac{8}{9}\right)$. Giá trị trò chơi là $\frac{14}{9}$.

Quay về ma trận cho ở bảng a), chúng ta tìm được chiến thuật tối ưu cho Blotto là $\left(\frac{4}{9};0;\frac{1}{9};0;\frac{4}{9}\right)$ và chiến thuật tối ưu cho Kije là $\left(\frac{1}{18};\frac{4}{9};\frac{4}{9};\frac{1}{18}\right)$. Giá trị trò chơi là $\frac{14}{9}$.

4. Giải trò chơi hạn chế

a) Chiến thuật Baye

Nếu người thứ nhất sử dụng chiến thuật p và người thứ hai sử dụng chiến thuật q thì giá trị trung bình trả về người thứ nhất sẽ là :

$$\sum_{i=1}^{m}\left(\sum_{j=1}^{n}a_{ij}q_j\right)p_i=\sum_{i=1}^{m}\sum_{j=1}^{n}p_ia_{ij}q_j=p^TAq.$$

Nếu người thứ nhất dự đoán người thứ hai đã chọn một chiến thuật cụ thể q, thì người thứ nhất sẽ chọn hàng i sao cho $\sum_{j=1}^{n}a_{ij}q_j$ đạt giá trị lớn nhất khi đó giá trị trung bình trả về cho người thứ nhất là $\max_{1\le i\le m}\left\{\sum_{j=1}^{n}a_{ij}q_j\right\}$, hoặc cách tương đương là người thứ nhất sẽ chọn chiến thuật p sao cho p^TAq đạt giá trị lớn nhất, giá trị trả về cho người thứ nhất sẽ là $\max_{p\in X^*}\left\{p^TAq\right\}$. Có thể chứng minh :

$$\max_{1\le i\le m}\left\{\sum_{j=1}^{n}a_{ij}q_j\right\}=\max_{p\in X^*}\left\{p^TAq\right\}.$$

Mỗi chiến thuật p để p^TAq đạt được giá trị lớn nhất được gọi là chiến thuật đối phó tốt nhất (hay chiến thuật Baye) của người thứ nhất chống lại chiến thuật q

của người thứ hai. Trong trò chơi hạn chế, luôn tồn tại chiến thuật nguyên thuỷ Baye chống lại mỗi chiến thuật q của người thứ hai.

Tương tự, nếu người thứ hai dự đoán người thứ nhất chơi chiến thuật cụ thể p thì người thứ hai sẽ chọn cột j sao cho $\sum_{i=1}^{m} p_i a_{ij}$ đạt giá trị nhỏ nhất hoặc cách tương đương là người thứ hai sẽ chọn chiến thuật q sao cho $p^T Aq$ đạt giá trị nhỏ nhất.

Tương tự có : $\underset{1\le j\le n}{Min}\left\{\sum_{i=1}^{m} p_i a_{ij}\right\} = \underset{q\in Y^*}{Min}\left\{p^T Aq\right\}$.

Mỗi chiến thuật q để $p^T Aq$ đạt giá trị nhỏ nhất được gọi là chiến thuật đối phó tốt nhất (hay là chiến thuật Baye) của người thứ hai chống lại chiến thuật p của người thứ nhất.

Sử dụng chiến thuật Baye là một trong các cách để chơi trò chơi. Tuy nhiên có thể sinh ra điều bất lợi (nguy hiểm) khi đối thủ chơi một chiến thuật tốt hơn chiến thuật mà bạn dự đoán đối thủ sẽ dùng.

b) Giá trị trên, giá trị dưới

Bây giờ giả sử trò chơi quy định : người thứ hai phải thông báo sự lựa chọn chiến thuật hỗn hợp q trước khi người thứ nhất lựa chọn chiến thuật của mình. Sự quy định này của trò chơi có lẽ tạo thuận lợi cho người thứ nhất. Nếu người thứ hai thông báo chọn q thì người thứ nhất sẽ chọn chiến thuật Baye chống lại q và người thứ hai sẽ thua trung bình một lượng là $\max_{p\in X^*}\left\{p^T Aq\right\}$. Tuy nhiên, nếu người thứ hai biết điều này thì người thứ hai sẽ chọn thông báo q sao cho $\max_{p\in X^*}\left\{p^T Aq\right\}$ đạt giá trị nhỏ nhất. Giá trị nhỏ nhất của $\max_{p\in X^*}\left\{p^T Aq\right\}$ với mọi q kí hiệu là $\overline{V}$ và được gọi là *giá trị trên* của trò chơi. Vậy :

$$\overline{V} = \underset{q\in Y^*}{\text{Min}}\left\{\underset{1\le i\le m}{\text{Max}}\sum_{j=1}^{n} a_{ij} q_j\right\} = \underset{q\,\in\, Y^*}{\text{Min}}\left\{\underset{p\in X^*}{\text{Max}}\left(p^T Aq\right)\right\}.$$

Giá trị trên $\overline{V}$ là sự thiệt hại trung bình nhỏ nhất của người thứ hai không phụ thuộc vào cách chọn chiến thuật của người thứ nhất. Một chiến thuật q để $\max_{p\in X^*}\left\{p^T Aq\right\}$ đạt giá trị nhỏ nhất được gọi là ***chiến thuật Minimax*** cho người thứ hai. Nó cực tiểu hoá sự thiệt hại tối đa của người thứ hai.

Phân tích tương tự cho trường hợp trò chơi quy định : người thứ nhất phải thông báo chiến thuật hỗn hợp mà người thứ nhất đã chọn là p trước khi người thứ hai lựa chọn chiến thuật của mình. Khi đó người thứ hai sẽ chọn q sao cho giá trị phải trả là $p^T Aq$ đạt giá trị nhỏ nhất. Nhưng người thứ nhất nếu biết tình huống này thì người thứ nhất sẽ chọn p sao cho $\underset{q\in Y^*}{\mathrm{Min}}\left(p^T Aq\right)$ đạt được giá trị lớn nhất, và giá trị trung bình người thứ nhất đạt được là

$$\underline{V} = \underset{p\in X^*}{\mathrm{Max}}\left\{\underset{1\le j\le n}{\mathrm{Min}}\left(\sum_{i=1}^{m} p_i a_{ij}\right)\right\} = \underset{p\in X^*}{\mathrm{Max}}\left\{\underset{q\in Y^*}{\mathrm{Min}}\left(p^T Aq\right)\right\}.$$

Đại lượng $\underline{V}$ gọi là giá trị dưới của trò chơi. Nó là giá trị lớn nhất người thứ nhất đạt được không phụ thuộc vào sự lựa chọn chiến thuật của người thứ hai. Một chiến thuật p sao cho $\underset{q\in Y^*}{\mathrm{Min}}\left(p^T Aq\right)$ đạt được giá trị lớn nhất gọi là ***chiến thuật Minimax*** cho người thứ nhất. Vậy trong trò chơi hạn chế, người thứ nhất cũng luôn có chiến thuật Minimax.

Lưu ý. Nếu quy luật chơi yêu cầu người chơi II cần thông báo một chiến thuật hỗn hợp của mình trước khi người thứ nhất chọn chiến thuật thì chưa hẳn đã là thuận lợi cho người thứ nhất vì người thứ hai có thể thông báo một chiến thuật Minimax.

Chúng ta thừa nhận những kết quả sau:

Định lí 1

Mỗi trò chơi hạn chế đều có giá trị trò chơi và cả hai người chơi đều có chiến thuật Minimax.

Định lí 2

Giá trị dưới thì nhỏ hơn hoặc bằng giá trị trên : $\underline{V} \le \overline{V}$.

- *Nếu* $\underline{V} < \overline{V}$ *thì giá trị trung bình trả về của trò chơi là* V *sẽ thoả mãn :* $\underline{V} \le V \le \overline{V}$.
- *Nếu* $\overline{V} = \underline{V}$ *thì tồn tại giá trị trò chơi, giá trị này kí hiệu là* V *và* $V = \overline{V} = \underline{V}$.
- *Nếu giá trị* V *tồn tại thì chiến thuật Minimax là một chiến thuật tối ưu.*

Định lí 3

> *Nếu $A = (a_{ij})$ và $A' = (a'_{ij})$ là các ma trận mà $a'_{ij} = c.a_{ij} + b$, với $c > 0$ thì chiến thuật Minimax cho người thứ nhất và người thứ hai trên ma trận A cũng là chiến thuật Minimax cho người thứ nhất và người thứ hai trên ma trận A'. Nếu V là giá trị trò chơi trên ma trận A thì giá trị trò chơi trên ma trận A' là $V' = cV + b$.*

c) Giản ước về bài toán lập trình tuyến tính

Bây giờ chúng ta hãy xem xét trò chơi trên quan điểm của người thứ nhất : Người thứ nhất muốn chọn $p = (p_1, p_2, \ldots, p_m)$ sao cho $\underset{1 \le j \le n}{\text{Min}} \left\{ \sum_{i=1}^{m} p_i a_{ij} \right\}$ đạt giá trị lớn nhất. Có thể coi đây là một bài toán : tìm $p_1, p_2, \ldots, p_m$ sao cho hàm mục tiêu $\underset{1 \le j \le n}{\text{Min}} \left\{ \sum_{i=1}^{m} p_i a_{ij} \right\}$ đạt giá trị lớn nhất với ràng buộc : $p_1 + p_2 + \ldots + p_m = 1$ và $p_i \ge 0$ với mọi $i = 1, 2, \ldots, m$. Có thể chuyển bài toán này về bài toán tuyến tính bằng cách thêm biến v thoả ràng buộc : $v \le \underset{1 \le j \le n}{\text{Min}} \left\{ \sum_{i=1}^{m} p_i a_{ij} \right\}$ và cố gắng cực đại v (càng lớn càng tốt trong ràng buộc mới này).

Bài toán đưa về: Chọn v và $p_1, p_2, \ldots, p_m$ sao cho v đạt giá trị lớn nhất với các ràng buộc: $v \le \sum_{i=1}^{m} p_i a_{i1}$; $v \le \sum_{i=1}^{m} p_i a_{i2}$; ...; $v \le \sum_{i=1}^{m} p_i a_{in}$; $p_1 + p_2 + \ldots + p_m = 1$; $p_i \ge 0$ với mọi $i = 1, 2, \ldots, m$.

Đây là một bài toán tuyến tính. Để giải bài toán này chúng ta có thể giải bằng một thuật toán đơn giản là thuật toán đơn hình.

Tương tự, xét bài toán trên quan điểm của người thứ hai cũng dẫn tới bài toán tuyến tính tương tự. Đó là bài toán sau :

Chọn w và $q_1, q_2, \ldots, q_n$ sao cho w đạt giá trị nhỏ nhất với các ràng buộc :

$w \ge \sum_{j=1}^{n} a_{1j} q_j$; $w \ge \sum_{j=1}^{n} a_{2j} q_j$; ... ; $w \ge \sum_{j=1}^{n} a_{mj} q_j$; $q_1 + q_2 + \ldots + q_m = 1$; $q_j \ge 0$ với mọi $j = 1, 2, \ldots, n$.

Một cách khác chuyển bài toán về dạng có thể lập trình tuyến tính là : Giả sử $v > 0$, đặt $x_i = \frac{p_i}{v}$ thì ràng buộc $p_1 + p_2 + \ldots + p_m = 1$ thành $x_1 + x_2 + \ldots + x_m = \frac{1}{v}$. Tìm v đạt giá trị lớn nhất tương đương $\frac{1}{v}$ đạt giá trị nhỏ nhất. Chúng ta loại v khỏi bài toán và có bài toán mới sau :

Chọn $x_1, x_2, \ldots, x_m$ sao cho $x_1 + x_2 + \ldots + x_m$ đạt giá trị nhỏ nhất với các ràng buộc : $1 \le \sum_{i=1}^{m} x_i a_{i1}$; $1 \le \sum_{i=1}^{m} x_i a_{i2}$; …; $1 \le \sum_{i=1}^{m} x_i a_{in}$ và $x_i \ge 0$ với mọi $i = 1, 2, \ldots, m$.

Sau khi giải xong bài toán này, lời giải của bài toán chuẩn là : $v = \dfrac{1}{x_1 + x_2 + \ldots + x_m}$ và chiến thuật tối ưu cho người thứ nhất sẽ là $p_i = v.x_i$ với mọi $i = 1, 2, \ldots, m$.

d) Thuật toán đơn hình để giải trò chơi hạn chế

Thuật toán đơn hình giải trò chơi hạn chế được dùng khi ma trận trò chơi không có điểm yên ngựa và cũng không còn thu gọn được bằng luật chi phối. Các bước thực hiện thuật toán như sau:

Bước 1. Cộng thêm một hằng số vào các phần tử của ma trận trò chơi để mọi phần tử đều dương. Cần ghi nhớ số này để cuối cùng sẽ trừ giá trị tìm được cho số này để có giá trị trò chơi trên ma trận ban đầu.

Bước 2. Tạo bảng có gia số -1 ở các ô thuộc dòng cuối (dòng $m + 1$), và gia số $+1$ ở các ô thuộc cột phải (cột $n + 1$), gia số 0 ở góc phải-dưới (dòng $m + 1$, cột $n + 1$).

	y_1	y_2	…	y_n	
x_1	a_{11}	a_{11}	…	a_{11}	1
x_2	a_{21}	a_{22}	…	a_{2n}	1
⋮	⋮	⋮		⋮	⋮
x_m	a_{m1}	a_{m1}	…	a_{mn}	1
	-1	-1	…	-1	0

Bước 3. Chọn một ô bên trong bảng làm điểm chốt, giả sử đó là ô thuộc dòng p cột q (p gọi là dòng chốt, q gọi là cột chốt, số tại điểm chốt gọi là số chốt), có những đặc điểm sau:

+ Gia số ở cột chốt là $A[m + 1 ; q]$ phải là số âm;

+ Số tại điểm chốt là $A[p ; q]$ phải dương;

+ Tỉ số giữa gia số trên dòng chốt và số chốt là $\dfrac{A[p;n+1]}{A[p;q]}$ phải nhỏ nhất với mọi số dương $A[p ; q]$ thuộc cột q.

Bước 4. Thay đổi giá trị các phần tử của bảng như sau :

+ Thay số tại các ô $A[i ; j]$ ($i \neq p$ và $j \neq q$) bởi $A[i;j]-\dfrac{A[p;j].A[i;q]}{A[p;q]}$.

+ Thay các số tại các ô thuộc dòng p (trừ ô chốt) bởi thương giữa số tại ô đó và số tại ô chốt.

+ Thay các số tại các ô thuộc cột q (trừ ô chốt) bởi số đối của thương giữa số tại ô đó và số tại ô chốt.

+ Thay số tại ô chốt bởi nghịch đảo của nó.

Có thể minh hoạ như sau :

		q		$j \neq q$	
p		x		y	
$i \neq p$		t		z	
Trước bước 4					

→

		q		$j \neq q$	
p		$\dfrac{1}{x}$		$\dfrac{y}{x}$	
$i \neq p$		$-\dfrac{t}{x}$		$\dfrac{zx-ty}{x}$	
Sau bước 4					

Bước 5. Thay nhãn của cột chốt và nhãn của dòng chốt cho nhau (nghĩa là p và q đổi chỗ cho nhau).

Bước 6. Nếu còn số âm trên dòng $m+1$ thì quay về bước 3.

Bước 7. Ngược lại, đưa kết quả ra:

a) Giá trị trò chơi v là nghịch đảo của số ở góc phải-dưới. Nếu ở bước 1 đã trừ mỗi ô cho một hằng số (để dễ tính toán) thì phải cộng thêm hằng số này vào v. Ngược lại, nếu ở bước 1 đã cộng vào mỗi ô một hằng số (để các số đều dương) thì ở bước này phải trừ v cho hằng số này.

b) Chiến thuật tối ưu của người thứ nhất được xây dựng như sau : Tại thời điểm cuối cùng, các biến của người thứ nhất còn lại trên cột trái sẽ có tần suất 0, các biến của người thứ nhất nằm ở trên dòng trên cùng của bảng sẽ nhận giá trị tại dòng cuối cùng thuộc cùng cột chia cho giá trị tại góc phải-dưới.

c) Chiến thuật tối ưu của người thứ hai được xây dựng như sau : Tại thời điểm cuối cùng, các biến của người thứ hai còn lại thuộc dòng trên cùng sẽ có tần suất 0, các biến của người thứ hai thuộc cột bên trái của bảng sẽ nhận giá trị tại cột bên phải cùng dòng chia cho giá trị tại góc phải-dưới.

Ví dụ. Xét ma trận trò chơi:

$$A=\begin{pmatrix} 2 & -1 & 6 \\ 0 & 1 & -1 \\ -2 & 2 & 1 \end{pmatrix}.$$

Ma trận này có phần tử nhỏ nhất là -2, do đó cần thực hiện bước 1 chuyển về ma trận : $A'=\begin{pmatrix} 4 & 1 & 8 \\ 2 & 3 & 1 \\ 0 & 4 & 3 \end{pmatrix}$, ma trận này không có điểm yên ngựa và cũng không thu gọn được bằng luật chi phối.

Thực hiện bước 2 có bảng sau :

	y_1	y_2	y_3	
x_1	4	1	8	1
x_2	2	3	1	1
x_3	0	4	3	1
	-1	-1	-1	0

Thực hiện bước 3 chọn được ô chốt (**thuộc một** cột tuỳ ý vì các số thuộc dòng cuối cùng của bảng tại các cột y_1, y_2 **và** y_3 **đều** âm), còn dòng của ô chốt phải theo quy luật đã nêu trên. Kết quả **chọn được** ô chốt là ô $(x_1 ; y_1)$.

Thực hiện bước 4 và 5 có bảng sau :

	y_1	y_2	y_3	
x_1	$\frac{1}{4}$	$\frac{1}{4}$	2	$\frac{1}{4}$
x_2	$-\frac{1}{2}$	$\frac{5}{2}$	-3	$\frac{1}{2}$
x_3	0	4	3	1
	$\frac{1}{4}$	$-\frac{3}{4}$	1	$\frac{1}{4}$

Thực hiện bước 6 thấy phải quay về bước 3 chọn ô chốt mới là $(x_2 ; y_2)$.

Thực hiện bước 4 và 5 có bảng sau:

	y_1	y_2	y_3	
x_1	0.3	-0.1	2.3	0.2
x_2	-0.2	0.4	-1.2	0.2
x_3	0.8	-1.6	7.8	0.2
	0.1	0.3	0.1	0.4

Thực hiện bước 6, chuyển sang bước 7:

Giá trị trò chơi là $v = \frac{1}{0.4} - 2 = \frac{5}{2} - 2 = \frac{1}{2}$ (phải trừ đi 2 vì ban đầu các phần tử của ma trận được cộng thêm 2).

Chiến thuật tối ưu cho người thứ nhất là : $p = (0.25 ; 0.75 ; 0)$.

Chiến thuật tối ưu cho người thứ hai là $q = (0.5 ; 0.5 ; 0)$.

5. *Dạng mở rộng của trò chơi*

Dạng chiến thuật của trò chơi là một cách chuẩn mực để diễn tả mô hình toán học của trò chơi. Tuy nhiên, có một số điều thú vị khác của trò chơi lại bị bỏ qua trong mô hình đơn giản này. Do đó, bây giờ chúng ta đề cập tới một mô hình toán học khác bàn đến các khái niệm cơ sở chưa được đề cập trong dạng chiến thuật trò chơi. Mô hình mới này gọi là dạng mở rộng của trò chơi. Ở đây đề cập tới ba khái niệm cơ sở: cây trò chơi, bước đi ngẫu nhiên và tập thông tin về trò chơi.

a) Cây trò chơi

Dạng mở rộng của trò chơi là mô hình dùng đồ thị có hướng để mô tả trò chơi. Đồ thị có hướng là cặp (T, F) trong đó T là tập không rỗng các đỉnh, F là một hàm sao cho với mỗi $x \in T$ có tập con $F(x) \in T$ (gọi $F(x)$ là tập các đỉnh theo sau x). Khi dùng đồ thị có hướng để biểu diễn trò chơi thì tập đỉnh biểu diễn các vị trí của trò chơi, $F(x)$ là tập các đỉnh có thể tới từ x sau một phép di chuyển (một bước đi hay là một lượt đi của một người chơi). Đường đi từ t_0 đến t_1 là dãy $t_0 = x_0, x_1, ..., x_n = t_1$ mà x_i là đỉnh theo sau của đỉnh x_{i-1} với $i = 1, ..., n$. Trong dạng mở rộng của trò chơi, chúng ta thường gặp một dạng riêng của đồ thị có hướng đó là dạng ***cây*** (hay còn gọi là cây trò chơi).

Định nghĩa

Cây là đồ thị có hướng (T, F) trong đó có một đỉnh đặc biệt t_0 gọi là gốc hoặc gọi là đỉnh khởi đầu, sao cho với mọi đỉnh $t \in T$ khác t_0 luôn có một đường đi duy nhất từ t_0 tới t. Đỉnh không có đường đi tới đỉnh khác gọi là lá của cây.

Trong dạng mở rộng của trò chơi, trò chơi được bắt đầu từ đỉnh gốc và tiếp tục theo các đường đi dần về lá khi người chơi lần lượt thực hiện các phép di chuyển của mình (lá của cây là các vị trí kết thúc trò chơi). Tại các lá, các luật chơi cho phép xác định giá trị trả về cho người chơi. Với trò chơi có n người chơi, tại lá sẽ có bộ n giá trị trả về.

Do chúng ta chỉ xét trò chơi hai người có tổng điểm bằng 0 nên giá trị trả về chỉ là một con số thể hiện điểm người thứ nhất thắng người thứ hai. Một số trong các đỉnh không là lá sẽ được gắn cho người thứ nhất có quyền chọn phép di chuyển từ đỉnh đó. Số đỉnh còn lại gắn cho người thứ hai có quyền chọn phép di chuyển từ đỉnh đó. Tuy nhiên cũng tồn tại một số đỉnh đơn lẻ có thể nhận các bước đi không biết trước là thuộc người nào (các đỉnh này thuộc bước đi ngẫu nhiên).

Bước đi ngẫu nhiên. Nhiều trò chơi chứa các bước đi ngẫu nhiên. Chẳng hạn, gieo xu, quay bánh xe trong trò chơi may rủi, quay chiếc lồng lăn bóng trong trò chơi xổ số lôtô... Trong những trò chơi này, những bước đi ngẫu nhiên có vai trò quan trọng. Ngay trong chơi cờ cũng có một bước đi ngẫu nhiên là phép gieo xu hoặc súc sắc để xác định người chơi nào được đi trước và do đó bước đi đầu tiên có thể tạo ra những lợi thế nào đó. Trong trò chơi bài, bước đi đầu tiên cũng là một bước đi ngẫu nhiên, đó là bước "tráo bài và chia bài", mỗi người chơi chỉ nhận biết được một phần kết quả của bước đi đầu tiên này (là những quân bài mà họ được nhận) nhưng họ không biết đầy đủ thông tin về kết quả này (những quân bài đối thủ nhận được là gì). Bước đi ngẫu nhiên như là đã tạo cho mỗi người chơi một tần suất khác nhau để nhận được một kết quả nhất định trong bước đi này. Kí hiệu bước đi ngẫu nhiên là N.

Thông tin. Một khía cạnh quan trọng khác khi chúng ta nghiên cứu dạng mở rộng của trò chơi đó là thông tin cho người chơi biết về thứ tự các bước di chuyển.

