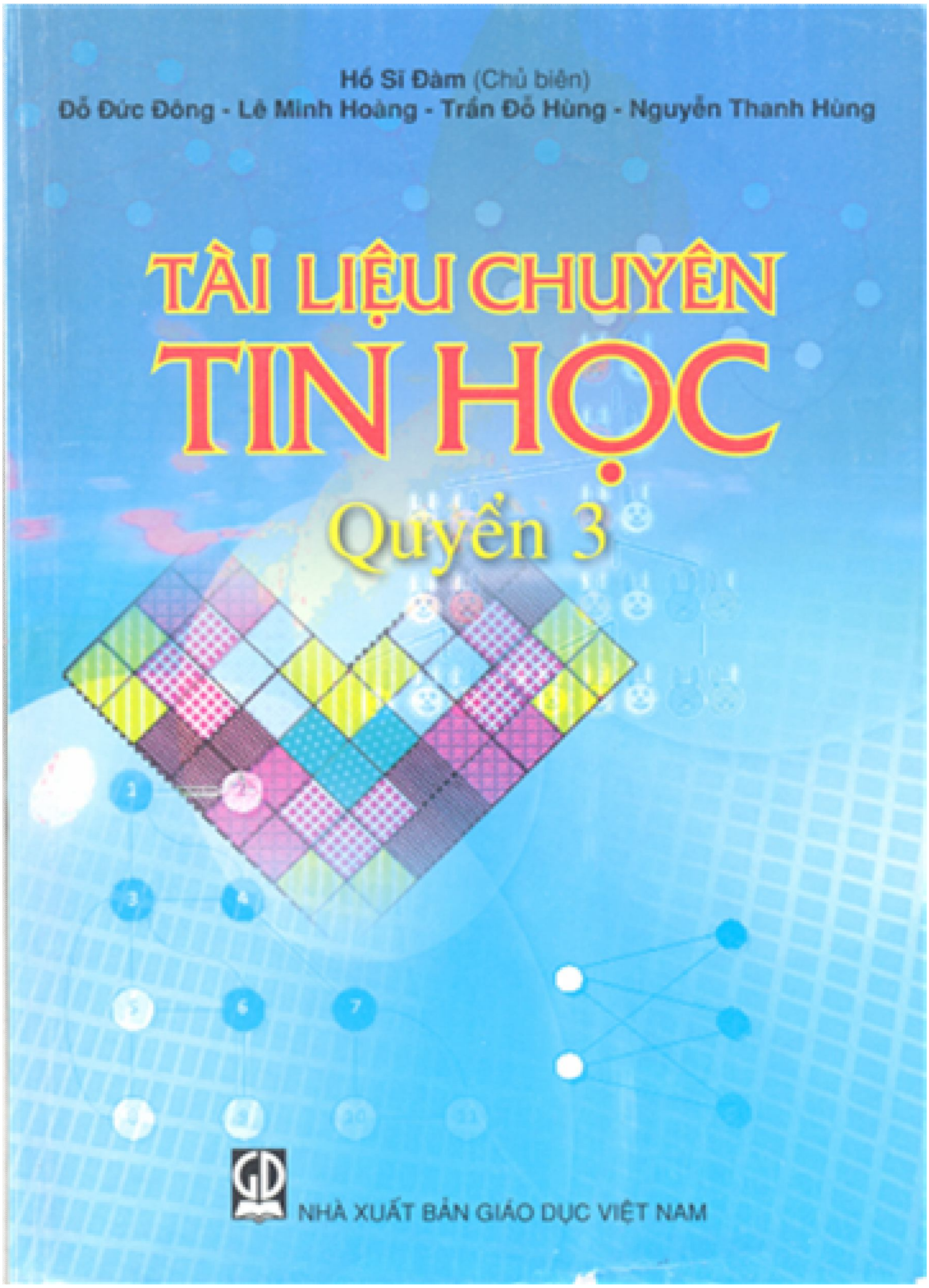
Ho Huu Son - Chuyen Nguyen Tat Thanh Kon Tum



MỤC LỤC

CHUYÊN ĐỀ 8. HÌNH HỌC TÍNH TOÁN

I. Một số khái niệm cơ bản 5
II. Một số bài toán cơ bản 18
III. Một số bài toán thông dụng khác 26

CHUYÊN ĐỀ 9. LÍ THUYẾT TRÒ CHƠI

I. Một số khái niệm 46
II. Trò chơi tổ hợp cân bằng 47
III. Trò chơi hai người có tổng điểm bằng 0 78

CHUYÊN ĐỀ 10. THUẬT TOÁN MÔ PHỎNG TỰ NHIÊN GIẢI BÀI TOÁN TỐI ƯU TỔ HỢP

I. Bài toán tối ưu tổ hợp 128
II. Thuật toán di truyền và tính toán tiến hoá 129
III. Phương pháp tối ưu hóa đàn kiến 134

HƯỚNG DẪN GIẢI BÀI TẬP 147

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.

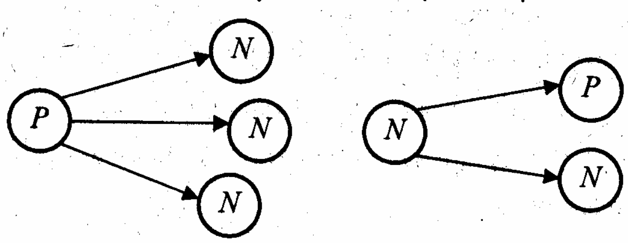
a) Tập P, tập N và cách tìm

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 quân 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 $\{(m', n')|\ m'$ hoặc n' 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 $(1 \leq s \leq m - 1)$, 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 $1 \leq s \leq n - 1$, 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, $1 \leq M \leq C$), 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 *Thừ 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:

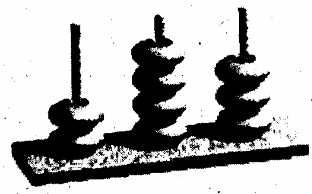
$$0 \oplus a = a,\ a \oplus a = 0, \text{ do đó nếu } a \oplus b = a \oplus c \text{ thì } b = c.$$

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 $F(x)$.

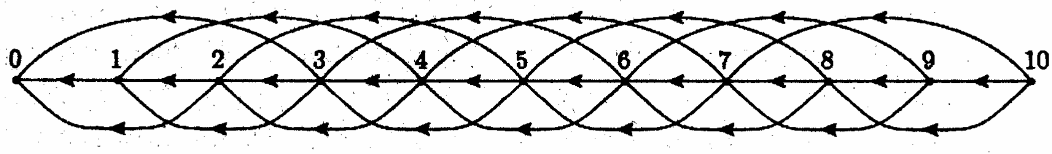
Trò chơi tổ hợp cân bằng có hai người chơi được mô phỏng trên đồ thị $G = (X, F)$ với đỉnh xuất phát $x_0 \in X$ và tuân theo các quy tắc sau:

(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:

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

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: $F(\mathrm{x}) = \{(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_n) : x_i' \in F(x_i)\}$.

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:* $g(x_1, x_2, \ldots, x_n) = g_1(x_1) \oplus g_2(x_2) \oplus \ldots \oplus g_n(x_n)$.

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