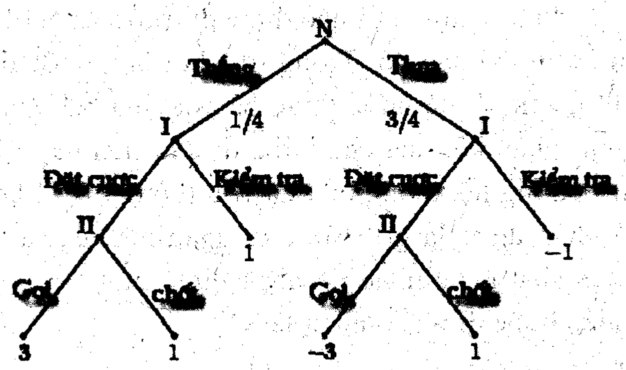
b) Ván cơ bản kết thúc trò chơi bài

Một trong các trạng thái của trò chơi bài có mô hình toán học đơn giản và hữu ích nhất là "đánh cược cổ điển" hoặc còn gọi là "ván cơ bản kết thúc trò chơi bài". Ván cơ bản kết thúc trò chơi bài như sau: Cả hai người đặt 1 đồng tiền cược đầu tiên vào giữa bàn. Số tiền giữa bàn bây giờ là 2 đồng (khi số tiền giữa bàn có từ 2 đồng trở lên được gọi là một bình tiền). Người thứ nhất rút một quân bài từ cỗ bài. Giả sử nó có thể là quân bài chiến thắng với tần suất $\frac{1}{4}$ hoặc có thể là quân bài thua với tần suất $\frac{3}{4}$ (quân bài rút ra vẫn úp). Sau đó, người thứ nhất sẽ chọn *kiểm tra* hoặc *đặt cược* :

- Nếu chọn *kiểm tra* thì quân bài của người thứ nhất sẽ được lật lên kiểm tra (xem nó là quân bài thắng hay thua). Nếu người thứ nhất có quân bài chiến thắng thì anh ta chiếm bình tiền (do đã lấy được 1 đồng từ người thứ hai), ngược lại thì người thứ nhất mất đồng tiền đặt trước cho người thứ hai.
- Nếu chọn *đặt cược* thì đặt thêm 2 đồng vào bình. Sau đó người thứ hai (không biết quân bài của người thứ nhất) sẽ *chốt bài* hoặc *gọi bài.*
    - o Nếu *chốt bài* thì mất 1 đồng đã cược.
    - o Nếu *gọi bài* thì đặt thêm 2 đồng vào bình.

Sau đó quân bài của người thứ nhất được lật lên, nếu là quân bài thắng thì người thứ nhất thắng 3 đồng (1 đồng đặt cược trước và 2 đồng đặt cược sau) nếu là quân bài thua thì người thứ nhất mất 3 đồng cho người thứ hai.

Cây của trò chơi này như hình sau:

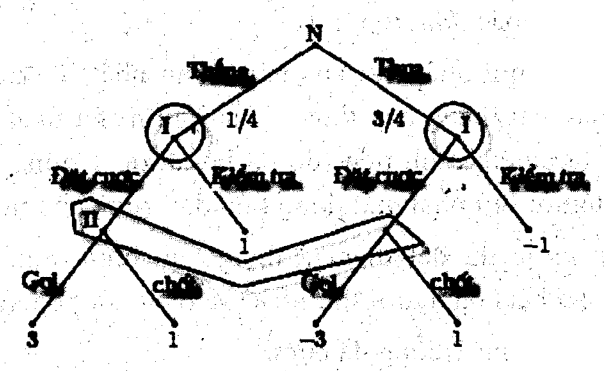


Hình 9.6

Trong trò chơi có ba phép di chuyển:

- Phép di chuyển ngẫu nhiên là N để chọn quân bài (thắng hay thua) cho người thứ nhất;
- Các phép di chuyển của người thứ nhất: *kiểm tra* hoặc *đặt cược*;
- Các phép di chuyển của người thứ hai: *chốt bài* hoặc *gọi bài.*

Tại mỗi đỉnh của cây, chúng ta gắn một nhãn là số hiệu người chơi đi từ đỉnh này. Khi lập trình, phép chuyển ngẫu nhiên (phép rút quân bài đầu tiên cho người thứ nhất), có thể được tạo ra bằng cách tạo ngẫu nhiên một số tự nhiên (chẳng hạn với tần suất thắng là $\frac{1}{4}$ và thua là $\frac{3}{4}$) có nhãn là N. Trên mỗi cung gán nhãn xác định là tên phép di chuyển.



Hình 9.7

Giá trị tại các lá thể hiện số tiền người thứ nhất thắng (nếu là số dương) và thua (nếu là số âm).

Tuy nhiên, còn một đặc điểm chưa được thể hiện rõ trên cây này: Khi người thứ hai di chuyển thì người thứ hai không rõ quân bài người thứ nhất nhận được thuộc loại nào. Nghĩa là khi thực hiện phép di chuyển, người thứ hai không biết nên thực hiện di chuyển từ vị trí nào trong hai vị trí của mình sau khi người thứ nhất đặt cược thêm 2 đồng. Chúng ta thể hiện điều này trên cây bằng cách khoanh hai vị trí này trong một đường kín và nói rằng hai đỉnh này tạo thành một *tập thông tin*.

Hai đỉnh mà người thứ nhất di chuyển từ chúng sau phép chuyển ngẫu nhiên N thì tạo thành hai tập thông tin riêng biệt do chúng diễn đạt kết quả tác động của một phép di chuyển ngẫu nhiên; chúng ta vẽ các vòng tròn nhỏ cho mỗi đỉnh này. Chúng ta cũng chỉ cần giữ lại một nhãn chung cho các đỉnh trong một tập thông tin. Nghĩa là chỉ cần gán nhãn cho từng tập thông tin.

c) Cây Kuhn

Cây Kuhn là cây trò chơi bao gồm: các giá trị trả về, tập thông tin, nhãn của các đỉnh và cung.

Định nghĩa

Trò chơi hạn chế, hai người có tổng điểm bằng 0 trong dạng mở rộng được định nghĩa là:

- Một cây T hữu hạn các đỉnh.
- Một hàm giá trị trả về được gắn giá trị cho mọi lá.
- Một tập T_0 gồm các đỉnh khác lá (mô tả các vị trí tại đó phép chuyển ngẫu nhiên xuất hiện) và với mỗi $t \in T_0$, có các tần suất phân bố trên các cung xuất phát từ t.
- Các đỉnh còn lại (không là lá, và không thuộc T_0) gồm hai nhóm các tập thông tin $T_{11}, T_{12}, \ldots, T_{1k_1}$ (cho người thứ nhất) và $T_{21}, T_{22}, \ldots, T_{2k_2}$ (cho người thứ hai).
- Với mỗi tập thông tin T_{jk} cho người j ($j = 1, 2$) tương ứng một tập các nhãn L_{jk} và với mỗi $t \in T_{jk}$ có ánh xạ 1-1 của L_{jk} lên tập các cung xuất phát từ t.

Vậy cây Kuhn có đầy đủ đặc trưng của trò chơi.

Trò chơi mà hai người đều biết cây Kuhn là trò chơi có thông tin đầy đủ.

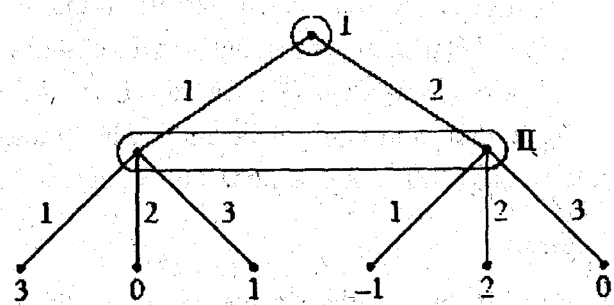
d) Mô tả trò chơi bằng dạng mở rộng

Trước hết chúng ta kiểm tra xem có thể đặt dạng chiến thuật của trò chơi vào trong dạng mở rộng hay không. Dạng chiến thuật của trò chơi mô tả được sự chọn lựa đồng thời của các người chơi, nhưng dạng mở rộng không mô tả trực tiếp được các phép chuyển đồng thời.

Tuy nhiên, các phép chuyển đồng thời có thể coi là hai phép chuyển liên tiếp theo nhau của hai người chơi mà người đi sau không biết gì về phép di chuyển của người đi trước. Sự không biết này có thể diễn đạt bằng cách sử dụng một tập thông tin. Chẳng hạn:

$$\begin{pmatrix} 3 & 0 & 1 \\ -1 & 2 & 0 \end{pmatrix}$$

Dạng ma trận



Dạng mở rộng

Hình 9.8

Người thứ nhất có tập chiến thuật nguyên thuỷ là $\{1; 2\}$, người thứ hai có tập chiến thuật nguyên thuỷ là $\{1; 2; 3\}$. Chúng ta giả sử rằng người thứ nhất di chuyển đầu tiên bằng cách chọn hàng 1 hoặc 2. Sau đó người thứ hai di chuyển nhưng không biết sự lựa chọn của người thứ nhất. Điều này được biểu thị bởi tập thông tin của người thứ hai. Sau đó người thứ hai chọn cột 1, 2 hoặc 3 và giá trị trả về thích hợp được tạo ra.

e) Thu dạng mở rộng về dạng chiến thuật

Đi theo hướng ngược lại, từ dạng mở rộng của trò chơi tìm dạng chiến thuật thì chúng ta cần quan tâm đến những chiến thuật nguyên thuỷ và những quy ước về các giá trị ngẫu nhiên.

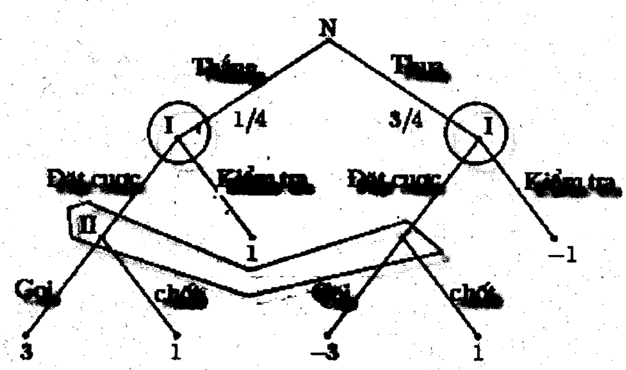
Các *chiến thuật nguyên thuỷ.* Cho trò chơi ở dạng mở rộng, trước tiên cần tìm X và Y là các tập gồm các chiến thuật nguyên thuỷ của các người chơi trong dạng chiến thuật. Một chiến thuật nguyên thuỷ của người thứ nhất là quy luật chỉ dẫn chính xác cho người thứ nhất thực hiện di chuyển trong một tập thông tin của mình. Giả sử $T_{11}, T_{12}, ..., T_{1k_1}$ là các tập thông tin của người thứ nhất và $L_{11}, L_{12}, ..., L_{1k_1}$ là các tập nhãn tương ứng. Một chiến thuật nguyên thuỷ của người thứ nhất là bộ $x = (x_1, x_2, ..., x_{k_1})$ trong đó x_i là một trong các phần tử của L_{1i} với $i = 1, 2, ..., k_1$. Nếu có m_i phần tử trong L_{1i} thì số lượng các bộ x bằng tích $m_1 \times m_2 \times ... \times m_{k_1}$. Vậy số lượng các tập chiến thuật nguyên thuỷ của người thứ nhất là $m_1 \times m_2 \times ... \times m_{k_1}$. Tập tất cả các chiến thuật nguyên thuỷ của người thứ nhất kí hiệu là X. Tương tự, nếu $T_{21}, ..., T_{2k_2}$ là các tập thông tin của người thứ hai và $L_{21}, ..., L_{2k_2}$ là các tập nhãn tương ứng, một chiến thuật nguyên thuỷ của người thứ hai là bộ $y = (y_1, ..., y_{k_2})$, trong đó $y_j \in L_{2j}$ với mỗi j. Người thứ hai có $n_1 \times n_2 \times ... \times n_{k_2}$ chiến thuật nguyên thuỷ nếu có n_j phần tử trong L_{2j}. Kí hiệu Y là tập các chiến thuật nguyên thuỷ của người thứ hai.

Các giá trị trả về ngẫu nhiên. Chúng ta quy ước giá trị trả về của phép di chuyển ngẫu nhiên như sau:

Nếu với chiến thuật nguyên thuỷ đã cố định $x \in X$ và $y \in Y$, giá trị trả về $A(x; y)$ là một số ngẫu nhiên thì thay nó bởi giá trị trung bình.

Ví dụ 1. Nếu cho chiến thuật $x \in X$ và $y \in Y$ người thứ nhất thắng ba điểm với xác suất $\frac{1}{4}$, thắng 1 điểm với xác suất $\frac{1}{4}$ và thua 1 điểm với xác suất $\frac{1}{2}$ thì giá trị trung bình trả về là $\frac{1}{4}.3 + \frac{1}{4}.1 + \frac{1}{2}.(-1) = \frac{1}{2}$. Vậy $A(x; y) = \frac{1}{2}$.

Ví dụ 2. Giả sử chúng ta cần tìm dạng chiến thuật cho ván cơ bản kết thúc trò chơi bài đã mô tả bằng chiến thuật mở rộng như cây sau:



Hình 9.9

Với người thứ nhất : Kí hiệu *Đặt cược* là *đ*, *Kiểm tra* là *k*. Người thứ nhất có hai tập thông tin. Trong mỗi tập thông tin có hai lựa chọn: *đ* và *k*. Do đó người thứ nhất có 2×2 chiến thuật nguyên thuỷ. Đó là các chiến thuật sau:

(*đ; đ*) : *Đặt cược* bất kể quân bài nhận được là quân bài thắng hay là quân bài thua.

(*đ; k*) : *Đặt cược* với quân bài thắng và *Kiểm tra* với quân bài thua.

(*k; đ*) : *Kiểm tra* với quân bài thắng và *Đặt cược* với quân bài thua.

(*k; k*) : *Kiểm tra* bất kể quân bài nhận được là quân bài thắng hay là quân bài thua.

Vậy tập các chiến thuật nguyên thuỷ là *X* = {(*đ; đ*), (*đ; k*), (*k; đ*), (*k; k*)}. Vậy *X* chứa mọi chiến thuật nguyên thuỷ kể cả tốt lẫn xấu (đặc biệt (*k; đ*) là chiến thuật khá vô lí!).

Kí hiệu *g* là *Gọi*, *c* là *Chốt*. Người thứ hai chỉ có một tập thông tin với hai lựa chọn: *g* và *c*. Do đó tập các chiến thuật nguyên thuỷ của người thứ hai là *Y* = {*g; c*}.

Bây giờ chúng ta tìm ma trận giá trị trả về: Giả sử người thứ nhất sử dụng (*đ; đ*) và người thứ hai sử dụng *g*. Nếu người thứ nhất được chia quân bài thắng (điều này xảy ra với xác xuất $\frac{1}{4}$) thì người thứ nhất thắng 3 điểm. Nếu người thứ nhất được chia quân bài thua (điều này xảy ra với xác suất $\frac{3}{4}$) thì người thứ nhất thua 3 điểm. Chiến thắng trung bình của người thứ nhất là

$$A((đ;đ), g) = 3.\frac{1}{4} + (-3).\frac{3}{4} = -\frac{3}{2} \text{ (nghĩa là thua 1,5 điểm).}$$

Tính toán tương tự có ma trận giá trị trả về là:

	g	*c*
(*đ; đ*)	$-\frac{3}{2}$	1
(*đ; k*)	0	$-\frac{1}{2}$
(*k; đ*)	−2	1
(*k; k*)	$-\frac{1}{2}$	$-\frac{1}{2}$

Chúng ta giải trò chơi 4×2 này. Dòng 3 bị chi phối bởi dòng 1 và dòng 4 bị chi phối bởi dòng 2. Điều này có thể giải thích như sau: Khi người thứ nhất được chia quân bài thắng thì dùng *Đặt cược* là tốt hơn *Kiểm tra*. Bỏ đi hai dòng cuối, ma trận còn là $\begin{pmatrix} -\frac{3}{2} & 1 \\ 0 & -\frac{1}{2} \end{pmatrix}$.

Dễ dàng giải được kết quả: Giá trị $V = -\frac{1}{4}$.

Chiến thuật tối ưu của người thứ nhất là $\left(\frac{1}{6}; \frac{5}{6}; 0; 0\right)$.

Chiến thuật tối ưu của người thứ hai là $\left(\frac{1}{2}; \frac{1}{2}\right)$.

f) Trò chơi với thông tin hoàn chỉnh

Trò chơi với thông tin hoàn chỉnh là trò chơi mà trong dạng mở rộng mỗi tập thông tin của mỗi người chơi chứa đúng một đỉnh.

Trò chơi với thông tin hoàn chỉnh có cấu trúc toán học đặc biệt đơn giản. Mỗi trò chơi với thông tin hoàn chỉnh khi thu gọn về dạng chiến thuật đều có điểm yên ngựa, cả hai người chơi đều có chiến thuật tối ưu.

6. Đệ quy và trò chơi ngẫu nhiên

a) Trò chơi ma trận có phần tử là trò chơi

Đó là những trò chơi ma trận mà người chơi khi thực hiện các chiến thuật nguyên thuỷ có thể phải thực hiện một trò chơi khác. Ví dụ trò chơi với $G = \begin{pmatrix} G_1 & 4 \\ 5 & G_2 \end{pmatrix}$, trong đó $G_1 = \begin{pmatrix} 0 & 3 \\ 2 & -1 \end{pmatrix}$ và $G_2 = \begin{pmatrix} 0 & 1 \\ 4 & 3 \end{pmatrix}$ cũng là những trò chơi.

Trong trò chơi G, vẫn quy ước người thứ nhất chọn dòng, người thứ hai chọn cột. Nếu ô trong dòng hoặc cột được chọn là số thì người thứ hai phải trả cho người thứ nhất giá trị bằng số đó và trò chơi kết thúc. Nếu ô chọn là G_1 thì trò chơi G_1 được chơi. Nếu ô chọn là G_2 thì trò chơi G_2 được chơi.

Để giải trò chơi G, trước hết chúng ta cần giải trò chơi G_1 và G_2.

- G_1: có giá trị trò chơi là $V_1 = 1$, chiến thuật tối ưu cho người thứ nhất là (0.5; 0.5). Chiến thuật tối ưu cho người thứ hai là $\left(\frac{2}{3};\frac{1}{3}\right)$.
- G_2: có giá trị trò chơi là $V_2 = 3$, chiến thuật tối ưu cho người thứ nhất là (0; 1). Chiến thuật tối ưu cho người thứ hai là (0; 1).

Khi chơi trò chơi G, các đấu thủ kết thúc trò chơi G_1 thì họ nhận được giá trị trả về trung bình là $V_1 = 1$, kết thúc trò chơi G_2 thì họ nhận được giá trị trả về trung bình là $V_2 = 3$. Do đó G có thể xem như trò chơi với ma trận: $\begin{pmatrix} 1 & 4 \\ 5 & 3 \end{pmatrix}$. Giải trò chơi này được: Giá trị trò chơi là $V = \frac{17}{5}$, chiến thuật tối ưu cho người thứ nhất là $\left(\frac{2}{5};\frac{3}{5}\right)$. Chiến thuật tối ưu cho người thứ hai là $\left(\frac{1}{5};\frac{4}{5}\right)$.

Phương pháp giải trò chơi G có thể tổng kết lại như sau: Nếu ma trận của trò chơi G có phần tử là những trò chơi khác, thì lời giải của G là lời giải của trò chơi mà trong ma trận ban đầu các phần tử là trò chơi khác sẽ được thay bằng giá trị trung bình trả về của trò chơi đó.

Sự phân giải ma trận. Trò chơi G nêu trong ví dụ trên còn có thể coi là trò chơi ma trận 4×4. Tập bốn chiến thuật nguyên thuỷ của người thứ nhất kí hiệu là : {(1; 1), (1; 2), (2; 1), (1; 2)}, với $(i; j)$ biểu thị dùng hàng i trong G và trong G_i đã dùng dòng j để chơi. Tương tự hiểu quy ước như vậy với người thứ hai (dùng cột). Ma trận 4×4 thành:

$$G = \begin{pmatrix} 0 & 3 & 4 & 4 \\ 2 & -1 & 4 & 4 \\ 5 & 5 & 0 & 1 \\ 5 & 5 & 4 & 3 \end{pmatrix}.$$

Chúng ta có thể giải trò chơi này bằng các phương pháp đã nêu (phương pháp đơn hình) được kết quả: Giá trị trò chơi là $\frac{17}{5}$.

Chiến thuật tối ưu cho người thứ nhất là (0.2; 0.2; 0; 0.6).

Chiến thuật tối ưu cho người thứ hai là (0.13333; 0.06667; 0; 0.8).

Từ đó có thể suy ra trong trò chơi G ban đầu, chiến thuật tối ưu cho người thứ nhất là $(0.2 + 0.2;\ 0 + 0.6) = \left(\frac{2}{5}; \frac{3}{5}\right)$. Chiến thuật tối ưu cho người thứ hai là

$(0.13333 + 0.06667;\ 0 + 0.8) = \left(\frac{1}{5}; \frac{4}{5}\right)$.

Nói chung, khi chúng ta có trò chơi G và nếu sau khi sắp xếp lại các hàng và các cột nào đó chúng ta phân giải được ma trận G thành dạng:

$G = \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix}$ mà G_{11} và G_{22} là ma trận tuỳ ý, còn G_{12} và G_{21} là ma trận hằng số (đó là ma trận mà các phần tử đều bằng một hằng số); thì chúng ta có thể giải G theo phương pháp nêu trên, coi như các người chơi chọn hàng hoặc cột từ ma trận đã được phân giải này, từ đó suy ra chọn hàng và cột từ ma trận ban đầu.

b) Trò chơi nhiều giai đoạn

Một trò chơi đóng vai trò là phần tử của một trò chơi khác thì bản thân nó có thể cũng có phần tử là trò chơi và phương pháp giải nêu trên được lặp lại hữu hạn lần.

Ví dụ. *Trò chơi thanh tra*

Người thứ hai cần phải thực hiện một hoạt động tại một giai đoạn nào đó trong n giai đoạn của một quá trình. Người thứ nhất bí mật theo dõi giám sát người thứ hai chỉ tại một giai đoạn trong n giai đoạn này. Nếu người thứ hai đang hoạt động mà người thứ nhất giám sát được thì người thứ hai mất cho người thứ nhất 1 đơn vị. Nếu người thứ nhất không giám sát được khi người thứ hai đang hoạt động thì giá trị trả về là 0.

		Hoạt động	Chờ
G_n	Giám sát	1	0
	Chờ	0	G_{n-1}

Kí hiệu trò chơi này là G_n. Trong đó $G_1 = (1)$.

Chúng ta giải bằng cách lặp đi lặp lại hữu hạn lần (đệ quy) tìm được:

Giá trị trò chơi G_1 là $\mathrm{Val}(G_1) = 1$.

Giá trị trò chơi $G_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ là $\mathrm{Val}(G_2) = \frac{1}{2}$.

Giá trị trò chơi $G_3 = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$ là $\mathrm{Val}(G_3) = \frac{1}{3}$.

...

Giá trị trò chơi $G_n = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{n-1} \end{pmatrix}$ là $\mathrm{Val}(G_n) = \frac{1}{n}$.

Công thức truy hồi tính giá trị trò chơi là: $\mathrm{Val}(G_n) = \dfrac{\frac{1}{n-1}}{1+\frac{1}{n-1}} = \frac{1}{n}$.

c) Trò chơi đệ quy

Định nghĩa. Trong một trò chơi có một số phần tử tham gia là trò chơi có thể lại dẫn về trò chơi ban đầu được gọi là trò chơi đệ quy.

Ví dụ 1. $G = \begin{pmatrix} G & 1 \\ 1 & 0 \end{pmatrix}$ là trò chơi đệ quy nhưng có thể lặp vô hạn (nếu hai người chơi luôn chơi dòng 1 và cột 1).

Ví dụ 2. $G = \begin{pmatrix} G & 1 & 0 \\ 1 & 0 & G \\ 0 & G & 1 \end{pmatrix}$.

Giả sử chiến thuật của người thứ nhất là $p = (p_1; p_2; p_3)$, theo nguyên lí cân bằng có

$$\begin{cases} v = v.p_1 + p_2 \\ v = p_1 + v.p_2 \\ v = v.p_2 + p_3 \end{cases}$$

suy ra $v = \frac{1}{2}$; $p_1 = p_2 = p_3 = \frac{1}{3}$. Giá trị trò chơi là $v = \frac{1}{2}$. Chiến thuật $\left(\frac{1}{3}; \frac{1}{3}; \frac{1}{3}\right)$ là chiến thuật tối ưu cho cả hai người chơi.

d) Sự di chuyển ngẫu nhiên giữa các trò chơi

Khái quát hoá khái niệm trò chơi **đệ quy**: cho phép chọn trò chơi tiếp theo không chỉ phụ thuộc chiến thuật nguyên thuỷ do người chơi chọn mà còn có thể chọn ngẫu nhiên. Giả sử $G_1, G_2, ..., G_n$ và $p_1, p_2, ..., p_n$ là các tần suất có tổng bằng 1. Chúng ta sử dụng kí hiệu $p_1G_1 + ... + p_nG_n$ để mô tả tình trạng trò chơi tiếp theo được chọn ngẫu nhiên là G_i với tần suất p_i. Chúng ta cũng kí hiệu số z là giá trị mà người thứ hai phải trả cho người thứ nhất thay bằng một trò chơi tầm thường ma trận (z) có kích thước 1×1. Ví dụ $\frac{1}{2}G_1 + \frac{1}{2}(3)$ biểu thị khi tung đồng xu đồng chất cân đối nếu nhận được mặt ngửa thì trò chơi G_1 được chơi và nếu nhận được mặt sấp thì người thứ hai trả cho người thứ nhất một lượng là 3 đơn vị.

Ví dụ. Giả sử G_1 và G_2 quan hệ với nhau như sau:

$$G_1 = \begin{pmatrix} \frac{1}{2}G_2 + \frac{1}{2}(0) & 1 \\ 2 & 0 \end{pmatrix}; \qquad G_2 = \begin{pmatrix} \frac{2}{3}G_1 + \frac{1}{3}(-2) & 0 \\ 0 & -1 \end{pmatrix}.$$

Để giải, chúng ta đặt v_i là giá trị trò chơi i, $i = 1$ và 2. Do $0 \le v_1 \le 1$ và $-1 \le v_2 \le 0$ nên G_1 và G_2 không có điểm yên ngựa, áp dụng công thức

$$Val(A) = Val\begin{pmatrix} a & b \\ d & c \end{pmatrix} = \frac{ac - bd}{(a-b)+(c-d)}$$

ta có: $v_1 = Val\begin{pmatrix} \frac{1}{2}v_2 & 1 \\ 2 & 0 \end{pmatrix} = \frac{4}{6 - v_2}$; $v_2 = Val\begin{pmatrix} \frac{2}{3}v_1 - \frac{2}{3} & 0 \\ 0 & -1 \end{pmatrix} = \frac{2(1 - v_1)}{5 - 2v_1}$.

Suy ra $v_1 = \dfrac{4}{6 + \dfrac{2(1-v_1)}{5-2v_1}}$ dẫn tới phương trình: $7v_1^2 - 20v_1 + 10 = 0$, giải được:

$$v_1 = \frac{10 - \sqrt{30}}{7}, \; v_2 = \frac{10 - 2\sqrt{30}}{5}.$$

Từ đó tìm được chiến thuật tối ưu cho hai người chơi.

e) Trò chơi ngẫu nhiên

Nếu trò chơi nhiều giai đoạn có thêm khả năng trả về giá trị tại mỗi giai đoạn thì trò chơi đó được gọi là trò chơi ngẫu nhiên. Đây là lĩnh vực đang được nghiên cứu mạnh mẽ hiện nay.

Định nghĩa

Một trò chơi ngẫu nhiên G gồm tập hữu hạn các vị trí hoặc giai đoạn $\{1, 2, ..., N\}$ trong đó có một giai đoạn là vị trí xuất phát. Chúng ta kí hiệu $G^{(k)}$ là giai đoạn thứ k. Liên quan với mỗi giai đoạn là một ma trận $A^{(k)} = (a_{ij}^{(k)})$. Khi trò chơi đang ở giai đoạn k, các người chơi đồng thời chọn hàng và cột của $A^{(k)}$ ví dụ hàng i cột j. Có hai điều quyết định kết quả.

- Người thứ nhất thắng một lượng là $a_{ij}^{(k)}$.
- Xác suất phụ thuộc i, j, k để trò chơi kết thúc hoặc chuyển sang giai đoạn khác. Xác suất để trò chơi kết thúc kí hiệu là $s_{ij}^{(k)}$ và xác suất để giai đoạn l là giai đoạn tiếp theo kí hiệu là $P_{ij}^{(k)}(l)$, thoả mãn: $s_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l) = 1$ với mọi i, j và k. Giá trị trả về được cộng dồn tích luỹ lại cho đến khi trò chơi kết thúc. Để bảo đảm trò chơi dần dần sẽ dừng, chúng ta chấp nhận một giả định là mọi xác suất dừng đều dương. Kí hiệu s là số nhỏ nhất trong các xác suất này thì: $s = \min_{i,j,k} s_{ij}^{(k)} > 0$.

Với giả định trên, trò chơi kết thúc sau một số hữu hạn bước đi sẽ có xác suất bằng 1. Giả định này cũng tạo ra tổng giá trị tích luỹ trả về là hữu hạn không phụ thuộc vào những trò chơi nào đã được chọn trong khi chơi. Gọi M là số lớn nhất trong các giá trị tuyệt đối của các giá trị trả về tại các giai đoạn ($M = \max_{i,j,k} |a_{ij}^{(k)}|$) thì tổng tích luỹ chờ đợi trả về cho người này hoặc người khác là hội tụ (có giới hạn) bởi vì $M + (1-s)M + (1-s)^2 M + ... = \dfrac{M}{s}$.

Người thứ nhất muốn tổng tích luỹ này lớn nhất còn người thứ hai muốn nhỏ nhất. Chúng ta mô tả trò chơi này là:

$$G^{(k)} = \left(a_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l).G^{(l)} \right).$$

Chú ý rằng tổng các xác suất cho mỗi thành phần trong ma trận là nhỏ hơn 1, phần xác suất còn lại là $s_{ij}^{(k)}$ là xác suất kết thúc trò chơi. Sau khi có giá trị trả về nó sẽ quyết định ngẫu nhiên là trò chơi được kết thúc hay giai đoạn khác sẽ được chơi tiếp theo. Do đó không có giới hạn cho độ dài của trò chơi, đây là trò chơi vô hạn.

Do những điều phức tạp nêu trên, với trò chơi ngẫu nhiên người ta thường đề cập tới *chiến thuật tối ưu tĩnh*. Đó là các chiến thuật chỉ rõ cho người chơi một phân bổ xác suất cho các lựa chọn của họ chỉ phụ thuộc vào trò chơi $G^{(k)}$ hiện đang chơi tại giai đoạn k không phụ thuộc vào n giai đoạn hoặc các giai đoạn đã chơi qua. Định lí sau phát biểu về sự tồn tại của chiến thuật tĩnh tối ưu:

Định lí (Shapley 1952)

> *Mỗi trò chơi $G^{(k)}$ có giá trị $v(k)$ là nghiệm duy nhất của hệ phương trình:* $v(k) = Val\left(a_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l).v(l)\right)$ *với $k=1, 2, ..., N$.*
>
> *Mỗi người chơi có một chiến thuật tĩnh tối ưu sao cho tại giai đoạn k sử dụng được chiến thuật hỗn hợp cho trò chơi có ma trận:*
>
> $$A^{(k)}(v) = \left(a_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l).v(l)\right),\ v=(v(1), v(2), ..., v(N)).$$

Ví dụ. Xét trò chơi ngẫu nhiên chỉ có một trạng thái:

$$G = \begin{pmatrix} 1+\frac{3}{5}G & 3+\frac{1}{5}G \\ 1+\frac{4}{5}G & 2+\frac{2}{5}G \end{pmatrix}.$$

Trên quan điểm của người thứ hai, cột 1 có vẻ tốt hơn cột 2 với phương diện giá trị trả về ngay (là các số 1), nhưng cột 2 có vẻ làm cho trò chơi kết thúc sớm hơn (vì xác suất thực hiện lại G nhỏ hơn). Vậy chọn các cột như thế nào?

Giả sử rằng mọi chiến thuật tối ưu là thực thi được nghĩa là trò chơi không có điểm yên ngựa. Chúng ta sẽ kiểm tra lại điều này khi trò chơi đã kết thúc để có thể nhìn thấy giả sử là đúng.

Khi đó:

$$v = Val\begin{pmatrix} 1+\frac{3}{5}v & 3+\frac{1}{5}v \\ 1+\frac{4}{5}v & 2+\frac{2}{5}v \end{pmatrix} = \frac{(1+\frac{4}{5}v)(3+\frac{1}{5}v)-(1+\frac{3}{5}v)(2+\frac{2}{5}v)}{(1+\frac{4}{5}v)+(3+\frac{1}{5}v)-(1+\frac{3}{5}v)-(2+\frac{2}{5}v)} = 1+v-\frac{2}{25}v^2$$

$$\Rightarrow v = \frac{5}{2}\sqrt{2} \text{ (do } v > 0\text{)}.$$

Thay giá trị này của v vào ma trận $\begin{pmatrix} 1+\frac{3}{5}v & 3+\frac{1}{5}v \\ 1+\frac{4}{5}v & 2+\frac{2}{5}v \end{pmatrix}$ ta được:

$$\begin{pmatrix} 1+\frac{3}{2}\sqrt{2} & 3+\frac{1}{2}\sqrt{2} \\ 1+2\sqrt{2} & 2+\sqrt{2} \end{pmatrix}.$$

Từ đó tìm được chiến thuật tối ưu cho người thứ nhất trên ma trận này là $p = \left(\sqrt{2}-1; 2-\sqrt{2}\right)$ và chiến thuật tối ưu cho người thứ hai là $q = \left(1-\frac{\sqrt{2}}{2}; \frac{\sqrt{2}}{2}\right)$. Do tồn tại các vectơ tần suất này nên giả sử ban đầu là đúng và có chiến thuật tối ưu, giá trị của trò chơi ngẫu nhiên này là $v = \frac{5}{2}\sqrt{2}$.

f) Lời giải gần đúng

Với một trò chơi ngẫu nhiên nhiều trạng thái, hệ phương trình $v(k) = Val\left(a_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l).v(l)\right)$ trở thành hệ khá phức tạp gồm những phương trình không tuyến tính đồng thời. Chúng ta không hi vọng giải được hệ này trong dạng tổng quát. Tuy nhiên, có một phương pháp giải bằng cách lặp đi lặp lại kết quả gần đúng nhiều lần. Phương pháp này cũng chính là cơ sở để chứng minh định lí Shapley đã nêu trên do đó được gọi là phương pháp lặp Shapley.

Đầu tiên chúng ta tạo một dự đoán về kết quả gọi là $v_0 = (v_0(1), v_0(2), ..., v_0(N))$. Bất kì dự đoán nào cũng được.

Chúng ta có thể dùng dự đoán ban đầu là $v_0 = 0 = (0, 0, ..., 0)$. Sau khi nhận được v_n chúng ta định nghĩa đệ quy v_{n+1} bởi đẳng thức sau:

$$v_{n+1}(k) = Val\left(a_{ij}^{(k)} + \sum_{l=1}^{N} P_{ij}^{(k)}(l).v_n(l)\right) \text{ với } k = 1, 2, ..., N.$$

Ví dụ. Xét trò chơi ngẫu nhiên chứa hai trạng thái tương ứng là hai trò chơi $G^{(1)}$ và $G^{(2)}$ liên hệ với nhau như sau:

$$G^{(1)} = \begin{pmatrix} 4+0.3G^{(1)} & 0+0.4G^{(2)} \\ 1+0.4G^{(2)} & 3+0.5G^{(1)} \end{pmatrix}; \qquad G^{(2)} = \begin{pmatrix} 0+0.5G^{(1)} & -5 \\ -4 & 1+0.5G^{(2)} \end{pmatrix}.$$

Dùng $v_0 = (0; 0)$ là dự đoán ban đầu, tìm được $v_1 = (2; -2)$ vì:

$$v_1(1) = Val\begin{pmatrix} 4 & 0 \\ 1 & 3 \end{pmatrix} = 2; \quad v_1(2) = Val\begin{pmatrix} 0 & -5 \\ -4 & 1 \end{pmatrix} = -2.$$

Lặp tiếp theo có: $v_2(1) = Val\begin{pmatrix} 4.6 & -8 \\ 0.2 & 4 \end{pmatrix} = 2.0174; \quad v_2(2) = Val\begin{pmatrix} 1 & -5 \\ -4 & 0 \end{pmatrix} = -2.$

Tiếp tục tìm được các kết quả sau:

$v_3(1) = 2.0210$	$v_3(2) = -1.9983$
$v_4(1) = 2.0220$	$v_4(2) = -1.9977$
$v_5(1) = 2.0224$	$v_5(2) = -1.9974$
$v_6(1) = 2.0225$	$v_6(2) = -1.9974$

Sai số lớn nhất mắc phải của v_6 nhiều nhất là 0.0002 (vì xác xuất dừng trò chơi nhỏ nhất là 0.5 nên sai số so với kết quả đúng sẽ giảm theo hàm số mũ có độ hội tụ ít nhất là 0.5^n).

Chiến thuật tối ưu khi dùng giá trị v_6 là dễ dàng tìm thấy:

- Với trò chơi $G^{(1)}$,

  chiến thuật tối ưu cho người thứ nhất là $p^{(1)} = (0.4134; 0.5866)$,

  chiến thuật tối ưu cho người thứ hai là $q^{(1)} = (0.5219; 0.4718)$.

- Với trò chơi $G^{(2)}$,

  chiến thuật tối ưu cho người thứ nhất là $p^{(2)} = (0.3996; 0.6004)$,

  chiến thuật tối ưu cho người thứ hai là $q^{(2)} = (0.4995; 0.5005)$.

Bài tập

9.26. Tội phạm B lên tàu từ Hà Nội về Hải Phòng để trốn sự truy nã của công an A. Công an A có thể lên tàu nhanh, bắt B tại Hải Phòng. Tại ga Hải Dương, B có thể xuống để tẩu thoát. Tuy nhiên công an A cũng biết điều này và có thể cũng xuống tàu tại Hải Dương để truy tìm. Sau đây là bảng lượng giá trị cho công an A trong kế hoạch truy tìm tội phạm B:

		B xuống ga	
		Hải Dương	Hải Phòng
A xuống ga	Hải Dương	100	–50
	Hải Phòng	0	100

Tìm chiến thuật hỗn hợp tối ưu cho A và B, giá trị trò chơi bằng bao nhiêu?

9.27. Giải trò chơi ma trận có ma trận lượng giá là $A = \begin{pmatrix} 0 & 2 \\ t & 1 \end{pmatrix}$ với t là số thực tuỳ ý. Vẽ đồ thị của $v(t)$ là giá trị trò chơi phụ thuộc t.

9.28. Giải trò chơi trên các ma trận sau bằng cách giảm bớt các hàng và cột bị chi phối đưa về ma trận 2×2:

a) $\begin{pmatrix} 5 & 4 & 1 & 0 \\ 4 & 3 & 2 & -1 \\ 0 & -1 & 4 & 3 \\ 1 & -2 & 1 & 2 \end{pmatrix}$; b) $\begin{pmatrix} 10 & 0 & 7 & 1 \\ 2 & 6 & 4 & 7 \\ 6 & 3 & 3 & 5 \end{pmatrix}$.

9.29. Người ta chứng minh được kết luận sau : "Với một trò chơi đã cho, các chiến thuật $(p_1, p_2, ..., p_m)$ và $(q_1, q_2, ..., q_n)$ tương ứng là các chiến thuật tối ưu cho người thứ nhất và người thứ hai nếu giá trị trung bình nhận được ít nhất của người thứ nhất bằng giá trị trung bình phải trả nhiều nhất của người thứ hai".

Chứng tỏ rằng chiến thuật $p = \left(\frac{6}{37}; \frac{20}{37}; 0; \frac{11}{37}\right)$ và $q = \left(\frac{14}{37}; \frac{4}{37}; 0; \frac{19}{37}; 0\right)$ tương ứng là chiến thuật tối ưu cho người thứ nhất và người thứ hai trong trò chơi ma trận như sau:

$$\begin{pmatrix} 5 & 8 & 3 & 1 & 6 \\ 4 & 2 & 6 & 3 & 5 \\ 2 & 4 & 6 & 4 & 1 \\ 1 & 3 & 2 & 5 & 3 \end{pmatrix}.$$

Tìm giá trị của trò chơi.

9.30. ***Trò chơi Mendelsohn***

Hai người chơi đồng thời chọn mỗi người một số nguyên không vượt quá mức nào đó. Chẳng hạn: Hai người chọn số nguyên trong phạm vi từ 1 đến 100. Nếu hai số bằng nhau thì giá trị trả về là 0, nếu chọn lớn hơn đối thủ 1 đơn vị thì thắng 1, nếu chọn lớn hơn đối thủ 2 đơn vị hoặc nhiều hơn thì thua 2. Bảng giá trị trả về cho người thứ nhất (chọn dòng) như sau:

	[redacted]					
1	0	-1	2	2	2	...
2	1	0	-1	2	2	...
3	-2	1	0	-1	2	...
4	-2	-2	1	0	-1	...
5	-2	-2	-2	1	0	...
...	...	...	...	...	...	...

Hãy tìm cách loại bỏ những dòng và cột bị chi phối, sau đó sử dụng định lí cân bằng tìm chiến thuật tối ưu cho hai người chơi.

9.31. Hãy lập trình tính ma trận nghịch đảo của ma trận khả đảo A biết một số định nghĩa như sau:

- Ma trận đơn vị cấp m là ma trận $m \times m$ có các phần tử trên đường chéo chính bằng 1, các phần tử còn lại bằng 0. Ví dụ ma trận đơn vị cấp 4 :

$$I = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

- Ma trận nghịch đảo của ma trận vuông $A(a_{ij})_{m\times m}$ là ma trận $B(b_{ij})_{m\times m}$ sao cho $A.B = B.A = I$ (I là ma trận đơn vị cấp m). Kí hiệu ma trận nghịch đảo của ma trận vuông A là A^{-1}. Để tìm ma trận nghịch đảo của ma trận khả đảo A có thể dùng các phép biến đổi sơ cấp như nhau biến đổi đồng thời ma trận A và ma trận đơn vị I. Khi A thành ma trận đơn vị thì ma trận I thành ma trận A^{-1}. Các phép biến đổi sơ cấp trên ma trận là: nhân các phần tử của một hàng với số k khác 0 (định thức của ma trận tăng k lần), đổi chỗ hai hàng cho nhau (định thức của ma trận đổi dấu), Cộng k lần hàng i_1 vào hàng i_2 (định thức của ma trận không đổi).
- Định thức của một ma trận vuông $A(a_{ij})_{m\times m}$ được kí hiệu là $\det(A)$ được tính theo công thức đệ quy sau :
  - Nếu A là ma trận cấp 1 : $A=(a_{11})$ thì $\det(A) = a_{11}$.

- Nếu A là ma trận vuông cấp 2 : $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ thì

$$\det(A) = a_{11}.a_{22} - a_{21}.a_{12}.$$

- Nếu A là ma trận vuông cấp m :

$$\det(A) = (-1)^{i+1}a_{i1}\det(M_{i1}) + (-1)^{i+2}a_{i2}\det(M_{i2}) + \ldots + (-1)^{i+m}a_{im}\det(M_{im})$$

trong đó M_{ij} là ma trận sinh ra từ A sau khi bỏ dòng i và cột j ($j = \overline{1..m}$, i là một hàng). Nếu kí hiệu $C_{ij} = (-1)^{i+j}\det(M_{ij})$ thì $\det(A) = \sum_{j=1}^{m} a_{ij}.C_{ij}$ với i chọn trong $[1,\ldots, m]$. Số C_{ij} gọi là phần bù đại số của phần tử a_{ij}. Kí hiệu C là ma trận gồm các phần bù đại số C_{ij} (nằm ở dòng i, cột j). C^T là ma trận chuyển vị của ma trận C thì ma trận nghịch đảo của ma trận A là:

$$A^{-1} = \frac{1}{\det(A)}.C^T = \frac{1}{\det(A)}.\begin{pmatrix} C_{11} & C_{21} & \cdots & C_{m1} \\ C_{12} & C_{22} & \cdots & C_{m2} \\ \vdots & \vdots & & \vdots \\ C_{1m} & C_{2m} & \cdots & C_{mm} \end{pmatrix}.$$

- Điều kiện để ma trận A có ma trận nghịch đảo là $\det(A) \neq 0$.

Định lí

Giả sử ma trận A là khả đảo và $\mathbf{1}^T A^{-1}\mathbf{1} \neq 0$ (nghĩa là tổng các số hạng của ma trận A^{-1} là khác 0) và tồn tại chiến thuật p và q sao cho $\mathbf{p} \geq \mathbf{0}$ và $\mathbf{q} \geq \mathbf{0}$, thì trò chơi trên ma trận A sẽ có giá trị $V = \dfrac{1}{\mathbf{1}^T A^{-1}\mathbf{1}}$ với chiến thuật tối ưu p và q mà: $p^T = V\mathbf{1}^T A^{-1}$, $q = VA^{-1}\mathbf{1}$, ở đây $\mathbf{1} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$ và $\mathbf{1}^T = \begin{pmatrix} 1 & 1 & \ldots & 1 \end{pmatrix}$.

9.32. Xét trò chơi có ma trận $\begin{pmatrix} -2 & 2 & -1 \\ 1 & 1 & 1 \\ 3 & 0 & 1 \end{pmatrix}$

a) Kiểm tra xem ma trận có điểm yên ngựa hay không?

b) Chứng tỏ tồn tại ma trận nghịch đảo.

c) Chứng tỏ người thứ hai có chiến thuật tối ưu với trọng số dương cho mỗi cột.

d) Vì sao trong trò chơi này không cần dùng đẳng thức $q = \dfrac{A^{-1}.\mathbf{1}}{\mathbf{1}^T.A^{-1}.\mathbf{1}}$ để tìm chiến thuật tối ưu của người thứ hai?

9.33. Người thứ hai chọn một số $j \in \{1, 2, 3, 4\}$ và người thứ nhất cố gắng phỏng đoán số mà người thứ hai đã chọn. Nếu đoán đúng thì thắng 2^j điểm từ người thứ hai, ngược lại không phải trả gì. Lập ma trận trò chơi và giải trò chơi.

9.34. Xét trò chơi trên ma trận $A = \begin{pmatrix} 0 & 7 & 2 & 4 \\ 1 & 4 & 8 & 2 \\ 9 & 3 & -1 & 6 \end{pmatrix}$. Qua kinh nghiệm chơi trò chơi này, người thứ nhất tin rằng người thứ hai sẽ dùng chiến thuật

$$q = \left(\frac{1}{5}; \frac{1}{5}; \frac{1}{5}; \frac{2}{5}\right).$$

a) Tìm chiến thuật Baye phản ứng lại chiến thuật q nêu trên.

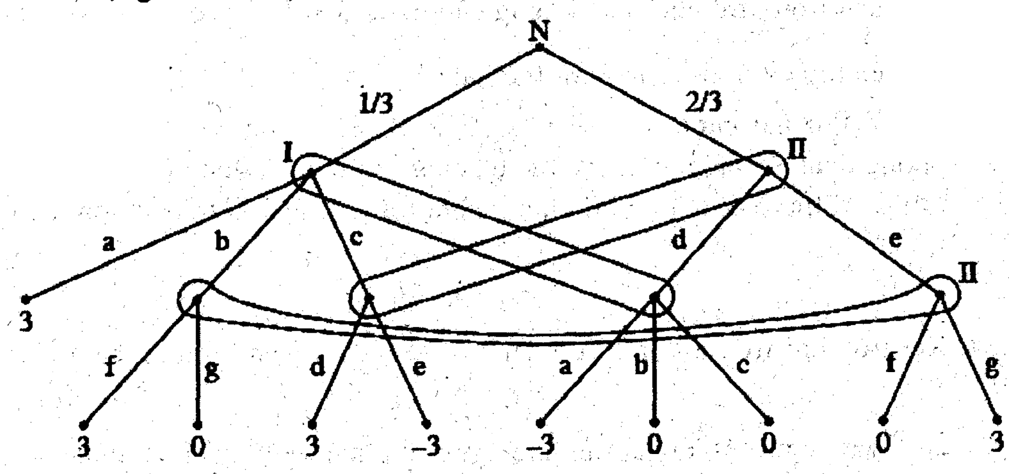
b) Giả sử người thứ hai dự đoán đúng rằng người thứ nhất sẽ dùng chiến thuật Baye để chống lại chiến thuật $q = \left(\frac{1}{5}; \frac{1}{5}; \frac{1}{5}; \frac{2}{5}\right)$ của mình. Hãy hướng dẫn cho người thứ hai tìm chiến thuật Baye chống lại người thứ nhất khi người thứ nhất phản ứng lại chiến thuật $q = \left(\frac{1}{5}; \frac{1}{5}; \frac{1}{5}; \frac{2}{5}\right)$.

9.35. Giải trò chơi với ma trận $A_{3\times 3} = \begin{pmatrix} 0 & 5 & -2 \\ -3 & 0 & 4 \\ 6 & -4 & 0 \end{pmatrix}$.

9.36. *Trò chơi với đồng bạc*

Người thứ hai chọn một trong hai phòng để dấu một đồng bạc. Người thứ nhất không biết phòng dấu đồng bạc và thử tìm. Nếu đồng bạc dấu ở phòng 1 và người thứ nhất tìm ở phòng 1 thì xác suất tìm thấy là $\frac{1}{2}$. Nếu đồng bạc dấu ở phòng 2 và người thứ nhất tìm ở phòng 2 thì xác suất tìm thấy là $\frac{1}{3}$. Nếu tìm được đồng bạc thì người thứ nhất được đồng bạc đó, ngược lại thì người thứ hai được. Hãy vẽ cây trò chơi và giải trò chơi.

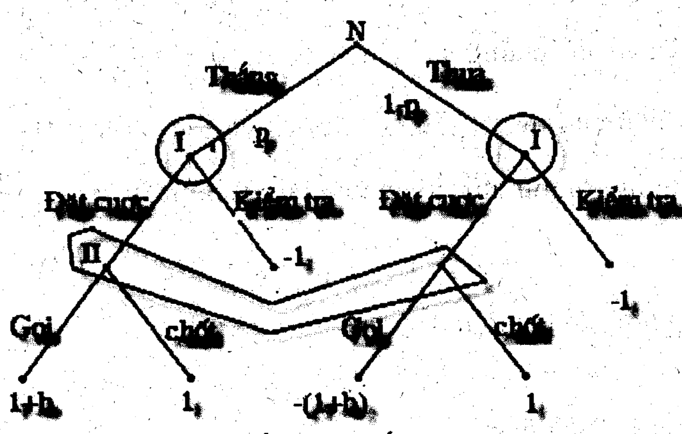
9.37. Tìm dạng chiến thuật tối ưu của trò chơi với cây trò chơi sau :



Sau đó giải trò chơi này

9.38. *Ván cơ bản kết thúc trò chơi bài*

Tổng quát hoá ván cơ bản kết thúc trò chơi bài là cách đặt xác suất cho quân bài thắng là một số tuỳ ý p, $0 \le p \le 1$ và cho tiền đặt cược là một số tuỳ ý b, $b > 0$.



Hãy tìm giá trị trò chơi và chiến thuật tối ưu cho từng người.

Lưu ý. Với $p \ge \dfrac{2+b}{2+2b}$ thì có điểm yên ngựa. Khi kết thúc chú ý rằng $p < \dfrac{2+b}{2+2b}$, chiến thuật tối ưu của người thứ hai không phụ thuộc vào p.

9.39. Giải hệ trò chơi gồm các ma trận sau:

a) $G = \begin{pmatrix} 0 & G_1 \\ G_2 & G_3 \end{pmatrix}$; $\quad G_1 = \begin{pmatrix} 4 & 3 \\ 1 & 2 \end{pmatrix}$; $\quad G_2 = \begin{pmatrix} 0 & 6 \\ 5 & 1 \end{pmatrix}$; $\quad G_3 = \begin{pmatrix} 0 & -2 \\ -2 & 0 \end{pmatrix}$.

b) Giải trò chơi ma trận sau:

$$G=\begin{pmatrix} 0 & 6 & 0 & 1 \\ 0 & 3 & 0 & 5 \\ 5 & 0 & 2 & 0 \\ 1 & 0 & 4 & 0 \end{pmatrix}.$$

9.40. *Trò chơi đệ quy*

Giải trò chơi : $G=\begin{pmatrix} G & 2 \\ 0 & 1 \end{pmatrix}.Q$ trong đó quy ước Q là giá trị trả về khi trò chơi kéo dài vô hạn.

9.41. Xét ba trò chơi có quan hệ với nhau sau đây:

$$G_1=\begin{pmatrix} G_1 & G_2 & G_3 \\ G_2 & G_3 & G_1 \\ G_3 & G_1 & G_2 \end{pmatrix}; \quad G_2=\begin{pmatrix} G_1 & 0 \\ 0 & 2 \end{pmatrix}; \quad G_3=\begin{pmatrix} G_2 & 1 \\ 1 & 0 \end{pmatrix}.$$

Giả sử trò chơi kết thúc và không có ma trận trò chơi nào chứa điểm yên ngựa. Hãy giải trò chơi.

9.42. Giải trò chơi ngẫu nhiên: $G=\begin{pmatrix} 4 & 1+\frac{1}{3}G \\ 0 & 1+\frac{2}{3}G \end{pmatrix}.$

9.43. Cho trò chơi ngẫu nhiên có hai trạng thái sau đây:

$$G^{(1)}=\begin{pmatrix} 2 & 2+0.5G^{(2)} \\ 0 & 4+0.5G^{(2)} \end{pmatrix}; \quad G^{(2)}=\begin{pmatrix} -4 & 0 \\ -2+0.5G^{(1)} & -4+0.5G^{(1)} \end{pmatrix}.$$

a) Dùng định lí Shapley tìm chính xác các giá trị $v(1)$ và $v(2)$ của hai trò chơi $G^{(1)}$ và $G^{(2)}$ nếu giả sử các trò chơi này khi kết thúc trả về được các giá trị đó.

b) Viết chương trình thực hiện lặp Shapley để tìm giá trị trò chơi ở giai đoạn k (nhập từ bàn phím) là $v_k = (v_k(1); v_k(2))$, bắt đầu với dự đoán cho giai đoạn khởi trị là $v_0 = (v_0(1) = 0; v_0(2) = 0)$ và so sánh với giá trị đúng tìm được từ câu a).

Chuyên đề 10

THUẬT TOÁN MÔ PHỎNG TỰ NHIÊN GIẢI BÀI TOÁN TỐI ƯU TỔ HỢP

Trong chuyên đề này chúng ta sẽ tìm hiểu hai thuật toán được sử dụng nhiều nhất hiện nay để giải quyết các bài toán tối ưu tổ hợp khó, đó là thuật toán di truyền và thuật toán tối ưu hoá đàn kiến. Thuật toán di truyền được sử dụng nhiều trong những năm 1970, tuy nhiên trong những năm gần đây thì thuật toán tối ưu hoá đàn kiến tỏ ra hiệu quả và được sử dụng nhiều hơn.

I. Bài toán tối ưu tổ hợp

Bài toán tối ưu tổ hợp tổng quát được phát biểu như sau:

Xét bài toán cực tiểu hoá (S, f, Ω) trong đó S là tập hợp hữu hạn trạng thái, f là hàm mục tiêu xác định trên S còn Ω là các ràng buộc để xác định S qua các thành phần của tập hữu hạn C và các liên kết của tập này. Các tập S, C và Ω có các đặc tính sau:

- $C = \{c_1, c_2, \ldots, c_n\}$ là tập hữu hạn gồm n thành phần. Ta kí hiệu X là tập các dãy trong C có độ dài không quá h :

$$X = \{<u_0, \ldots, u_k> \mid u_k \in C\ \forall i \leq k \leq h\}.$$

- Tồn tại tập con X^* của X và ánh xạ φ từ X^* lên S sao cho $\varphi^{-1}(s)$ không rỗng với mọi $s \in S$ và là tập xây dựng được từ tập con C_0 của C và Ω.
- Từ C_0 mở rộng được thành X^* theo thủ tục tuần tự :
  i) $x_0 =< u_0 >$ là mở rộng được với $\forall u_1 \in C_0$.
  ii) Nếu $x_k =< u_1,...,u_k >$ là mở rộng được thì tồn tại từ Ω xác định được tập con $J(x_k)$ của C sao cho với mọi $u_{k+1} \in J(x_k)$ thì $x_{k+1} =< u_0, u_1,...,u_{k+1} >$ là mở rộng được và $x_k \in X^*$ khi $J(x_k)$ là rỗng.

Với mọi $u_0 \in C_0$, thủ tục mở rộng nêu trên xây dựng được mọi phần tử của X^*. Không giảm tổng quát ta giả thiết rằng có tương ứng giữa các phần tử trong X^* với mỗi đường đi được mở rộng từ mỗi u_0 trong C_0.

II. Thuật toán di truyền và tính toán tiến hoá

Ý tưởng thuật toán di truyền (Genetic Algorithms - GA) dựa trên mô phỏng theo quá trình tiến hoá tự thích nghi của các quần thể sinh học dựa trên thuyết Darwin. Thoạt tiên, nó được sử dụng để giải quyết các bài toán riêng rẽ xuất phát từ sinh học vào cuối những năm 1950 và được Holland trình bày một cách có hệ thống để giải quyết bài toán tối ưu hàm nhiều biến. Nó nhanh chóng được nhiều tác giả cải tiến một cách phong phú để giải quyết các bài toán khó trong thực tiễn với tên gọi chung là tính toán tiến hoá. Việc giải quyết các bài toán tuy đa dạng nhưng thủ tục áp dụng vẫn dựa trên lược đồ GA cổ điển.

1. *Thuật toán di truyền cổ điển*

GA cổ điển được Holland giới thiệu để giải bài toán tối ưu:

$$\mathrm{Max}\{f(x) \mid x \in M\},$$

ở đây M là hình hộp trong không gian số thực n-chiều, $f(x)$ dương với mọi $x \in M$.

Thủ tục GA được thực hiện như sau:

- Mỗi x trong M được mã hoá tương ứng bởi một xâu nhị phân độ dài m: $z = (z_1, z_2, \ldots, z_m)$ gọi là nhiễm sắc thể (còn gọi là cá thể), mỗi z_i được gọi là một gene và chỉ nhận một trong hai giá trị 0 hoặc 1.

  Ví dụ, về một nhiễm sắc thể trong GA cổ điển:

0	0	1	1	1	0	1	0	1	1

  Mỗi kiểu gene (tức một nhiễm sắc thể cụ thể) biểu thị một lời giải có thể của bài toán, một quá trình tiến hoá được thực hiện trên một quần thể (một tập hợp nhiễm sắc thể) tương đương với sự tìm kiếm trong không gian các lời giải có thể.

  Sự tìm kiếm này đòi hỏi sự cân bằng giữa hai mục đích: tìm lời giải tốt nhất và khám phá không gian tìm kiếm.

- Thuật toán di truyền cổ điển thực hiện tìm kiếm theo nhiều hướng bằng cách duy trì một tập lời giải có thể, khuyến khích sự hình thành và trao đổi thông tin giữa các hướng. Tập lời giải trải qua các quá trình tiến hoá và cuối cùng cho ta một lời giải đủ tốt tuỳ theo yêu cầu. Tại mỗi thế hệ các lời giải tương đối tốt được tái sinh, trong khi đó các lời giải tương đối tồi bị loại bỏ dần.

- Trên tập nhiễm sắc thể, người ta xác định hàm *eval* để đánh giá độ "thích nghi" của mỗi cá thể hay chính là độ tốt, xấu của từng lời giải (hàm này đóng vai trò của một môi trường sống trong thuyết tiến hoá):

  $eval(z) = f(x)$, trong đó x là vectơ tương ứng với z.

- Tạo quần thể ban đầu $P(0)$ gồm N phần tử và thực hiện quá trình tiến hoá theo cấu trúc:

Procedure GA
Begin
  $t \leftarrow 0$;
  Khởi tạo $P(t)$;
  Đánh giá $P(t)$;
  Repeat
    $t \leftarrow t + 1$;
    Chọn lọc $Q(t)$ từ $P(t - 1)$; *// nhờ bánh xe xổ số*
    Tái tạo $P(t)$ từ $Q(t)$; *// nhờ các toán tử di truyền*
    Đánh giá $P(t)$ và chọn lọc cá thể tốt nhất;
  Until điều_kiện_kết_thúc;
  Biểu diễn lời giải;
End;

Quá trình tiến hoá được diễn ra trong vòng lặp: Tại thế hệ thứ t thuật toán duy trì một tập lời giải $P(t) = \{x_1^t, x_2^t, ..., x_N^t\}$. Mỗi lời giải x_i^t được đánh giá "độ thích nghi". Một tập lời giải mới được xây dựng (vòng lặp thứ $t + 1$) bằng cách "chọn lọc" các cá thể thích nghi hơn, ta được tập lời giải trung gian. Sau đó một số cá thể trong tập lời giải đã được chọn bị biến đổi bằng các toán tử di truyền để tạo thành lời giải mới cho thế hệ thứ $(t + 1)$.

Các thủ tục chọn lọc một quần thể theo phương pháp *bánh xe xổ số* và tái tạo nhờ các *toán tử di truyền* được thực hiện như sau:

a) Thủ tục chọn lọc

Phép chọn lọc là một quá trình chọn lọc các cá thể để tham gia vào các pha tiếp theo của quá trình tiến hoá. Việc lựa chọn các cá thể từ một quần thể dựa trên độ thích nghi của cá thể đó, nghĩa là những cá thể nào có giá trị hàm thích nghi cao có nhiều khả năng được chọn để tái tạo trong các thế hệ tiếp theo.

Với mỗi quần thể $P(t-1)$ gồm N nhiễm sắc thể: $P(t-1) = \{v_1, v_2, \ldots, v_n\}$ ta xây dựng *bánh xe xổ số* và thực hiện quá trình chọn lọc:

- Bánh xe xổ số

  Đánh giá độ phù hợp toàn phần: $F = \sum_{i=1}^{N} eval(v_i)$;

  Tính các xác suất chọn p_i của nhiễm sắc thể v_i: $p_i = \dfrac{eval(v_i)}{F}$;

  Tính các xác suất tích luỹ q_i của các nhiễm sắc thể v_i: $q_i = \sum_{j=1}^{i} q_j$.

- Quá trình chọn lọc quần thể $Q(t)$ từ $P(t-1)$ dựa vào bánh xe xổ số được thực hiện theo cách sau:

  Đối với mỗi số tự nhiên $k \in \{1, 2, \ldots, N\}$ tạo một số ngẫu nhiên $r_k \in [0,1]$.

  Nếu $q_i \geq r_k \geq q_{i-1}$ thì chọn v_i thuộc $Q(t)$. Hiển nhiên, ở đây mỗi nhiễm sắc thể có thể được chọn nhiều lần và $Q(t)$ vẫn có N phần tử. Các cá thể v có độ thích nghi $eval(v)$ lớn sẽ có khả năng được lựa chọn nhiều hơn.

b) Quá trình tái tạo

Quá trình tái tạo dựa trên các toán tử di truyền: *tương giao chéo* và *biến dị*.

- Các toán tử di truyền

Toán tử tương giao chéo (hay còn được gọi là phép lai ghép), kết hợp các đặc tính trên nhiễm sắc thể cha và mẹ để tạo thành hai cá thể mới bằng cách tráo đổi các đoạn gene tương ứng trên các nhiễm sắc thể của cha và mẹ.

Với hai nhiễm sắc thể $x = (x_1, x_2, \ldots, x_m)$ và $y = (y_1, y_2, \ldots, y_m)$ chọn điểm tương giao k (có thể ngẫu nhiên) ta sẽ sinh được hai nhiễm sắc thể mới:

$$x' = (x_1, \ldots, x_k, y_{k+1}, \ldots, y_m) \text{ và } y' = (y_1, \ldots, y_k, x_{k+1}, \ldots, x_m).$$

Ví dụ. Với hai nhiễm sắc thể cha và mẹ

Parent1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Parent2	1	1	1	0	0	1	0	0	1

Chọn điểm tương giao $k = 3$, thì sau khi thực hiện trao đổi tương giao chéo các nhiễm sắc thể của bố, mẹ ta được hai nhiễm sắc thể mới:

Child1	[illegible]	[illegible]	[illegible]	0	0	1	0	0	1
Child2	1	1	1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Toán tử biến dị (hay còn gọi là đột biến) là sự sửa đổi một vài gene của một nhiễm sắc thể được chọn.

Nếu gene x_k của nhiễm sắc thể $x = (x_1, \ldots, x_m)$ biến dị thì ta được nhiễm sắc thể mới $x' = (x_1, \ldots, x_{k-1}, 1 - x_k, x_{k+1}, \ldots, x_m)$.

- Thủ tục tái tạo

Cho trước các sác xuất tương giao chéo p_c và xác suất biến dị p_m.

Đối với mỗi nhiễm sắc thể v_i (i chạy từ 1 đến N) thuộc $Q(t)$, ta tạo ra một số ngẫu nhiên $r \in [0,1]$. Nếu $r < p_c$ thì v_i được đưa vào tập tương giao chéo. Tập này được chia thành cặp, nếu lẻ thì có thể thêm hoặc bớt ngẫu nhiên một nhiễm sắc thể khác và áp dụng toán tử tương giao chéo để tạo nên hậu duệ mới thay thế cho nó.

Sau khi tương giao chéo, đối với mỗi gene của mỗi nhiễm sắc thể tạo ra một số ngẫu nhiên $r \in [0,1]$. Nếu $r < p_m$ thì gene này là biến dị.

Quá trình trên cho ta quần thể $P(t)$ của thế hệ t và được đánh giá để chọn phần tử có độ thích nghi tốt nhất.

2. Đánh giá về GA

Các đánh giá về sự hội tụ của GA còn rất ít. Các kết quả đạt được chủ yếu dựa trên định lí về lược đồ chứng minh sự hội tụ theo xác suất tới lời giải tối ưu của bài toán. Tuy nhiên, về mặt thực hành, giải thuật di truyền vẫn là một giải thuật được ưa thích để giải các bài toán khó trong thực tế và cho lời giải đủ tốt. Đối với các bài toán đã có phương pháp giải tốt bằng phương pháp truyền thống thì GA kém hiệu quả hơn.

3. Tính toán tiến hoá

Khi bài toán có miền chấp nhận được lớn trong không gian nhiều chiều thì độ rộng của mỗi nhiễm sắc thể lớn nên việc áp dụng GA cổ điển rất khó khăn, đặc biệt khi có các ràng buộc phức tạp thì các toán tử di truyền theo kiểu đã nêu tỏ ra kém hiệu quả sử dụng. Hàng loạt các phát triển phong phú của GA cổ điển về kiểu gene, cấu trúc nhiễm sắc thể và các toán tử di truyền đã được đề xuất và ứng dụng có hiệu quả để giải các bài toán khác nhau. Mặt khác để tìm lời giải cho các bài toán khó trong thực tiễn, người ta đưa ra các hàm "đích" đo độ "thích nghi" của mỗi lời giải tiềm năng và áp dụng GA để tìm các lời giải. Các phát triển đó có tên gọi khác nhau chẳng hạn:

Chiến lược tối ưu: Chiến lược này được phát triển ở Đức, bởi Rechenberg và Schwefel. Ý tưởng thuật toán bắt chước các nguyên tắc tiến hoá trong tự nhiên để tạo ra một phương pháp giải các bài toán tối ưu với các tham số thay đổi liên tục. Gần đây ý tưởng đó mở rộng cho các bài toán rời rạc, trong đó cách biểu diễn gene trên các vectơ thực được sử dụng để xử lí các ràng buộc và giảm khối lượng xử lí dữ liệu.

Lập trình tiến hoá: Ý tưởng của kĩ thuật này là ứng dụng GA trong trí tuệ nhân tạo nhờ sử dụng các ôtômat hữu hạn để tạo ra các chương trình thích ứng với yêu cầu, giúp tạo hành vi cho các rô-bốt hoặc agent thông minh.

Chương trình tiến hoá: Ứng dụng GA để tìm lời giải cho các bài toán khác nhau khi không gian tìm kiếm phức tạp. Chúng có tên gọi chung là tính toán tiến hoá (Evolutionary Computation - EC). Lược đồ chung của thuật toán tiến hoá là:

- Chọn một kiểu gene và cấu trúc nhiễm sắc thể thích hợp cho các lời giải tiềm năng của bài toán. Xây dựng thủ tục chuyển đổi giữa chúng.
- Đưa ra một hàm để đo "độ tốt" của các lời giải tiềm năng nhờ đó xác định hàm thích nghi cho EC.
- Xác định các toán tử di truyền (tương giao chéo và biến dị) thích hợp cho từng bài toán và các ràng buộc của chúng. Các toán tử có thể nhiều để vận dụng thích hợp xử lí ràng buộc.
- Xây dựng thủ tục tạo quần thể ban đầu và lặp nhiều lần quá trình chọn lọc, tái tạo để nhận được lời giải.

Có thể mô tả thuật toán như sau:

```
Procedure EC;
Begin
    t ← 0;
    Khởi tạo P(t);          //khởi tạo quần thể
    Đánh giá P(t);          //đánh giá độ thích nghi
    while not kết_thúc do         //vòng lặp tiến hoá
        begin
            P'(t) ← Biến đổi (P(t)); //biến đổi quần thể
            Đánh giá P'(t); // đánh giá độ thích nghi mới
            P(t + 1) ← Chọn lọc (P'(t)); //tạo ra thế hệ con mới
            t ← t + 1;
        end;
End;
```

Để sử dụng thuật toán tiến hoá, khó khăn chính là chọn lọc được kiểu gene, cấu trúc nhiễm sắc thể và các thủ tục tạo mẫu, toán tử di truyền thích hợp để xử lí các ràng buộc. Còn các khó khăn trong xây dựng phần mềm thể hiện giải thuật là tổ chức dữ liệu.

III. Phương pháp tối ưu hoá đàn kiến

Tối ưu hoá đàn kiến (*Ant Colony Optimization - ACO*) là cách tiếp cận meta-heuristic tương đối mới được đề xuất bởi Dorigo vào năm 1991 mô phỏng hành vi tìm đường đi từ tổ tới nguồn thức ăn và ngược lại của con kiến trong tự nhiên để giải gần đúng các bài toán tối ưu tổ hợp NP-khó.

Trên đường đi của mình các con kiến để lại một vết hoá chất được gọi là vết mùi (*pheromone trail*), đặc điểm sinh hoá học của vết mùi này là có khả năng ứ đọng, bay hơi và là phương tiện giao tiếp báo cho các con kiến khác thông tin về đường đi đó một cách gián tiếp. Các con kiến sẽ lựa chọn đường đi nào tồn đọng lượng mùi hay có cường độ vết mùi lớn nhất tại thời điểm lựa chọn để đi, nhờ cách giao tiếp mang tính gián tiếp và cộng đồng này mà đàn kiến trong tự nhiên tìm được đường đi ngắn nhất trong quá trình tìm thức ăn mang về tổ và ngược lại.

Theo ý tưởng này, các thuật toán ACO sử dụng thông tin heuristic kết hợp thông tin học tăng cường qua các vết mùi của các con kiến nhân tạo (*artificial ant*) để giải các bài toán tối ưu tổ hợp NP-khó bằng cách đưa về bài toán tìm đường đi tối ưu trên đồ thị cấu trúc tương ứng được xây dựng từ đặc điểm của từng bài toán cụ thể. Thuật toán ACO đầu tiên là hệ kiến (*Ant System - AS)* giải bài toán Người bán hàng (*Travelling Salesman Problem - TSP*), đến nay các thuật toán ACO đã áp dụng một cách phong phú để giải nhiều bài toán tối ưu tổ hợp khác nhau và hiệu quả nổi trội của nó đã được chứng tỏ bằng thực nghiệm.

Hệ kiến là thể hiện đầu tiên và điển hình của các thuật toán ACO, hầu hết các thuật toán ACO hiện dùng đều được phát triển từ thuật toán này. Vì vậy, trước tiên chúng ta tìm hiểu về thuật toán AS và các cải tiến quan trọng của nó là ACS và MMAS.

1. AS và bài toán TSP

a) Bài toán TSP

TSP là bài toán tối ưu tổ hợp điển hình, nó được phát biểu như sau : Với n thành phố đã cho, người bán hàng cần tìm một chu trình có đường đi ngắn nhất qua mỗi thành phố đúng một lần. Ta có thể đưa TSP về bài toán tìm đường đi trên đồ thị đầy (có hướng hoặc không) $G=(V,E)$, trong đó tập đỉnh V (được đánh số từ 1 đến n) kí hiệu là tập các thành phố và E là tập các cạnh (i,j) biểu thị đường đi nối các đỉnh $i,j \in V$ và có độ dài $d_{i,j}$ tương ứng ($d_{i,j}$ là khoảng cách từ thành phố i tới thành phố j và $d_{i,j} \neq d_{j,i}$ nếu đó là đường một chiều).

Như vậy, với đồ thị không đối xứng sẽ có $(n-1)!$ đường đi chấp nhận được và $\frac{(n-1)!}{2}$ với đồ thị đối xứng. Với n lớn thì ta không thể tìm hết các đường đi và chỉ có thể tìm được một lời giải đủ tốt bằng các phương pháp tìm kiếm địa phương, tìm kiếm heuristic, tính toán tiến hoá hay là các phương pháp kết hợp giữa chúng. TSP là bài toán có nhiều ứng dụng và vẫn được xem là bài toán mẫu để kiểm tra hiệu quả của các thuật toán tối ưu tổ hợp. Chúng ta sẽ tìm hiểu các thuật toán đàn kiến dựa trên tư tưởng lời giải của bài toán TSP.

b) Thuật toán AS

Để giải bài toán TSP, AS sử dụng các vệt mùi τ_{ij} gắn với mỗi cạnh (i,j) và ban đầu được khởi tạo bởi giá trị τ_0. Có m con kiến nhân tạo, và tại mỗi bước lặp t của thuật toán, chúng thực hiện các thủ tục xây dựng lời giải và cập nhật mùi.

Xây dựng lời giải

Ở bước khởi tạo, mỗi con kiến được đặt ngẫu nhiên tại một đỉnh xuất phát và lần lượt đi thăm các đỉnh còn lại để xây dựng đường đi với quy tắc chuyển trạng thái như sau :

Quy tắc chuyển trạng thái : con kiến thứ k đang ở đỉnh i sẽ chọn đỉnh j tiếp theo với xác suất

$$P_{ij}^{k}(t)=\begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}}, & j\in N_i^k \\ 0, & \text{trong trường hợp ngược lại} \end{cases} \tag{10.1}$$

trong đó α, β là hai tham số thể hiện : xác suất lựa chọn cạnh (i,j) tỉ lệ thuận với cường độ vệt mùi $\tau_{ij}(t)$ tại bước lặp đang xét và thông tin heuristic $\eta_{ij}=\frac{1}{d_{ij}}$ (nếu $\alpha = 0$ con kiến sẽ lựa chọn đỉnh tiếp theo dựa vào tư tưởng tham lam thông thường, tức là cạnh nào ngắn nhất sẽ được ưu tiên chọn trước, nếu $\beta = 0$ sự lựa chọn sẽ chỉ phụ thuộc vào cường độ vệt mùi, bài toán dễ rơi vào trường hợp tối ưu địa phương), N_i^k là tập các đỉnh chưa được con kiến thứ k đi qua khi nó đang ở đỉnh i. Tiếp tục như vậy, con kiến sẽ tìm được một chu trình chấp nhận được.

Cập nhật mùi

Sau khi các con kiến xây dựng xong các lời giải, các vệt mùi sẽ được cập nhật. Với thuật toán AS, các vệt mùi trên các cạnh sẽ được bay hơi một đại lượng theo một tham số $\rho \in (0, 1)$ (được gọi là tham số bay hơi mùi) và mỗi con kiến sẽ đặt mùi tại các cạnh mà chúng đi qua theo công thức :

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad (10.2)$$

trong đó

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{1}{L^k(t)}, \text{ nếu } (i, j) \text{ thuộc hành trình của con kiến thứ } k \\ 0, \text{ trong trường hợp ngược lại.} \end{cases} \qquad (10.3)$$

($L^k(t)$ là độ dài hành trình của con kiến thứ k tại bước lặp t).

Tham số ρ được sử dụng để tích tụ không giới hạn các vệt mùi và cho phép lãng quên những cạnh ít sử dụng (các cạnh không thuộc hành trình của con kiến nào sẽ có cường độ vệt mùi giảm rất nhanh theo hàm mũ của số vòng lặp).

Độ phức tạp của AS

Độ phức tạp của thuật toán là $O(Nc * n^3)$, trong đó Nc là số vòng lặp chính của thuật toán và n là số đỉnh của đồ thị.

Ưu và nhược điểm của AS

Ưu điểm:

- Việc tìm kiếm ngẫu nhiên dựa trên các thông tin heuristic làm cho phép tìm kiếm linh hoạt và mềm dẻo trên không gian rộng hơn phương pháp heuristic sẵn có, do đó cho ta lời giải tốt hơn và có thể tìm được lời giải tối ưu.
- Sự kết hợp được việc học tăng cường (*reinforcement learning*) trong đó những lời giải tốt hơn sẽ được sự tăng cường cao hơn thông qua thông tin về cường độ vết mùi cho phép ta từng bước thu hẹp không gian tìm kiếm và vẫn không loại bỏ các lời giải tốt, do đó nâng cao chất lượng thuật toán.

Nhược điểm: Khi số đỉnh của đồ thị lớn thì cường độ vết mùi trên mỗi cạnh không thuộc lời giải tốt (hay ít được con kiến lựa chọn) sẽ nhanh chóng giảm dần về 0, như vậy cơ hội khám phá hay tìm kiếm ngẫu nhiên của thuật toán sẽ giảm mà đây là một trong những điểm mạnh của các thuật toán mô phỏng tiến hoá tự nhiên nên thuật toán hệ kiến AS kém hiệu quả.

Chất lượng thuật toán phụ thuộc nhiều vào chất lượng của thông tin heuristic, điều mà chúng ta rất khó cải thiện.

2. Các cải tiến của AS

Có rất nhiều các cải tiến của thuật toán AS được đề xuất, ở đây chúng ta tìm hiểu hai cải tiến quan trọng và được ưa dùng nhất hiện nay là thuật toán ACS và thuật toán MMAS. Hiệu quả của hai thuật toán này được xem là tương đương nhau nhưng với các bài toán ứng dụng, thuật toán MMAS được sử dụng nhiều hơn bởi tính đơn giản và dễ áp dụng.

a) Thuật toán ACS (Ant Colony System)

Thuật toán ACS là thuật toán cải tiến của AS dựa trên ý tưởng tăng tầm quan trọng của các thông tin được tích luỹ bởi các con kiến trước với hi vọng mở rộng không gian tìm kiếm.

Việc này được thực hiện nhờ hai cải tiến ở quy tắc chuyển trạng thái xây dựng lời giải và quy tắc cập nhật mùi :

Quy tắc chuyển trạng thái : Cung cấp một cách trực tiếp để cân bằng giữa *sự khám phá* cạnh mới và *sự khai thác* độ ưu tiên và thông tin được tích luỹ của bài toán. Giả sử con kiến k đang ở đỉnh i, nó sẽ chọn đỉnh s tiếp theo nhờ quy tắc sau:

$$s = \begin{cases} \arg\max_{j \in N_i^k} \left\{ \tau_{ij}(t)\eta_{ij}^{\beta} \right\}, \text{ nếu } q \le q_0 \\ j, \text{ trong trường hợp ngược lại} \end{cases} \qquad (10.4)$$

trong đó q là giá trị ngẫu nhiên phân phối đều còn q_0 là một tham số còn j là đỉnh được chọn theo công thức (1) với $\alpha = 1$.

Cập nhật mùi : Có hai loại cập nhật mùi

- *Cập nhật mùi địa phương* : trong khi xây dựng lời giải của mình, mỗi con kiến sẽ được cập nhật mùi cho cạnh (i, j) mà nó đi qua theo công thức

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0 \qquad (10.5)$$

- *Cập nhật mùi toàn cục* : được áp dụng cho những cạnh thuộc đường đi ngắn nhất $w^*(t)$ theo công thức :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + L_{gl}^{-1}(t) \qquad (10.6)$$

trong đó $L_{gl}(t)$ là độ dài của hành trình $w^*(t)$.

Như vậy, việc tìm kiếm sẽ tập trung vào các cạnh thường sử dụng và ưu tiên những cạnh thuộc lời giải tốt nhất. Hiệu quả của ACS qua thực nghiệm cho thấy nổi trội hơn hẳn so với thuật toán AS ban đầu.

Nhược điểm của thuật toán ACS là nếu cạnh không được các con kiến nào dùng tới kể từ thời gian t thì nó vẫn được giữ nguyên mức cường độ mùi trong khi đáng lẽ nó phải giảm bớt so với các cạnh hay được sử dụng.

b) Thuật toán MMAS (Max-Min Ant System)

Thuật toán AS có thể được cải tiến bằng cách chỉ cho phép duy nhất con kiến tốt nhất cập nhật mùi trong mọi vòng lặp của thuật toán. MMAS chỉ cho phép một con kiến tốt nhất (hoặc global-best là con kiến có hành trình tốt nhất $w(t)$ kể từ khi bắt đầu thuật toán hoặc iteration-best là con kiến có hành trình tốt nhất $w(t)$ tại vòng lặp hiện thời) cập nhật mùi. Để tránh hiện tượng tắc nghẽn (*stagnation*) do con kiến lựa chọn đi lựa chọn lại một đường đi tốt do nồng độ mùi tập trung ở đó quá cao, MMAS đưa vào hai cận trên ($\tau_{\max}$) và cận dưới ($\tau_{\min}$) để khống chế nồng độ vết mùi trên mỗi cạnh (đây là lí do thuật toán được gọi là hệ kiến MAX-MIN).

MMAS sử dụng cận dưới $\tau_{\min}$ là giới hạn nhỏ nhất và cận trên $\tau_{\max}$ là giới hạn lớn nhất cho nồng độ các vệt mùi, tức là những cạnh ít sử dụng có xác suất được lựa chọn là rất nhỏ nhưng vẫn lớn hơn 0 và hạn chế khả năng lựa chọn mãi một cạnh tốt của các con kiến. Điều này tránh cho thuật toán khỏi hiện tượng tắc nghẽn tìm kiếm đặc biệt với số vòng lặp lớn dẫn đến mở rộng khả năng khám phá cho các con kiến. Các vết mùi ban đầu được khởi tạo bằng $\tau_{\max}$ cho tất cả các cạnh. Sau mỗi vòng lặp, tất cả các cạnh đều bị bay hơi đi theo một tỉ lệ như nhau (được cho bởi tham số bay hơi ρ).

Chỉ có các vệt mùi trên các cạnh thuộc hành trình tốt nhất được gia cố thêm một lượng nồng độ (trong chứng minh lí thuyết, hành trình tốt nhất được chọn là *g*-best còn trong thực nghiệm *i*-best hiệu quả hơn do tăng cường được khả năng khám phá). Như vậy, cường độ vết mùi trên các cạnh *tồi* hay *ít sử dụng* sẽ giảm chậm còn trên các cạnh tốt được đặt thêm một lượng nên sẽ vẫn được các con kiến lựa chọn nhiều hơn.

Trong hệ này, thủ tục xây dựng lời giải thực hiện như trong AS (được thực nghiệm với tham số α ở công thức (10.1) được chọn bằng 1). Sau mỗi bước lặp vết mùi trên mỗi cạnh được cập nhật toàn cục theo công thức :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta_{ij} \tag{10.7}$$

trong đó $$\Delta_{ij} = \begin{cases} \rho L^{-1}(w(t)), & (i,j) \in w(t) \\ \max\{\tau_{\min} - (1-\rho)\tau_{ij}, 0\}, & \text{trong trường hợp ngược lại.} \end{cases} \tag{10.8}$$

Nhược điểm của thuật toán này là sẽ tập trung tìm kiếm vào các cạnh thuộc lời giải tốt nhất tìm được mà không phân biệt các cạnh không dùng được với các cạnh có được dùng nhưng không thuộc lời giải tốt vì vậy hạn chế khả năng khám phá nếu $\tau_{\min}$ chọn bé. Còn nếu chọn $\tau_{\min}$ lớn thì thuật toán sẽ gần với tìm kiếm ngẫu nhiên dựa trên thông tin heuristic nhưng lại giảm khả năng học tăng cường. Và với việc đặt $\tau_{\min} = \frac{1}{2}\tau_{\max}$ làm giảm đáng kể khả năng học tăng cường của thuật toán.

3. Thuật toán tổng quát

Với bài toán tổng quát trên, về lí thuyết ta có thể áp dụng thủ tục mở rộng để xây dựng X^* và chọn lời giải tốt nhất bằng phương pháp vét cạn nhưng trên thực tế, do sự bùng nổ tổ hợp thì với số phần tử *n* của *C* lớn thì không thực hiện được và bài toán được xét thuộc dạng NP-khó. Thông thường ta sẽ có các phương pháp heuristic để tìm lời giải đủ tốt cho bài toán. Các thuật toán ACO kết hợp thông tin heuristic này với phương pháp học tăng cường nhờ mô phỏng hành vi của đàn kiến để tìm lời giải tốt hơn.

Giả sử với mỗi cạnh $(i,j) \in C$ có trọng số heuristic $h_{i,j}$ để định hướng chọn thành phần mở rộng là *j* khi thành phần cuối cùng của x_k là *i* theo thủ tục nêu trên ($h_{i,j} > 0\ \forall (i,j)$). Đàn kiến *m* con sẽ xây dựng lời giải trên đồ thị đầy đủ có trọng số $G = (V, E, H, \tau)$ trong đó V là tập đỉnh tương ứng với tập thành phần C ở trên, E là tập các cạnh, H là vectơ các trọng số heuristic của cạnh tương ứng còn τ là vectơ vết mùi tích luỹ được ban đầu với khởi tạo bằng τ_0. Đồ thị G được gọi là đồ thị cấu trúc của bài toán.

Với điều kiện dừng đã chọn (giả sử như là với số lần lặp N_c xác định trước) các thuật toán được mô tả hình thức như sau :

procedure *Thuật toán ACO cho các bài toán tối ưu tổ hợp tĩnh*

Đặt các tham số và vệt mùi khởi tạo

while (chưa gặp điều kiện dừng) **do**

Xây dựng các lời giải

Áp dụng tìm kiếm địa phương (có thể có hoặc không)

Cập nhật các vết mùi.

end;

end;

a) Xây dựng lời giải

Sau khi khởi tạo các tham số và cường độ mùi ban đầu, các con kiến thực hiện thủ tục xây dựng lời giải. Trong mỗi lần lặp t, mỗi con kiến chọn ngẫu nhiên một đỉnh xuất phát trong C_0 và kết hợp thông tin heuristic với thông tin mùi để xây dựng lời giải ngẫu nhiên theo thủ tục mở rộng tuần tự nêu ở phần trước với xác suất chọn đỉnh tiếp theo như sau :

Quy tắc chuyển trạng thái :

Giả sử con kiến s đã xây dựng $x_k = < u_0, ..., u >$, nó sẽ chọn đỉnh y thuộc $J(x_k)$ để $x_{k+1} = < u_0, ..., u, y >$ với xác suất :

$$P(y|\tau, x_k) = \begin{cases} \dfrac{\tau_{u,y}^{\alpha} h_{u,y}}{\sum\limits_{j \in J(x_k)} \tau_{u,y}^{\alpha} h_{u,j}}, & y \in J(x_k) \\ 0, \text{ trong trường hợp ngược lại.} \end{cases} \tag{10.9}$$

Quá trình này tiếp tục cho tới khi mỗi con kiến l đều tìm được lời giải chấp nhận được $x(l) \in X^*$ và do đó $s(l) = \varphi(x(l)) \in S$. Để tiện trình bày, về sau ta sẽ xem

$x(l)$ và $s(l)$ như nhau đều là thể hiện của lời giải chấp nhận được và không phân biệt X^* với S. Kí hiệu $w(t)$ là trạng thái tốt nhất các con kiến tìm được cho tới lúc này và $w^i(t)$ là lời giải tốt nhất trong bước lặp, ta sẽ quy ước chọn $w(t) = w(t-1)$ trong trường hợp $w^i(t) = w(t-1)$ và quan tâm đến lời giải gần đúng $w(t)$ này. Trong lí thuyết để tiện cho việc chứng minh, các tác giả đã chọn $w(t)$ còn trong thực nghiệm $w^i(t)$ lại tỏ ra hiệu quả hơn do việc này tăng cường tính khám phá của thuật toán. Do giả thiết của bài toán và để tiện cho trình bày, về sau ta không phân biệt mỗi $x \in X^*$ với trạng thái $s \in S$ tương ứng.

b) Quy tắc cập nhật mùi

Giả sử g là một hàm giá trị thực xác định trên S sao cho $0 < g(s) < \infty, \forall s \in S$ và $g(s) > g(s')$ nếu $f(s) < f(s')$, khi đó ở mỗi bước lặp cường độ vết mùi sẽ thay đổi theo một trong hai quy tắc thường dùng sau đây, được gọi là các quy tắc cập nhật mùi hai mức.

Quy tắc ACS : Quy tắc này phỏng theo quy tắc cập nhật mùi của thuật toán hệ đàn kiến ACS bao gồm cả cập nhật địa phương và cập nhật toàn cục.

- *Cập nhật mùi địa phương* : nếu con kiến l thăm cạnh (i, j), tức là $(i, j) \in s(l)$ thì cạnh này sẽ thay đổi mùi theo công thức :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_1 \qquad (10.10)$$

- *Cập nhật mùi toàn cục* : Cập nhật mùi toàn cục chỉ áp dụng cho các cạnh thuộc $w(t)$ như sau :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho g(w(t)) \qquad \forall (i, j) \in w(t) \qquad (10.11)$$

Quy tắc MMAS : Quy tắc này thực hiện theo cách cập nhật mùi của thuật toán MMAS, sau khi mỗi con kiến xây dựng xong lời giải ở mỗi bước lặp, vết mùi được thay đổi theo công thức sau :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta_{ij} \qquad (10.12)$$

trong đó $\Delta_{ij} = \begin{cases} \rho L^{-1}(w(t)), & (i, j) \in w(t) \\ \max\{\tau_{\min} - (1-\rho)\tau_{ij}, 0\}, & \text{trong trường hợp ngược lại.} \end{cases}$

4. Các thuật toán và các tham số

Do chất lượng thuật toán tuỳ thuộc nhiều vào chất lượng của thông tin heuristic mà chất lượng của của các phương pháp heuristic là khó can thiệp, vì vậy ta sẽ quan tâm tới việc tổ chức cấu trúc học, đặc biệt là cách cập nhật mùi và chọn các tham số để nâng cao chất lượng thuật toán. Dưới đây là các nhận định về các thuật toán và tham số.

a) Các thuật toán

- *Các thuật toán theo quy tắc cập nhật mùi ACS* : Theo quy tắc này nếu khởi tạo mùi $\tau_0 = \tau_{\min}$ thì mức mùi sẽ dần về τ_1. Như vậy, việc tìm kiếm sẽ tập trung vào các cạnh thường sử dụng và ưu tiên những cạnh thuộc lời giải tốt nhất. Mức độ tập trung tìm kiếm có thể điều khiển bởi tỉ lệ độ lệch giữa τ_0, τ_1 và $g(w(t))$. Nhược điểm của thuật toán này là nếu cạnh không được các con kiến dùng tới kể từ thời gian T thì chúng vẫn giữ nguyên mức mùi trong khi lẽ ra nó phải giảm bớt so với các cạnh hay sử dụng.
- *Các thuật toán theo quy tắc cập nhật mùi MMAS* : Các thuật toán tập trung tìm kiếm các cạnh thuộc vào lời giải tốt nhất mà không phân biệt các cạnh không được dùng với các cạnh có được dùng nhưng không thuộc lời giải tốt vì vậy sẽ giảm khả năng khám phá nếu $\tau_{\min}$ bé. Nếu chọn $\tau_{\min}$ lớn thì thuật toán sẽ gần với tìm kiếm ngẫu nhiên dựa trên thông tin heuristic và giảm hiệu quả học tăng cường.

b) Các tham số

- *Tham số α* : Nếu chọn α bé thì giảm giá trị thông tin học được thể hiện qua các vết mùi còn nếu α lớn thì xác suất được chọn rất nhạy cảm với các thay đổi của vết mùi. Với các bài toán NP-khó (đặc biệt là bài toán TSP), xác suất để một lời giải tốt được tìm thấy trong một số ít lần lặp là rất bé. Vì vậy nếu α lớn thì ngay từ đầu có nhiều cạnh thuộc các lời giải tốt (thậm chí là tối ưu) do may rủi mà sớm có mức mùi thấp, làm hạn chế hiệu quả thuật toán. Thông thường chọn $\alpha = 1$, việc điều tiết học tăng cường sẽ thực hiện nhờ chọn các tham số ρ, τ_1 thích hợp.
- *Tham số ρ* : Vì $(1-\rho)$ giảm rất nhanh khi ρ tăng trong khi xác suất để một cạnh thuộc một lời giải tốt trong một phép lặp lại bé nên nếu chọn ρ lớn thì thuật toán kém hiệu quả.

- *Số lượng kiến* : Để cho mô hình ổn định và dễ khảo sát toán học thì nên chọn số lượng kiến đủ lớn để các xấp sỉ của các biến ngẫu nhiên có ý nghĩa thống kê.

5. *Một số nguyên lí ứng dụng tối ưu đàn kiến*

a) Thông tin học tăng cường

Một trong những điều quan trọng đầu tiên trong việc áp dụng các thuật toán ACO là công việc xác định thông tin học tăng cường qua các vệt mùi. Ví dụ, khi áp dụng ACO cho bài toán TSP, vệt mùi τ_{ij} được sử dụng trong các ứng dụng ACO vào bài toán này thể hiện độ thích hợp của việc chọn trực tiếp thành phố j sau khi đã thăm thành phố i, hay nó cung cấp thông tin để đánh giá mối quan hệ giữa thành phố i và thành phố j. Ngoài ra ta cũng có thể coi τ_{ij} là đánh giá xem khả năng thành phố i và thành phố j có nằm trong một hành trình hay nó là độ thích hợp vị trí.

Một cách khác, khi áp dụng thuật toán ACO cho bài toán lập lịch cho máy đơn các kết quả đạt được là tốt hơn với vệt mùi τ_{ij} là độ thích hợp khi sắp công việc j vào vị trí i. Đó là do vai trò khác nhau của các hoán vị kết quả trong hai bài toán đó. Trong bài toán TSP, các hoán vị là tuần hoàn và chỉ có duy nhất thứ tự quan hệ giữa các thành phần mới là quan trọng và chú ý rằng hoán vị $\pi = (1\ 2\ \ldots\ n)$ và hoán vị $\pi' = (n\ 1\ 2\ \ldots\ n-1)$ biểu diễn cùng một hành trình. Vì thế, một vị trí quan hệ dựa trên vết mùi thể hiện lựa chọn được gắn với nó. Ngược lại, trong bài toán lập lịch trên máy đơn (cũng như các bài toán lập lịch khác), π và π' biểu diễn hai lời giải khác nhau với những chi phí khác nhau hoàn toàn. Tuy nhiên cần chú ý rằng, trong lí thuyết cả hai lựa chọn đều là có thể bởi vì bất kì lời giải nào trong không gian tìm kiếm với hai cách thể hiện này.

Công việc định nghĩa các vết mùi rất quan trọng và một sự lựa chọn tồi tại một bước sẽ có thể dẫn đến giảm hiệu quả của thuật toán. May mắn là với nhiều bài toán, lựa chọn theo trực giác đồng thời là một lựa chọn tốt, như trường hợp một số ứng dụng của thuật toán ACO. Thỉnh thoảng việc định nghĩa các vết mùi rất phức tạp như trong trường hợp ứng dụng ACO vào bài toán tìm siêu chuỗi (*supersequence)* chung ngắn nhất chẳng hạn.

b) Các thông tin heuristic

Việc sử dụng các thông tin heuristic để xây dựng lời giải của các con kiến trực tiếp theo xác suất là quan trọng vì nó tận dụng các tri thức xác định của bài toán. Các tri thức này thường là những yếu tố có sẵn tĩnh hoặc động. Trong các bài toán tĩnh, các thông tin heuristic η sẽ chỉ được tính một lần trước khi bắt đầu thuật toán, ví dụ trong ứng dụng TSP độ dài d_{ij} của cung nối thành phố i và thành phố j định nghĩa được thông tin heuristic $\eta_{ij} = \frac{1}{d_{ij}}$. Các thông tin heuristic tĩnh có các lợi điểm sau :

(i) Dễ tính toán,

(ii) Chỉ phải tính một lần trước khi bắt đầu thuật toán, và

(iii) Tại mỗi vòng lặp của thuật toán ACO có thể tính trước một bảng các giá trị $\tau_{ij}(t)[\eta_{ij}]^{\beta}$, sẽ tiết kiệm được đáng kể thời gian tính toán.

Trong trường hợp bài toán động, thông tin heuristic phụ thuộc vào các lời giải thành phần được xây dựng cho nên nó được tính tại mỗi bước đi của con kiến. Điều này gây ra chi phí tính toán lớn nhưng có thể đạt được độ chính xác cao trong tính toán các giá trị heuristic chẳng hạn, trong ứng dụng ACO cho bài toán lập lịch.

Cuối cùng, chú ý rằng việc sử dụng các thông tin heuristic là quan trọng cho các dạng thuật toán ACO, nó được kết hợp với tìm kiếm địa phương để cải tiến lời giải. Vì sự thật là tìm kiếm địa phương có được những thông tin chi phí để cải tiến lời giải theo cách trực tiếp hơn. Một cách may mắn các thuật toán ACO khi kết hợp với tìm kiếm địa phương sẽ tìm ra được những lời giải tốt hơn cho những bài toán khó xây dựng được những thông tin heuristic.

c) Kết hợp tìm kiếm địa phương

Với nhiều ứng dụng của các bài toán tối ưu tổ hợp như TSP, bài toán phân chia sản phẩm, hay bài toán định tuyến cho xe cộ, các thuật toán ACO đều có kết quả rất tốt khi kết hợp các thuật toán tìm kiếm địa phương. Các thuật toán tìm kiếm địa phương thường tìm ra các lời giải tối ưu cục bộ và những lời giải đó được dùng để cập nhật vết mùi. Công việc của nó thường là định nghĩa ra được một miền cục bộ của lời giải hiện tại (thông qua các phép biến đổi láng giềng) và thực hiện tìm kiếm địa phương trong phạm vi này để có được lời giải tối ưu toàn cục.

Ở một khía cạnh khác, việc sinh ra các lời giải ban đầu cho các thuật toán tìm kiếm địa phương không phải là một nhiệm vụ đơn giản. Ví dụ, với hầu hết bài toán, việc lặp đi lặp lại các tìm kiếm địa phương từ những lời giải ban đầu được sinh ra ngẫu nhiên không phải là hiệu quả lắm. Trong thực hành, các con kiến kết hợp với các thành phần của lời giải tối ưu cục bộ một cách ngẫu nhiên sẽ sinh ra được những lời giải mới hứa hẹn hơn cho thuật toán tìm kiếm địa phương. Bằng thực nghiệm, có thể thấy sự kết hợp của các lời giải tham lam thích nghi theo xác suất với các thuật toán tìm kiếm địa phương có thể đạt được những kết quả hơn cả.

Mặc dù trong thực tế, việc sử dụng các thuật toán tìm kiếm địa phương là một yếu tố quyết định tính hiệu quả cho các ứng dụng ACO, nhưng chú ý rằng các thuật toán ACO thực hiện dễ dàng thì các thuật toán tìm kiếm địa phương lại không áp dụng một cách đơn giản được. Một ví dụ là bài toán tìm đường trện mạng hay bài toán tìm siêu chuỗi ngắn nhất chẳng hạn.

d) Cân bằng giữa sự khai thác và sự khám phá

Bất kì thuật toán meta-heuristic hiệu quả tốt nào cũng có sự kết hợp giữa việc sử dụng kinh nghiệm tìm kiếm với việc đi thăm các không gian tìm kiếm mới. Trong ACO, có một số cách để đạt được sự cân bằng này, thông thường là qua sự quản lí các vết mùi. Thực tế, các vết mùi đưa ra một sự phân bố xác suất trên không gian tìm kiếm và xác định các phần của không gian tìm kiếm có thể có các lời giải thường xuyên hơn. Chú ý rằng, phụ thuộc vào sự phân bố của các vết mùi, sự phân bố mẫu có thể biến đổi từ sự phân chia cố định đến một sự phân chia khác mà các lời giải đơn có xác suất là 1 còn lại có xác suất là 0. Thực tế, trường hợp sau tương ứng với hiện tượng tắc nghẽn đã được giải thích ở thuật toán MMAS trong phần lịch sử phát triển các thuật toán đàn kiến.

Cách đơn giản nhất để sử dụng kinh nghiệm tìm kiếm của các con kiến là thiết đặt việc cập nhật mùi với một hàm đo hiệu quả của lời giải được cho bởi mỗi con kiến đặc biệt. Chưa hết, xu hướng này thường không có được những lời giải tốt như trong thực nghiệm của bài toán TSP. Vì thế trong nhiều thuật toán ACO chiến thuật *elitist* tìm ra những lời giải tốt nhất trong khi cập nhật các vết mùi được giới thiệu. Một sự khai thác tốt hơn sẽ học được các vết mùi trong quá trình xây dựng lời giải bằng cách áp dụng luật chuyển trạng thái như trong thuật toán ACS.

Mở rộng không gian tìm kiếm trong ACO chính là các thủ tục xây dựng lời giải ngẫu nhiên của các con kiến. Xem xét khi thuật toán ACO không sử dụng thông tin heuristic (bằng cách đặt tham số $\beta = 0$). Trong trường hợp này, quá trình cập nhật mùi sẽ gây ra sự thay đổi từ không gian tìm kiếm mẫu cố định ban đầu thành một không gian mẫu khác. Thêm vào đó, việc mở rộng không gian tìm kiếm sẽ tốt hơn trong các vòng lặp ban đầu của thuật toán và sẽ giảm thiểu được sự tính toán. Hiển nhiên là phải tránh sự tập trung quá mạnh ở các vùng tốt bên ngoài không gian tìm kiếm khiến cho thuật toán ACO gặp phải hiện tượng tắc nghẽn.

Có một số cách để tránh hiện tượng này, ví dụ như trong thuật toán ACS các con kiến sử dụng các luật cập nhật mùi cục bộ khi xây dựng lời giải và thuật toán MMAS sử dụng một cận dưới của các cường độ vệt mùi sao cho cấp độ nhỏ nhất của mở rộng không gian luôn được đảm bảo. MMAS đồng thời có sử dụng việc khởi tạo lại các vệt mùi như là một cách mở rộng không gian tìm kiếm. Kinh nghiệm cho thấy việc khởi tạo lại các vệt mùi kết hợp với các lựa chọn cập nhật vệt mùi giúp tìm kiếm trên các vùng không gian khác nhau. Cuối cùng, một điều quan trọng trong cân bằng giữa khai thác và khám phá đó là các tham số α và β, định nghĩa sự ảnh hưởng của vệt mùi và thông tin heuristic.

e) Sử dụng danh sách ứng cử viên

Một trong những vấn đề khó khăn khi áp dụng các thuật toán ACO đó là gặp phải những tập láng giềng kích thước lớn trong quá trình xây dựng lời giải. Thực tế, một con kiến bao giờ cũng gặp phải một tập láng giềng kích thước lớn để lựa chọn. Các bài toán tốt là các bài toán giảm đáng kể việc xây dựng lời giải và xác suất nhiều con kiến thăm cùng một trạng thái là nhỏ. Ví dụ thực tế trên như là bài toán TSP và SCP với các bộ dữ liệu lớn chẳng hạn.

Trong các trường hợp này, có thể giảm tải các vấn đề nêu trên bằng việc sử dụng các danh sách ứng cử viên. Danh sách ứng cử viên là tập con của tập các láng giềng hứa hẹn của lời giải hiện tại. Chúng được tạo ra dựa trên các thông tin tri thức có sẵn của bài toán nếu có hoặc các thông tin động được sinh ra. Việc sử dụng này cho phép các thuật toán ACO tập trung vào các thành phần ưa thích của chúng và giảm không gian tìm kiếm.

Một ví dụ khi xem xét các ứng dụng ACO cho bài toán TSP. Bài toán TSP thường cho những kết quả tối ưu trên những đồ thị nhỏ bao gồm các thành phố và các cạnh nối giữa chúng sao cho mỗi một thành phố chỉ có một ít các láng giềng gần nhất. Ví dụ như một bài toán TSP trong thư viện TSPLIB là *pr2392.tsp* với 2392 thành phố đạt lời giải tối ưu trên đồ thị con với các láng giềng 8-gần nhất. Kĩ thuật này có thể dùng để sử dụng để xây dựng các danh sách ứng cử viên, lần đầu tiên xuất hiện trong các thuật toán ACO. Đó là danh sách các ứng cử viên bao gồm các láng giềng gần nhất của mỗi thành phố. Trong quá trình xây dựng lời giải, con kiến luôn lựa chọn thành phố trong số các tập láng giềng của nó, chỉ có những thành phố như thế mới được chọn.

Xa hơn, trong các thuật toán ACO việc sử dụng các danh sách ứng cử viên hay các tiếp cận tương tự vẫn chưa được nghiên cứu nhiều. Các kĩ thuật sử dụng danh sách ứng cử viên được sử dụng trong các thuật toán như Tabu Search hay GRASP có thể ứng dụng cho việc áp dụng các chiến lược ứng cử viên hiệu quả cho các thuật toán ACO.

HƯỚNG DẪN GIẢI BÀI TẬP

Chuyên đề 8

8.1. Tìm tập các điểm thoả mãn một trong hai tính chất sau:

a) Là giao điểm của đoạn thẳng và tam giác.

b) Là đầu mút của đoạn thẳng nằm trong tam giác.

Nếu tập trên gồm hai điểm, kết quả là khoảng cách giữa hai điểm đó, ngược lại kết quả bằng 0.

8.2. Tìm tập các điểm thoả mãn một trong hai tính chất sau:

a) Là giao điểm của hai đoạn thẳng, mỗi đoạn thuộc một tam giác.

b) Là đỉnh của một tam giác nằm trong tam giác còn lại.

Tính diện tích bao lồi của tập điểm trên.

8.3. Các đường thẳng chứa các cạnh của các hình chữ nhật chia mặt phẳng toạ độ thành một lưới hình chữ nhật. Ta xét lần lượt các khe ngang của lưới, với mỗi khe, xét các cạnh dọc của các hình chữ nhật theo thứ tự toạ độ x tăng dần. Chỉ quan tâm đến các cạnh phủ hết khe đang xét. Ta có biến đếm s. Nếu gặp cạnh trái của một hình chữ nhật thì tăng s lên 1. Nếu gặp cạnh phải thì giảm s đi 1. Khi s tăng từ 0 lên 1, đánh dấu lại toạ độ x của cạnh đang xét, gọi là x_1. Khi s giảm về 0, đánh dấu toạ độ x của cạnh đang xét, gọi là x_2. Lấy $x_2 - x_1$ nhân với độ rộng khe ngang đang xét rồi cộng vào kết quả.

Độ phức tạp là N^2, với N là số hình chữ nhật.

Để đạt được độ phức tạp $N\log N$ có thể sử dụng cấu trúc Segment tree.

Đề bài và bộ test có tại website http://www.spoj.pl/problems/NKMARS.

8.4. Tìm bao lồi của tập điểm. Hai điểm cần tìm là hai đỉnh xa nhau nhất của bao lồi.

8.5. Giả sử kết quả đang là s. Phân tập điểm thành các tập con sao cho khoảng cách giữa hai điểm bất kì thuộc cùng một tập không vượt quá s. Nếu phân được nhiều hơn K tập thì tăng s. Nếu phân được ít hơn hoặc bằng K tập thì giảm s.

8.6. Xét từng cặp điểm, mỗi cặp điểm viết phương trình đường thẳng đi qua hai điểm. Sau đó đếm số điểm nằm ở hai nửa mặt phẳng chia bởi đường thẳng.

8.7. Áp dụng luồng min-cost.

8.8. Xây dựng đồ thị đầy đủ với các đỉnh là các điểm trong tập, trọng số của một cạnh là khoảng cách giữa hai điểm ứng với hai đầu mút của cạnh. Tìm cây khung của đồ thị.

8.9. Tương tự như bài 8.3 tính diện tích các hình chữ nhật, bài này là tính chu vi.

8.10. Xét tập các tia gồm Ox, Oy, các tia gốc O đi qua một trong các đầu mút của các đoạn thẳng. Sắp xếp các tia trong tập theo góc tạo bởi tia và trục hoành (theo tan). Xét các góc tạo bởi hai tia liên tiếp. Với mỗi góc, xét các đoạn thẳng phủ hết góc và đánh dấu đoạn gần gốc O nhất là nhìn thấy được (lấy khoảng cách gần nhất trong số hai khoảng cách từ O đến hai giao điểm của hai tia và đoạn thẳng). Kết quả là số các đoạn thẳng được đánh dấu.

8.11. Mỗi điểm chỉ có thể nối được với một trong hai điểm sau:

a) Cùng toạ độ x, toạ độ y lớn hơn và gần nhất.

b) Cùng toạ độ x, toạ độ y nhỏ hơn và gần nhất.

Với mỗi điểm, ta đếm xem có bao nhiêu điểm cùng toạ độ x mà toạ độ y lớn hơn toạ độ y của nó. Nếu số lượng đó là lẻ, ta nối điểm đang xét với điểm loại a, nếu không, ta nối với điểm loại b.

8.12. Biểu diễn mỗi đỉnh núi bằng một đoạn thẳng nằm trên trục hoành thể hiện chân núi (cạnh huyền của tam giác vuông). Sắp xếp các đoạn thẳng theo thứ tự toạ độ x của đầu mút bên trái tăng dần. Xét từng đoạn thẳng, nếu đầu mút bên trái của nó không vượt quá đầu mút bên phải xa nhất của các đoạn thẳng đã xét thì đánh dấu ngọn núi ứng với đoạn thẳng đó nhìn thấy được.

8.13. Với mỗi kết quả s, cần kiểm tra xem có tồn tại một hình tròn trong sân sao cho nó không giao với bất cứ hình tròn bán kính R_i nào. Điều này tương đương với việc kiểm tra xem có tồn tại một điểm trong sân sao cho nó không nằm trong bất cứ hình tròn bán kính R_{i+s} nào.

Với hình chữ nhật lớn ban đầu, kiểm tra xem trọng tâm của nó có phải điểm cần tìm không. Nếu không, chia hình chữ nhật thành bốn hình chữ nhật con bằng nhau rồi tiếp tục kiểm tra từng hình bằng cách tương tự.

Chuyên đề 9

9.1. Dùng phương pháp phân tích ngược, xét từ vị trí kết thúc đến vị trí khởi đầu. Vị trí kết thúc là vị trí ứng với số quân bằng 1, đây là vị trí P.

9.2. a) $P = \{p \in X, p \bmod 2 = 0\}$;

b) $P = \{p \in X, p$ chia 9 có dư là 0, 2, 4$\}$;

c) $P = \{p \in X, p$ chia hết cho 3$\}$.

9.3. a) Di chuyển tiếp theo là chiếm ô (3; 1).

b) Có (chứng minh bằng phản chứng).

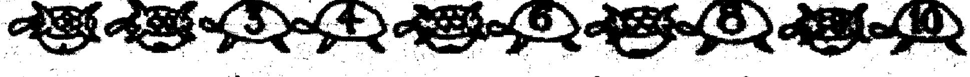
9.4. *Gợi ý.* Biểu diễn số thẻ dưới dạng nhị phân, mỗi lần đi theo thuật thắng là loại đi số 1 bên phải nhất của biểu diễn nhị phân của số thẻ hiện tại. Nếu số N ban đầu không phải là luỹ thừa của 2 thì người đi đầu luôn thắng, ngược lại người thứ hai có nhiều khả năng thắng.

9.5. Nếu số quân ban đầu không là số Fibonacci thì người đi đầu sẽ thắng theo thuật sau: mỗi lần cần chọn số hạng nhỏ nhất trong phân tích số thẻ hiện tại thành tổng các số Fibonacci phân biệt không kề nhau (nếu còn phân tích được, ngược lại thì lấy toàn bộ cọc).

9.6. Thuật toán như trò Nim chuẩn ba cọc, chỉ thêm bước sau: Nếu B di chuyển bằng cách tăng thêm x quân vào một cọc nào đó thì A lại rút đúng x quân đó ra khỏi cọc này.

9.7. a) Có thể coi trò chơi này là một hình thức trá hình của trò chơi Nim: Rùa đứng ở ô n coi như tương ứng một cọc có n quân. Một phép di chuyển rùa: lật ngửa rùa tại ô x, kèm theo lật ngược lại một rùa ở bên trái ô x là ô y ($y < x$). Nếu rùa tại ô y đang đứng mà bị đặt ngửa lại thì tương đương với việc di chuyển hết quân tại hai cọc x và cọc y. Nếu rùa tại ô y đang nằm ngửa lại được lật thành đứng trên bốn chân thì tương đương với việc xoá cọc x và thêm cọc y, hay có thể coi như di chuyển $x - y$ quân khỏi cọc x để còn y quân.

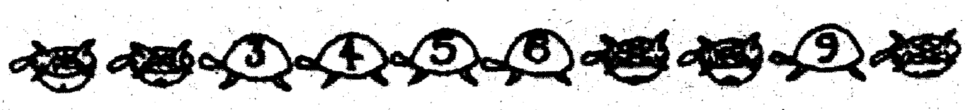
b) Hình vẽ tương ứng với $n = 10$.



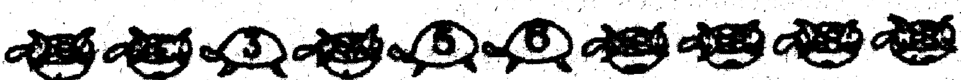
Trạng thái ban đầu là (3, 4, 6, 8, 10) có tổng Nim bằng 3 (khác 0) nên người đi đầu sẽ thắng. Do tổng Nim của 3, 4, 6, 8 bằng 9, do đó cần lật rùa 10 và úp rùa 9 thì chiếm được vị trí mới có tổng Nim bằng 0 ($9 \oplus 9 = 0$).

Các rùa đang bò bây giờ là 3, 4, 6, 8, 9 tạo thành một vị trí *P* (tổng bằng 0).

Giả sử người thứ hai đi tiếp bằng cách lật rùa 8 và 5, các rùa đang bò lại là 3, 4, 5, 6, 9:



Tổng Nim bây giờ là 13, nên chỉ có một phép chuyển tốt cho vị trí này là lật úp 9 và 4 (nghĩa là bỏ đi 13 quân). Trạng thái mới là (3, 5, 6) có tổng Nim bằng 0:



Kết luận. Một bước chuyển trong Nim thành một bước lật rùa như sau: Chúng ta giảm cọc tới một kích thước nhỏ hơn bằng cách lật ngửa một rùa và lật úp một rùa như phép đầu tiên (ngửa 10, úp 9). Nếu loại hai cọc với các kích thước hiện đang có chúng ta phải lật ngửa hai rùa (ví dụ bước đi lần thứ hai của người thứ nhất đã lật 9 và 4). Để loại một cọc, chúng ta chỉ cần lật ngửa một rùa thích hợp ví dụ do 4, 6, 8, 10 có tổng Nim bằng 0 (là một vị trí *P*) nên bước đầu tiên người thứ nhất cũng có thể chỉ cần lật ngửa rùa 3.

9.10. a) $S = \{1, 3, 4\}$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	1	0	1	2	3	2	0	1	0	1	2	3	2	0	1	0	1	2	3

b) *S* là tập các số nguyên tố lẻ

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	0	0	1	1	1	2	2	2	3	0	3	4	1	4	3	0	3	4	1

c) *S* là tập các số nguyên tố

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	0	1	1	2	2	3	3	4	0	0	1	1	2	2	3	3	4	4	5

d) S là tập các số Fibonacci

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	1	2	3	0	1	2	3	4	5	0	1	2	3	0	1	2	3	4	5

e) S là tập các số Lucas

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
$G(x)$	0	1	0	1	2	3	2	3	0	1	0	1	2	3	2	3	0	1

9.11. Hàm Sprague-Grundy của trò chơi *At Most Haft*

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	0	1	0	2	1	3	0	4	2	5	1	6	3	7	0	8	4	9	2

9.12. Hàm Sprague-Grundy của trò chơi *Dim*$^+$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$G(x)$	0	1	2	1	3	1	2	1	4	1	2	1	3	1	2	1	5	1	2	1

9.15. Hàm Sprague-Grundy của trò chơi Wythoff

x \ y	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	0	4	5	3	7	8	6	10
2	2	0	1	5	3	4	8	6	7	11
3	3	4	5	6	2	0	1	9	10	12
4	4	5	3	2	7	6	9	0	1	8
5	5	3	4	0	6	8	10	1	2	7
6	6	7	8	1	9	10	3	4	5	13
7	7	8	6	9	0	1	4	5	3	14
8	8	6	7	10	1	2	5	3	4	15
9	9	10	11	12	8	7	13	14	15	16

9.16. Ta kí hiệu hàm Sprague-Grundy của trò chơi tổng là g, của các trò chơi thành phần lần lượt là $g^{(1)}$, $g^{(2)}$, $g^{(3)}$. Ta có:

$$g(18, 17, 7) = g^{(1)}(18) \oplus g^{(2)}(17) \oplus g^{(3)}(7) = 8 \oplus 5 \oplus 7 = 10 > 0$$

nên (18, 17, 7) là một vị trí N.

Phép chuyển tối ưu là chuyển tới vị trí (6, 17, 7) vì

$$g(6, 17, 7) = g^{(1)}(6) \oplus g^{(2)}(17) \oplus g^{(3)}(7) = 2 \oplus 5 \oplus 7 = 0.$$

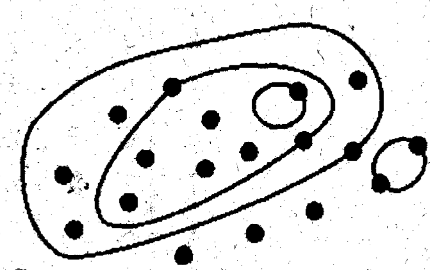
9.17. Hai vị trí kết thúc là cọc 1 quân và cọc 2 quân. Các giá trị hàm Sprague-Grundy cho trò chơi này khi cọc có kích thước nhỏ có thể tính được không khó khăn, nhưng khi kích thước lớn thì việc tính toán rất nhiều thời gian. Dãy giá trị này có tuần hoàn không? Điều này đến nay chưa rõ!

9.18. Giá trị hàm Sprague-Grundy của trò chơi Kayler $g(y + z)$, y là các bội của 12, $z = 0, 1, \ldots, 11$.

z / y	0	1	2	3	4	5	6	7	8	9	10	11
0	**0**	1	2	**3**	1	4	**3**	2	1	4	2	**6**
12	4	1	2	**7**	1	4	**3**	2	1	4	**6**	7
24	4	1	2	8	**5**	4	7	2	1	8	**6**	7
36	4	1	2	**3**	1	4	7	2	1	8	2	7
48	4	1	2	8	1	4	7	2	1	**4**	2	7
60	4	1	2	8	1	4	7	2	1	8	**6**	7
72	4	1	2	8	1	4	7	2	1	8	2	7

Khi $x = 72$, các giá trị bắt đầu là tuần hoàn với chu kì 12, các giá trị của dòng cuối bảng trên được lặp mãi.

9.19. Chơi trò Rims ta có thể coi như chơi theo Nim nhiều cọc.



Ví dụ: Hình bên vẽ vị trí (3, 4, 5) có giá trị Nim là $3 \oplus 4 \oplus 5 = 2$ là vị trí N, cần chuyển đến vị trí P (1, 4, 5) vì $1 \oplus 4 \oplus 5 = 0$. Điều này có thể thực hiện bằng cách vẽ vòng tròn qua hai trong ba chấm ngoài cùng.

9.20. Giá trị hàm Sprague-Grundy của trò chơi cờ Dawson ($N = 18$)

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$g(x)$	0	1	1	2	0	3	1	1	0	3	3	2	2	4	0	5	2	2	3

9.21. Hàm tính tích Nim của hai số x và y:

```
function tichnim(y,x : longint) : longint;
var x1, x2, t,i : longint;
begin
 if x<y then t:= tichnim(x,y) else
 if y=0 then t := 0 else
 if y=1 then t := x else
 if fec(x) then {x=2, 4, 16, 256, 65536,...} begin
 if x=y then t := (x*3) div 2 else
 t := x*y;
 end
 else {x không là luỹ thừa dạng ferma }
 begin {tìm x1, x2 nhỏ hơn hoặc bằng x mà x1 xor x2 = x}
 x1 := 1;
 while x1*2<=x do x1 := 2*x1;
 x2 := x xor x1;
 if (x2=0) then {không tìm được x1, x2 mà x=x1 xor x2, x=x1 là luỹ thừa của 2}
 begin {tìm x=x1.x2}
 i := 0;
 while fecmat[i]<x do inc(i);
 if fecmat[i]>=x then dec(i);
 x1 := fecmat[i];
 x2 := x div x1; {x là luỹ thừa của 2 nên chia hết cho luỹ thừa dạng ferma nhỏ hơn}
{bây giờ x = x1.x2, y.x= y.x1.x2}
 if (random(1000) mod 2 = 0) then
 t := tichnim(tichnim(y,x1),x2)
 else
 t := tichnim(tichnim(y, x2), x1);
 end
 else {y.x=y.(x1+x2)=y.x1 + y.x2}
 t := tichnim(y,x1) xor tichnim(y,x2);
 end;
 tichnim := t;
end;
```

9.25. Trong trò chơi lật xu bốn góc *Turning Corners*, $g(x, y) = x \otimes y$, do đó giải trò chơi này cũng là một cách để tính tích Nim.

```
uses crt;
const max = 100;
type mang = array[0..max, 0..max] of integer;
var g : mang; n,x,y : integer;
 v : array[0..max*max] of integer;
function sg(x,y : integer): integer;
var a,b,i : integer;
begin
 fillchar(v, sizeof(v),0);
 for a:=0 to x-1 do
 for b:=0 to y-1 do
 v[g[x,b] xor g[a,y] xor g[a,b]] := 1;
 i := 0;
 while v[i]=1 do inc(i);
 sg := i;
end;
BEGIN
 clrscr;
 n := 15;
 fillchar(g,sizeof(g),0);
 g[1,1] := 1;
 for y:=2 to n do g[1,y] := y;
 for x:=2 to n do g[x,1] := x;
 for y:=2 to n do
 for x:=2 to n do
 g[x,y]:=sg(x,y);
 for x:=0 to n do
 for y:=0 to n do
 begin
 if y=n then writeln(g[x,y]:4)
 else write(g[x,y]:4);
 end;
 readln;
END.
```

9.26. Chiến thuật tối ưu cho công an A là dừng ở Hải Dương với tần suất $p = \frac{2}{5}$, dừng ở Hải Phòng với tần suất $1 - p = \frac{3}{5}$. Chiến thuật tối ưu cho tội phạm B là dừng ở Hải Dương với tần suất $\frac{3}{5}$, dừng ở Hải phòng với tần suất $\frac{2}{5}$.

9.27. Nếu $t \le 0$ thì a_{11} là điểm yên ngựa. Người thứ nhất chọn hàng 1 theo chiến thuật tối ưu là (1; 0), người thứ hai chọn cột 1 theo chiến thuật tối ưu là (1; 0). Giá trị trò chơi là $v = 0$.

Nếu $0 < t \le 1$ thì a_{21} là điểm yên ngựa. Người thứ nhất chọn hàng 2 theo chiến thuật tối ưu là (0; 1), người thứ hai chọn cột 1 theo chiến thuật tối ưu là (1; 0). Giá trị trò chơi là $v = t$.

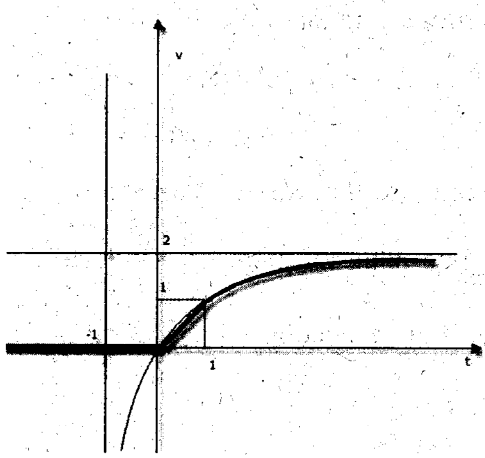
Nếu $t > 1$ không có điểm yên ngựa. Khi đó áp dụng các công thức tính p, q và v:

$$p = \frac{t-1}{t+1},\ q = \frac{1}{t+1},\ v = \frac{2t}{t+1}.$$

Vậy giá trị trò chơi là:

$$v = \left[\begin{array}{lll} 0 & \text{với} & t \le 0 \\ t & \text{với} & 0 < t \le 1 \\ \dfrac{2t}{t+1} & \text{với} & t > 1. \end{array}\right.$$

Đồ thị như sau:



9.28. a) $\begin{pmatrix} 5 & 4 & 1 & 0 \\ 4 & 3 & 2 & -1 \\ 0 & -1 & 4 & 3 \\ 1 & -2 & 1 & 2 \end{pmatrix} \Rightarrow \begin{pmatrix} 4 & 1 & 0 \\ 3 & 2 & -1 \\ -1 & 4 & 3 \\ -2 & 1 & 2 \end{pmatrix} \Rightarrow \begin{pmatrix} 4 & 1 & 0 \\ 3 & 2 & -1 \\ -1 & 4 & 3 \end{pmatrix} \Rightarrow \begin{pmatrix} 4 & 0 \\ 3 & -1 \\ -1 & 3 \end{pmatrix}$

$\Rightarrow \begin{pmatrix} 4 & 0 \\ -1 & 3 \end{pmatrix}$.

Ma trận 2×2 không có điểm yên ngựa. Tính $p = \frac{1}{2}$, $q = \frac{3}{8}$, $v = \frac{3}{2}$.

Suy ra chiến thuật hỗn hợp tối ưu của người thứ nhất là: $p = \left(\frac{1}{2}; 0; \frac{1}{2}; 0\right)$ và chiến thuật tối ưu của người thứ hai là $q = \left(0; \frac{3}{8}; 0; \frac{5}{8}\right)$.

b) $\begin{pmatrix} 10 & 0 & 7 & 1 \\ 2 & 6 & 4 & 7 \\ 6 & 3 & 3 & 5 \end{pmatrix} \Rightarrow \begin{pmatrix} 10 & 0 & 7 \\ 2 & 6 & 4 \\ 6 & 3 & 3 \end{pmatrix} \Rightarrow \begin{pmatrix} 10 & 0 & 7 \\ 2 & 6 & 4 \end{pmatrix}$

(từ ma trận 3×3 chuyển thành ma trận 2×3 vì: $\frac{1}{2}\left(a_{1j} + a_{2j}\right) \geq a_{3j}$ với mọi $j = 1, 2, 3$).

Giả sử người thứ nhất chọn hàng 1 với tần suất p, hàng 2 với tần suất $1 - p$. Khi đó nếu người thứ hai lần lượt chọn cột 1, 2 và 3 thì giá trị người thứ nhất nhận được tương ứng là:

$$v = 10p + 2(1 - p) = 2 + 8p; \ v = 6(1 - p) = 6 - 6p;$$

$$v = 7p + 4(1 - p) = 4 + 3p.$$

Vẽ ba đường thẳng và thấy đường đậm thấp nhất chỉ nằm trên đồ thị ứng với cột 1 và 2 (giao nhau tại điểm $\left(\frac{5}{7}; \frac{30}{7}\right)$, do đồ thị cột 3 nằm hoàn toàn phía trên đường đậm nên cũng có thể dùng ma trận chỉ gồm cột 1 và 2: $\begin{pmatrix} 10 & 0 \\ 2 & 6 \end{pmatrix}$ để tính $p = \frac{2}{7}$, $v = \frac{30}{7}$.

9.29. a) Tính $\text{Min}\left\{\sum_{j=1}^{n} p[i]*a[i,j] \quad \forall i=1,2,...,m\right\} = 3.27027...$

Tính $\text{Max}\left\{\sum_{i=1}^{m} q[j]*a[i,j] \quad \forall j=1,2,...,n\right\} = 3.27027...$

Suy ra chiến thuật $p=\left(\frac{6}{37};\frac{20}{37};0;\frac{11}{37}\right)$ và $q=\left(\frac{14}{37};\frac{4}{37};0;\frac{19}{37};0\right)$ tương ứng là chiến thuật tối ưu cho người thứ nhất và người thứ hai.

9.30. Dòng 1 chi phối các dòng 4, 5, 6, ..., cột 1 chi phối các cột 4, 5, 6, ... nên có thể loại bỏ các dòng 4, 5, 6, ... và các cột 4, 5, 6, ... :

$$\begin{pmatrix} 0 & -1 & 2 & 2 & 2 & \dots \\ 1 & 0 & -1 & 2 & 2 & \dots \\ -2 & 1 & 0 & -1 & 2 & \dots \\ -2 & -2 & 1 & 0 & -1 & \dots \\ -2 & -2 & -2 & 1 & 0 & \dots \\ \dots & & & & & \dots \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & -1 & 2 \\ 1 & 0 & -1 \\ -2 & 1 & 0 \end{pmatrix}.$$

Ma trận là đối xứng bù, giá trị trò chơi là $V = 0$ và chiến thuật tối ưu của hai người là như nhau. Giải hệ: $\begin{cases} p_2 - 2p_3 = 0 \\ -p_1 + p_3 = 0. \\ 2p_1 - p_2 = 0 \end{cases}$

Ta có: $p = q = \left(\frac{1}{4};\frac{1}{2};\frac{1}{4};0;0;...\right)$ là chiến thuật tối ưu cho hai người.

9.33. $B = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 16 \\ 0 & 0 & 0 & 0 \\ \dots \end{pmatrix}$ có thể thu gọn về $A = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 16 \end{pmatrix}$ và có ma trận nghịch đảo là :

$$A^{-1} = \begin{pmatrix} 0,5 & 0 & 0 & 0 \\ 0 & 0,25 & 0 & 0 \\ 0 & 0 & 0,125 & 0 \\ 0 & 0 & 0 & 0,0625 \end{pmatrix}$$

Giá trị trò chơi là : 1.06667. Chiến thuật tối ưu cho người thứ hai là :

(0.53333; 0.26667; 0.13333; 0.06667).

Chiến thuật tối ưu cho người thứ nhất là :

(0.53333; 0.26667; 0.13333; 0.06667; 0; 0; ...).

9.34. a) Chiến thuật Baye của người thứ nhất phản ứng lại chiến thuật $q = \left(\frac{1}{5}; \frac{1}{5}; \frac{1}{5}; \frac{2}{5}\right)$ của người thứ hai là $p = (p_1 ; p_2 ; p_3)$ sao cho p^TAq đạt giá trị lớn nhất.

$$p^TAq = (p_1 ; p_2 ; p_3).\begin{pmatrix} 0 & 7 & 2 & 4 \\ 1 & 4 & 8 & 2 \\ 9 & 3 & -1 & 6 \end{pmatrix}.\begin{pmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.4 \end{pmatrix}$$

$$= (p_2 + 9p_3 ; 7p_1 + 4p_2 + 3p_3 ; 2p_1 + 8p_2 - p_3 ; 4p_1 + 2p_2 + 6p_3).\begin{pmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.4 \end{pmatrix}$$

$$= (0.2p_2 + 1.8p_3 + 1.4p_1 + 0.8p_2 + 0.6p_3 + 0.4p_1 + 1.6p_2 - 0.2p_3 + 1.6p_1 + 0.8p_2 + 2.4p_3)$$

$$= (3.4p_1 + 3.4p_2 + 4.6p_3).$$

Vậy giá trị $v = 3.4p_1 + 3.4p_2 + 4.6p_3$ (1) trong đó $p_1 + p_2 + p_3 = 1$ (2) và $p_1 \geq 0$; $p_2 \geq 0$; $p_3 \geq 0$. Thay (2) vào (1) có $v = 3.4 + 1.2p_3$.

Vậy v lớn nhất khi $p_3 = 1$; $p_2 = 0$; $p_1 = 0$. Chiến thuật Baye của người thứ nhất đáp ứng lại chiến thuật q của người thứ hai là chiến thuật $p = (0; 0; 1)$.

b) Người thứ hai cần dùng chiến thuật Baye đáp ứng lại chiến thuật (0; 0; 1) của người thứ nhất, đó là chiến thuật nhằm p^TAq đạt giá trị nhỏ nhất :

$$p^T A q = (0;0;1).\begin{pmatrix}0 & 7 & 2 & 4\\ 1 & 4 & 8 & 2\\ 9 & 3 & -1 & 6\end{pmatrix}.\begin{pmatrix}q_1\\ q_2\\ q_3\\ q_4\end{pmatrix} = (9;3;-1;6).\begin{pmatrix}q_1\\ q_2\\ q_3\\ q_4\end{pmatrix} = (9q_1 + 3q_2 - q_3 + q_4).$$

Vậy cần chọn q sao cho $v = (9q_1 + 3q_2 - q_3 + q_4)$ (3) đạt giá trị nhỏ nhất đồng thời thoả mãn điều kiện : $q_1 + q_2 + q_3 + q_4 = 1$ (4) và $q_j \geq 0$ với mọi $j = 1, 2, 3, 4$.

Thay (4) vào (3) có $v = 10q_1 + 4q_2 + 2q_4 - 1 \geq -1$. Suy ra v đạt giá trị nhỏ nhất khi $q_1 = q_2 = q_4 = 0$ và $q_3 = 1$, khi đó người thứ hai không mất điểm mà còn được 1 điểm (do người thứ nhất nhận giá trị trả về là $v = -1$ điểm). Vậy chiến thuật đáp ứng lại của người thứ hai là $q = (0 ; 0 ; 1 ; 0)$.

9.35. Xét ma trận trò chơi $A_{3\times3} = \begin{pmatrix}0 & 5 & -2\\ -3 & 0 & 4\\ 6 & -4 & 0\end{pmatrix}$, kiểm tra thấy A không có điểm yên ngựa và cũng không thể rút gọn bằng luật chi phối.

Lượt 1

Bước 1 : Cộng các phần tử của ma trận với $x = 4$, được ma trận sau :

$$A' = \begin{pmatrix}4 & 9 & 2\\ 1 & 4 & 8\\ 10 & 0 & 4\end{pmatrix}.$$

Bước 2 :

	y_1	y_2	y_3	
x_1	4	9	2	1
x_2	1	4	8	1
x_3	10	0	4	1
	–1	–1	–1	0

Bước 3 : Chọn chốt tại dòng x_3 cột y_1 (chứa giá trị 10).

Bước 4 : Thay đổi lại các giá trị của bảng, có bảng mới như sau :

	y_1	y_2	y_3	
x_1	–0.4	9	0.4	0.6
x_2	–0.1	4	7.6	0.9
x_3	0.1	0	0.4	0.1
	0.1	–1	–0.6	0.1

Bước 5. Thay nhãn cột 1 là y_1 và dòng 3 là x_3 cho nhau :

	x_3	y_2	y_3	
x_1	–0.4	9	0.4	0.6
x_2	–0.1	4	7.6	0.9
y_1	0.1	0	0.4	0.1
	0.1	–1	–0.6	0.1

Bước 6 : Quay lại bước 3.

Lượt 2

Bước 3 : Chọn chốt là ô thuộc dòng 1 (nhãn x_1) và cột 2 (nhãn y_2).

Bước 4 : Thay đổi lại các giá trị của bảng, có bảng mới với giá trị gần đúng như sau:

	x_3	y_2	y_3	
x_1	–0.0444	0.1111	0.0444	0.0667
x_2	0.0778	–0.4444	7.4222	0.6333
y_1	0.1	0	0.4	0.1
	0.0556	0.1111	–0.5556	0.1667

Bước 5 : Thay nhãn cột 2 là y_2 và dòng 1 là x_1 cho nhau :

	x_3	x_1	y_3	
y_2	–0.0444	0.1111	0.0444	0.0667
x_2	0.0778	–0.4444	7.4222	0.6333
y_1	0.1	0	0.4	0.1
	0.0556	0.1111	–0.5556	0.1667

Bước 6 : Quay lại bước 3.

Lượt 3

Bước 3 : Chọn chốt là ô thuộc dòng 2 (nhãn là x_2) cột 3 (nhãn là y_3).

Bước 4 và 5.

	x_3	x_1	x_2	
y_2	–0.0449	0.1138	–0.0060	0.0629
y_3	0.0105	–0.0599	0.1347	0.0853
y_1	0.0958	0.0240	–0.0539	0.0659
	0.0614	0.0778	0.0749	0.2141

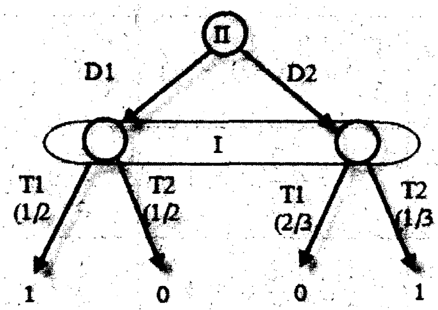
Bước 6, Bước 7: Giá trị trò chơi là 0,6713.

Chiến thuật tối ưu cho người thứ nhất là $p = (0.3636; 0.3497; 0.2867)$.

Chiến thuật tối ưu cho người thứ hai là $q = (0.3077; 0.2937; 0.3986)$.

9.36. Lưu ý rằng "Người thứ nhất không biết đồng tiền được dấu ở phòng nào" do đó hai vị trí mà người thứ nhất có thể chọn để di chuyển trong cây là cùng một tập thông tin.

Kí hiệu D_1 và D_2 là các chiến thuật nguyên thuỷ của người thứ hai, tương ứng là dấu đồng bạc ở phòng 1 hoặc dấu đồng bạc ở phòng 2.



Kí hiệu T_1 và T_2 là các chiến thuật nguyên thuỷ của người thứ nhất, tương ứng là tìm đồng bạc ở phòng 1 hoặc tìm ở phòng 2.

Theo đề bài có các phân bố xác suất như trên cây Kuhn.

Khi người thứ nhất thực hiện T_1, nếu người thứ hai thực hiện D_1 thì giá trị trả về là $\frac{1}{2}.1=\frac{1}{2}$.

Khi người thứ nhất thực hiện T_2, nếu người thứ hai thực hiện D_2 thì giá trị trả về là $\frac{1}{3}.1=\frac{1}{3}$.

Bảng giá trị trả về cho người thứ nhất là :

	D_1	D_2
T_1	$\frac{1}{2}$	0
T_2	0	$\frac{1}{3}$

Thực hiện chiến thuật cân bằng: Gọi tần suất thực hiện T_1 là p, thực hiện T_2 là $1-p$ thì có: $\frac{p}{2}=\frac{1-p}{3}$ suy ra $p=\frac{2}{5}$.

Vậy chiến thuật cân bằng tối ưu cho người thứ nhất là $\left(\frac{2}{5};\frac{3}{5}\right)$. Giá trị trò chơi là $\frac{1}{5}$. Tương tự tìm được chiến thuật tối ưu cho người thứ hai là $\left(\frac{2}{5};\frac{3}{5}\right)$.

9.37. Bảng tính giá trị trung bình trả về cho người thứ nhất :

	d-f	*d-g*	*e-f*	*e-g*
a	$\frac{1}{3}.3+\frac{2}{3}.(-3) = -1$	$\frac{1}{3}.3+\frac{2}{3}.(-3) = -1$	$\frac{1}{3}.3+\frac{2}{3}.0 = 1$	$\frac{1}{3}.3+\frac{2}{3}.3 = 3$
b	$\frac{1}{3}.3+\frac{2}{3}.0 = 1$	$\frac{1}{3}.0+\frac{2}{3}.0 = 0$	$\frac{1}{3}.3+\frac{2}{3}.0 = 1$	$\frac{1}{3}.0+\frac{2}{3}.3 = 2$
c	$\frac{1}{3}.3+\frac{2}{3}.0 = 1$	$\frac{1}{3}.3+\frac{2}{3}.0 = 1$	$\frac{1}{3}.(-3)+\frac{2}{3}.0 = -1$	$\frac{1}{3}.(-3)+\frac{2}{3}.3 = 1$

Vậy có ma trận trò chơi là:

$$\begin{array}{c} \\ a \\ b \\ c \end{array}\begin{array}{c} \begin{array}{cccc} df & dg & ef & eg \end{array} \\ \begin{pmatrix} -1 & -1 & 1 & 3 \\ 1 & 0 & 1 & 2 \\ 1 & 1 & -1 & 1 \end{pmatrix} \end{array}$$

Dòng (*a*) bị chi phối bởi dòng (*b*) nên có thể loại bỏ dòng (*a*). Cột (*eg*) bị chi phối nghiêm ngặt bởi cột (*ef*) nên có thể loại cột (*eg*). Cột (*df*) bị chi phối nghiêm ngặt bởi tổ hợp các cột (*dg*) và (*ef*) với tần suất 0.5 cho mỗi cột. Dẫn tới ma trận 2×2:

$$\begin{array}{c} \\ b \\ c \end{array}\begin{array}{c} \begin{array}{cc} dg & ef \end{array} \\ \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix} \end{array}$$

Dễ dàng tìm được giá trị trò chơi là $\frac{1}{3}$. Chiến thuật tối ưu cho người thứ nhất là $\left(0;\frac{2}{3};\frac{1}{3}\right)$. Chiến thuật tối ưu cho người thứ hai là $\left(0;\frac{2}{3};\frac{1}{3};0\right)$.

9.38. Với người thứ nhất : Kí hiệu *Đặt cược* là *đ*, *Kiểm tra* là *k*. Người thứ nhất có hai tập thông tin. Trong mỗi tập thông tin có thể tạo ra một lựa chọn trong hai lựa chọn. Do đó người thứ nhất có 2×2 chiến thuật nguyên thuỷ. Đó là các chiến thuật sau :

(*đ; đ*) : Đặt cược bất kể quân bài nhận được là quân bài thắng hay là quân bài thua;

(*đ; k*) : Đặt cược với quân bài thắng và kiểm tra với quân bài thua;

(*k; đ*) : Kiểm tra với quân bài thắng và đặt cược với quân bài thua;

(*k; k*) : Kiểm tra với quân bài thắng hoặc thua.

Vậy tập các chiến thuật nguyên thuỷ là $X = \{(đ;\ đ), (đ;\ k), (k;\ đ), (k;\ k)\}$. Chúng ta đã chứa trong X mọi chiến thuật nguyên thuỷ kể cả tốt lẫn xấu.

Người thứ hai chỉ có một tập thông tin. Do đó $Y = \{g;\ c\}$ với kí hiệu g là *Gọi*, c là *Chốt*.

Người thứ hai sử dụng g hoặc c nếu người thứ nhất vừa dùng đặt cược (*đ; đ*).

Ma trận giá trị trả về cho người thứ nhất như bảng sau:

	Cột 1: *g (gọi)*	Cột 2: *c (chốt)*
Dòng 1: *(đặt cược; đặt cược) = (đ;đ)*	$p(1 + b) + (1 - p)(-1 - b)$	$p.1 + (1 - p).1$
Dòng 2: *(đặt cược; kiểm tra) = (đ;k)*	$p.(1 + b) + (1 - p).(-1)$	$p.1 + (1 - p).(-1)$
Dòng 3: *(kiểm tra; đặt cược) = (k;đ)*	$p.1 + (1 - p).(-1 - b)$	$p.1 + (1 - p).1$
Dòng 4: *(kiểm tra; kiểm tra) = (k;k)*	$p.1 + (1 - p).(-1)$	$p.1 + (1 - p).(-1)$

Dòng 3 bị chi phối bởi dòng 1, nên có thể loại bỏ dòng 3. Dòng 4 bị chi phối bởi dòng 2 nên có thể loại bỏ dòng 2. Dẫn tới ma trận A kích thước 2×2 :

	Cột 1: *g (gọi)*	Cột 2: *c (chốt)*
Dòng 1: *(đặt cược; đặt cược) = (đ;đ)*	$p(1 + b) + (1 - p)(-1 - b)$	$p.1 + (1 - p).1$
Dòng 2: *(đặt cược; kiểm tra) = (đ;k)*	$p.(1 + b) + (1 - p).(-1)$	$p.1 + (1 - p).(-1)$

Khi $p = 1$

	Cột 1: *g (gọi)*	Cột 2: *c (chốt)*
Dòng 1: *(đặt cược; đặt cược) = (đ;đ)*	$1 + b$	p
Dòng 2: *(đặt cược; kiểm tra) = (đ;k)*	$1 + b$	p

A có hai dòng giống nhau, có hai điểm yên ngựa là $A(1;2)$ và $A(2;2)$.

Vậy giá trị trò chơi là p. Chiến thuật tối ưu cho người thứ nhất là $(x;\ 1-x;\ 0;\ 0)$ với $0 \le x \le 1$. Chiến thuật tối ưu cho người thứ hai là (0;1).

Khi $p = 0$

	Cột 1: *g (gọi)*	Cột 2: *c (chốt)*
Dòng 1: *(đặt cược; đặt cược) = (đ;đ)*	$-1-b$	1
Dòng 2: *(đặt cược; kiểm tra) = (đ;k)*	-1	-1

$A(2;1)$ là điểm yên ngựa. Giá trị trò chơi là -1.

Chiến thuật tối ưu cho người thứ nhất là (0; 1; 0; 0). Chiến thuật tối ưu cho người thứ hai là (1; 0).

Khi $\begin{cases} p \ne 0 \\ p \ne 1 \end{cases}$

Nhận thấy $A(1;\ 1)$ không là ô lớn nhất cột 1, $A(2;\ 1)$ không là ô nhỏ nhất dòng 2. $A(2;\ 2)$ không lớn nhất cột 2. Vậy nếu A có điểm yên ngựa thì chỉ có thể là $A(1;\ 2)$.

$A(1;\ 2)$ lớn nhất cột 2 do đó sẽ là điểm yên ngựa nếu nó nhỏ nhất dòng 1, nghĩa là:

$$A(1;\ 1) \ge A(1;\ 2) \Leftrightarrow p(1+b) + (1-p)(-1-b) \ge p.1 + (1-p).1 \Leftrightarrow p \ge \frac{2+b}{2(1+b)}$$

Vậy khi $p \ge \dfrac{2+b}{2(1+b)}$ (và $p \ne 1$) thì $A(1;\ 2)$ là điểm yên ngựa, do đó giá trị trò chơi là 1. Chiến thuật tối ưu cho người thứ nhất là (1; 0; 0; 0) và chiến thuật tối ưu cho người thứ hai là (0; 1).

Cuối cùng xét trường hợp $p < \dfrac{2+b}{2(1+b)}$ (và $p \ne 0$).

Bằng các công thức giải trò chơi ma trận 2×2 tìm chiến thuật cân bằng tối ưu sẽ có: Giá trị trò chơi là $V = \dfrac{(4p-1)(1+b)-1}{2+b}$.

Chiến thuật tối ưu cho người thứ nhất là:

$$\left(\frac{pb}{(1-p)(2+b)};\frac{2-2p+b-2pb}{(1-p)(2+b)};0;0\right).$$

Chiến thuật tối ưu cho người thứ hai là: $\left(\frac{2}{2+b};\frac{b}{2+b}\right)$.

9.39. a) Sử dụng các công thức tính giá trị và các chiến thuật tối ưu của trò chơi ma trận $\begin{pmatrix} a & b \\ d & c \end{pmatrix}$ không có điểm yên ngựa:

$$p=\frac{c-d}{a+c-b-d};\qquad p=\frac{c-b}{a+c-b-d};\qquad V=\frac{ac-bd}{a+c-b-d}$$

Ta có kết quả sau:

Trò chơi G_1 có điểm yên ngựa tại ô (1; 2) nên là giá trị trò chơi là 3, chiến thuật cho người thứ nhất là (1; 0) chiến thuật tối ưu cho người thứ hai là (0; 1).

Trò chơi G_2 không có điểm yên ngựa, giá trị trò chơi là 3, chiến thuật cho người thứ nhất là $\left(\frac{2}{5};\frac{3}{5}\right)$ chiến thuật tối ưu cho người thứ hai là $\left(\frac{1}{2};\frac{1}{2}\right)$.

Trò chơi G_3 không có điểm yên ngựa, giá trị trò chơi là -1, chiến thuật cho người thứ nhất là $\left(\frac{1}{2};\frac{1}{2}\right)$ chiến thuật tối ưu cho người thứ hai là $\left(\frac{1}{2};\frac{1}{2}\right)$.

Do đó trò chơi G xem như trò chơi với ma trận $G'=\begin{pmatrix} 0 & 3 \\ 3 & -1 \end{pmatrix}$. Trò chơi G' không có điểm yên ngựa, giá trị trò chơi là $\frac{9}{7}$, chiến thuật cho người thứ nhất là $\left(\frac{4}{7};\frac{3}{7}\right)$ chiến thuật tối ưu cho người thứ hai là $\left(\frac{4}{7};\frac{3}{7}\right)$.

b) Đổi cột 1 và cột 4 cho nhau, được ma trận: $G'=\begin{pmatrix} 1 & 6 & 0 & 0 \\ 5 & 3 & 0 & 0 \\ 0 & 0 & 2 & 5 \\ 0 & 0 & 4 & 1 \end{pmatrix}$.

Trò chơi trên G' coi như trò chơi $G'' = \begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix}$ mà

$$G_1 = \begin{pmatrix} 1 & 6 \\ 5 & 3 \end{pmatrix} \text{ và } G_2 = \begin{pmatrix} 2 & 5 \\ 4 & 1 \end{pmatrix}.$$

Giải G_1 và G_2 được kết quả:

	Giá trị trò chơi	Chiến thuật tối ưu cho người thứ nhất	Chiến thuật tối ưu cho người thứ hai
G_1	$V_1 = \frac{27}{7}$	$p_1 = \left(\frac{2}{7};\frac{5}{7}\right)$	$q_1 = \left(\frac{3}{7};\frac{2}{7}\right)$
G_2	$V_2 = 3$	$p_2 = \left(\frac{1}{2};\frac{1}{2}\right)$	$q_2 = \left(\frac{2}{3};\frac{1}{3}\right)$

Vậy G'' coi như trò chơi G^* sau đây:

$$G^* = \begin{pmatrix} \frac{27}{7} & 0 \\ 0 & 3 \end{pmatrix}.$$

Giải trò chơi G^* được kết quả là: $V = \frac{27}{16}$; chiến thuật tối ưu cho người thứ nhất là $p^* = \left(\frac{7}{16};\frac{9}{16}\right)$ chiến thuật tối ưu cho người thứ hai là $q^* = \left(\frac{7}{16};\frac{9}{16}\right)$.

Suy ra trò chơi G ban đầu có giá trị là $V = \frac{27}{16}$, chiến thuật tối ưu cho người thứ nhất là:

$$p = \left(\frac{7}{16}\times\frac{2}{7};\frac{7}{16}\times\frac{5}{7};\frac{9}{16}\times\frac{1}{2};\frac{9}{16}\times\frac{1}{2}\right) = \left(\frac{1}{8};\frac{5}{16};\frac{9}{32};\frac{9}{32}\right),$$

chiến thuật tối ưu cho người thứ hai là

$$q = \left(\frac{7}{16}\times\frac{3}{7};\frac{7}{16}\times\frac{4}{7};\frac{9}{16}\times\frac{2}{3};\frac{9}{16}\times\frac{1}{3}\right) = \left(\frac{3}{16};\frac{1}{4};\frac{3}{8};\frac{3}{16}\right).$$

9.40. Nếu trò chơi kết thúc với giá trị v thì : $v = Val\begin{pmatrix} v & 2 \\ 0 & 1 \end{pmatrix}$. Nếu trò chơi vô hạn ta gọi giới hạn của giá trị trả về là Q. Nếu $0 \leq Q < 2$ thì ô (1; 1) là điểm yên ngựa, giá trị trò chơi là Q. Nếu $Q > 2$ thì ô (1; 2) là điểm yên ngựa, giá trị trò chơi là 2. Nếu $Q < 0$ thì ô (2; 1) là điểm yên ngựa, giá trị trả về là 0.

Nếu trò chơi kết thúc với giá trị v, giải đẳng thức $v = Val\begin{pmatrix} v & 2 \\ 0 & 1 \end{pmatrix}$ sẽ có:

$v = \dfrac{v}{v-1}$ suy ra $v = 0$ hoặc $v = 2$. Khi đó chiến thuật tối ưu cho người thứ nhất là $\left(\dfrac{1}{v-1}; \dfrac{v-2}{v-1}\right)$. Chỉ có thể chọn $v = 2$ nghĩa là có chiến thuật (1; 0).

Với chiến thuật này người thứ hai luôn luôn chọn dòng 1, nhưng khi đó nếu người thứ hai luôn chọn cột 1 thì trò chơi không thể kết thúc (mâu thuẫn với giả sử).

Kết luận: Trong mọi trường hợp giá trị trò chơi là $v \in [0; 2]$.

9.41. Chúng ta hãy giải đồng thời các trò chơi này. Gọi giá trị của các trò chơi G_1, G_2, G_3 tương ứng là v_1, v_2 và v_3. Nếu không có trò chơi nào dẫn tới ma trận có điểm yên ngựa ta có:

$$v_2 = Val\begin{pmatrix} v_1 & 0 \\ 0 & 2 \end{pmatrix} \Rightarrow v_2 = \frac{2v_1}{2+v_1} \qquad \text{(a)}$$

$$v_3 = Val\begin{pmatrix} v_2 & 1 \\ 1 & 0 \end{pmatrix} \Rightarrow v_3 = \frac{1}{2-v_2} \Rightarrow v_3 = \frac{2+v_1}{4} \qquad \text{(b)}$$

$$v_1 = Val\begin{pmatrix} v_1 & v_2 & v_3 \\ v_2 & v_3 & v_1 \\ v_3 & v_1 & v_2 \end{pmatrix}.$$

Giả sử chiến thuật của người thứ nhất là $p = (p_1; p_2; p_3)$.

Sử dụng nguyên lí cân bằng có hệ :

$$\begin{cases} v_1 = p_1.v_1 + p_2.v_2 + p_3.v_3 \\ v_1 = p_1.v_2 + p_2.v_3 + p_3.v_1. \\ v_1 = p_1.v_3 + p_2.v_1 + p_3.v_2 \end{cases}$$

Cộng từng vế ba đẳng thức trên có :

$3v_1 = (v_1 + v_2 + v_3).(p_1 + p_2 + p_3) \Rightarrow 2v_1 = v_2 + v_3$ (c)

Từ hệ gồm (a), (b) và (c) suy ra $7v_1^2 + 4v_1 - 4 = 0 \Rightarrow v_1 = \dfrac{4\sqrt{2}-2}{7}$ (do $0 < v_1$).

9.42. Đây là trò chơi ngẫu nhiên một trạng thái (là G). Giả sử không có điểm yên ngựa khi trò chơi tiếp diễn.

Giá trị trò chơi là $v = Val\begin{pmatrix} 4 & 1+\dfrac{v}{3} \\ 0 & 1+\dfrac{2v}{3} \end{pmatrix} = \dfrac{4(1+\dfrac{2v}{3})}{4+(1+\dfrac{2v}{3})-(1+\dfrac{v}{3})} = \dfrac{12+8v}{12+v}$.

Dẫn tới phương trình : $v^2 + 4v - 12 = 0$ suy ra $v = 2$ (loại $v = -6$).

Thử lại $v = 2$, có ma trận $\begin{pmatrix} 4 & \dfrac{5}{3} \\ 0 & \dfrac{7}{3} \end{pmatrix}$, không có điểm yên ngựa. Từ đó suy ra:

Chiến thuật tối ưu cho người thứ nhất là (0.5; 0.5) vì: $p = \dfrac{\dfrac{7}{3}-0}{4+\dfrac{7}{3}-0-\dfrac{5}{3}} = \dfrac{1}{2}$.

Chiến thuật tối ưu cho người thứ hai là : $\left(\dfrac{1}{7};\dfrac{6}{7}\right)$ vì $q = \dfrac{\dfrac{7}{3}-\dfrac{5}{3}}{4+\dfrac{7}{3}-0-\dfrac{5}{3}} = \dfrac{1}{7}$.

9.43. a) $v_1 = Val\begin{pmatrix} 2 & 2+0.5v_2 \\ 0 & 4+0.5v_2 \end{pmatrix} = \dfrac{8+v_2}{4}$; $v_2 = \begin{pmatrix} -4 & 0 \\ -2+0.5v_1 & -4+0.5v_1 \end{pmatrix} = \dfrac{v_1-8}{3}$.

Từ đó dẫn tới hệ $\begin{cases} 4v_1 - v_2 = 8 \\ v_1 - 3v_2 = 8. \end{cases}$

Suy ra: $\begin{cases} v(1) = v_1 = \dfrac{16}{11} \\ v(2) = v_2 = -\dfrac{24}{11}. \end{cases}$

Thay các giá trị v_1 và v_2 vào các ma trận trên có

$$G^{(1)} = \begin{pmatrix} 2 & \dfrac{10}{11} \\ 0 & \dfrac{32}{11} \end{pmatrix}; \qquad G^{(2)} = \begin{pmatrix} -4 & 0 \\ -\dfrac{14}{11} & -\dfrac{36}{11} \end{pmatrix}.$$

Trong $G^{(1)}$ chiến thuật tối ưu của người thứ nhất là $\left(\dfrac{8}{11};\dfrac{3}{11}\right)$, của người thứ hai là $\left(\dfrac{1}{2};\dfrac{1}{2}\right)$.

Trong $G^{(2)}$ chiến thuật tối ưu của người thứ nhất là $\left(\dfrac{1}{3};\dfrac{2}{3}\right)$, của người thứ hai là $\left(\dfrac{6}{11};\dfrac{5}{11}\right)$.

b) Chương trình

```
uses crt;
const fo = 'shapley.dat';
type mang = array[1..2,1..2] of real;
var v1, v2 : real;
 g1, g2 : mang;
 p, q : real;
 k,i : integer;
 f : text;
procedure tinhv;
var mauso : real;
begin
 v1 := g1[1,1]*g1[2,2]-g1[2,1]*g1[1,2];
 mauso := g1[1,1]+g1[2,2]-g1[2,1]-g1[1,2];
```

```
 v1 := v1/mauso;
 p := g1[2,2]-g1[2,1];
 p := p/mauso;
 q := g1[2,2]-g1[1,2];
 q := q/mauso;
 write(f,'G1 co v1= : ', v1:5:2);
 write(f,' p=(',p:4:2,',',(1-p):4:2,')');
 writeln(f,' q=(',q:4:2,',',(1-q):4:2,')');
 v2 := g2[1,1]*g2[2,2]-g2[2,1]*g2[1,2];
 mauso := g2[1,1]+g2[2,2]-g2[2,1]-g2[1,2];
 v2 := v2/mauso;
 p := g2[2,2]-g2[2,1];
 p := p/mauso;
 q := g2[2,2]-g2[1,2];
 q := q/mauso;
 write(f,'G2 co v2= : ', v2:5:2);
 write(f,' p=(',p:4:2,',',(1-p):4:2,')');
 writeln(f,' q=(',q:4:2,',',(1-q):4:2,')');
end;
BEGIN
 clrscr;
 assign(f, fo);
 rewrite(f);
 write('Cac giai doan tu 0 den k. Nhap so k : ');
readln(k);
 v1 := 0;
 v2 := 0;
 for i:=1 to k do
 begin
 g1[1,1] := 2; g1[1,2] := 2+0.5*v2;
 g1[2,1] := 0; g1[2,2] := 4+0.5*v2;
 g2[1,1] := -4; g2[1,2] := 0;
 g2[2,1] := -2+0.5*v1; g2[2,2] := -4+0.5*v1;
 writeln(f,'Giai doan ',i, ' : ');
 tinhv;
 end;
 close(f);
END.
```

MỤC LỤC

Chịu trách nhiệm xuất bản :

Chủ tịch HĐQT kiêm Tổng Giám đốc NGÔ TRẦN ÁI

Phó Tổng Giám đốc kiêm Tổng biên tập NGUYỄN QUÝ THAO

Tổ chức bản thảo và chịu trách nhiệm nội dung:

Phó tổng biên tập PHAN XUÂN THÀNH

Giám đốc Công ty CP Dịch vụ xuất bản Giáo dục Hà Nội PHAN KẾ THÁI

Biên tập nội dung:

DƯƠNG VŨ KHÁNH THUẬN

Trình bày bìa:

LƯƠNG QUỐC HIỆP

Sửa bản in:

PHẠM THỊ THANH NAM

Chế bản:

PHÙNG MINH TRỤ

TÀI LIỆU CHUYÊN TIN HỌC - QUYỂN 3

Mã số: 8I769H1-CPD

In 3.000 bản, (QĐ số 11) khổ 17 x 24 cm tại Công ty In Thống kê & Sản xuất Bao bì Huế - 36 Phạm Hồng Thái Tp.Huế. Số đăng kí KHXB: 120-2011/CXB/22-83/GD. In xong và nộp lưu chiểu tháng 03 năm 2011.

Tìm đọc sách tham khảo về Tin học
của Nhà xuất bản Giáo dục Việt Nam

1. **Tài liệu chuyên Tin học - Quyển 1, 2, 3**	Hồ Sĩ Đàm (*Chủ biên*)
2. **Cấu trúc dữ liệu và giải thuật - Cẩm nang cho người lập trình**	Hồ Sĩ Đàm (*Chủ biên*), Bùi Thế Duy, Nguyễn Việt Hà
3. **Một số vấn đề chọn lọc trong môn Tin học** (*hai tập*)	Nguyễn Xuân My (*Chủ biên*)
4. **Em tập lập trình** (*hai tập*)	Trần Đỗ Hùng
5. **Sáng tạo trong thuật toán và lập trình**	Nguyễn Xuân Huy
6. **Bộ đề và đáp án các kì thi sát hạch chuẩn kĩ sư công nghệ thông tin**	Đức Minh
7. **Toán học rời rạc ứng dụng trong Tin học**	Kenneth H. Rosen, Phạm Văn Thiều, Đặng Hữu Thịnh *dịch*

Bạn đọc có thể mua sách tại các Công ty Sách - Thiết bị trường học ở địa phương hoặc các Cửa hàng sách của Nhà xuất bản Giáo dục Việt Nam:

- TP. Hà Nội : 187 Giảng Võ ; 232 Tây Sơn ; 23 Tràng Tiền ; 25 Hàn Thuyên ; 32E Kim Mã ; 14/3 Nguyễn Khánh Toàn.

- TP. Đà Nẵng : 78 Pasteur ; 247 Hải Phòng.

- TP. Hồ Chí Minh : 104 Mai Thị Lựu - 2A Đinh Tiên Hoàng, Quận 1 ; 240 Trần Bình Trọng - 231 Nguyễn Văn Cừ, Quận 5 ;

- TP. Cần Thơ : 5/5 Đường 30/4.

- Tại Website bán hàng trực tuyến www.sach24.com

Website : www.nxbgd.vn.

8 934994 074629

Giá: 30.000đ