

Nim

Bergur Snorrason

March 11, 2024

- ▶ Skoðum einfaldan leik.
- ▶ Þú og einn andstæðingur eruð með hrúgu af  $n$  steinum.
- ▶ Þið skiptist á að taka einn eða tvo steina úr hrúgunni þar til hrúgan er tóm.
- ▶ Sá sem á að gera þegar hrúgan er tóm tapar.
- ▶ Hver vinnur?
  - ▶ Sá sem byrjar?
  - ▶ Sá sem byrjar ekki?
  - ▶ Fer eftir öðrum aðstæðum?

- ▶ Við samasöfum stöður í leiknum við fjölda steina í hrúgunni.
- ▶ Við segjum því að 0 sé *tapstaða*.
- ▶ Staða er sögð *vinningsstaða* ef það er leikur sem breytir henni í tapstöðu.
- ▶ Almennt er staða er sögð tapstaða ef það er enginn leikur sem breytir henni í tapstöðu.
- ▶ Munið að við getum annaðhvort fjarlægt einn eða tvo steina, svo stöðurnar 1 og 2 eru vinningsstöður.
- ▶ Almennt fæst að staða  $n$  er tapstaða þá og því aðeins að  $n$  gangi upp í þrjá.

- ▶ Almennt fæst að staða  $n$  er tapstaða þá og því aðeins að  $n$  gangi upp í þrjá.
- ▶ Þetta fæst með þrepun.
- ▶ Gerum ráð fyrir að  $3 \cdot k$  sé tapstaða.
- ▶ Þá eru stöður  $3 \cdot k + 1$  og  $3 \cdot k + 2$  vinningsstöður.
- ▶ Þetta eru líka einu stöðurnar sem maður kemst í frá stöðunni  $3 \cdot (k + 1)$ , svo hún er tapstaða.

- ▶ Tökum aðeins flóknara dæmi.
- ▶ Gerum ráð fyrir að við séum með  $m$  hrúgur með  $n_1, \dots, n_m$  steinum í hverri hrúgu.
- ▶ Nú skiptast keppendur á að velja hrúgu og taka eins marga steina úr henni og þeir vilja (að minnsta kosti einn).
- ▶ Aftur tapar sá sem getur ekki gert.
- ▶ Hvernig getum við lýst vinningsstöðunum og tapstöðunum?
- ▶ Þessi spurning er flóknari.
- ▶ Hér samsömum við  $m$ -dir við stöður.
- ▶ Hægt er að sýna að staðan  $(k_1, \dots, k_m)$  er tapstaða þá og því aðeins að  $k_1 \oplus \dots \oplus k_m = 0$ .

- ▶ Gerum ráð fyrir að við séum með leik þar sem:
  - ▶ Engar huldar upplýsingar.
  - ▶ Leikurinn klárast eftir endanlegan fjölda leikja.
  - ▶ Fyrir tiltekna stöðu geta báðir spilarar gert það sama.
  - ▶ Leikurinn klárast þegar það er enginn löglegur leikur og sá sem á að gera tapar.
- ▶ Þá getum við úthlutað hverjum leik heiltölu endurkvæmt.
- ▶ Ef engin löglegur leikur er í boði fær staðan heiltöluna 0.
- ▶ Annars fær staðan minnstu jákvæðu heiltöluna sem er ekki hægt að komst í úr stöðunni.
- ▶ Við höfum þá að staða er tapstaða þá og því aðeins að henni sé úthlutað töluna 0.
- ▶ Þessar tölur kallast Grundy tölur staðanna.

- ▶ Gerum ráð fyrir að við séum að spila marga leiki í einu, þar sem hver leikur er eins og lýst var að ofan.
- ▶ Hver aðgerð felur þá í sér að velja einn af leikjunum sem er verið að spila og framkvæm leik í honum.
- ▶ Sá tapar sem lendir í því að eiga engan löglegan leik.
- ▶ Ef við gerum ráð fyrir að undirleikirnir hafi Grundy tölurnar  $k_1, \dots, k_m$  þá er Grundy tala sameinaða leiksins  $k_1 \oplus \dots \oplus k_m$ .

- ▶ Tökum nú dæmi.
- ▶ Gerum ráð fyrir að við teiknum  $n$  kassa á blað í línu, nema það er búið að krotta yfir kassann í miðjunni.
- ▶ Hver leikur felur í sér að krotta yfir eins marga aðliggjandi kassa og manni lystir (að minnsta kosti einn).
- ▶ Sá sem getur ekki leikið tapar.



- ▶ Þegar við krotum yfir aðliggjandi reiti erum við í rauninni að skipta einum leik í tvo.
- ▶ Gerum ráð fyrir að það séu  $n$  reitir (númeraðir  $1, 2, \dots, n$ ) og að Grundy tala leiks fyrir  $k$  reiti sé  $g_k$ .
- ▶ Ef við krotum yfir reiti  $i, i + 1, \dots, j - 1, j$  skiptist leikurinn í tvennt, annar undirleikurinn með  $i - 1$  kassa og hinn með  $n - j$  kassa.
- ▶ Grundy talan fyrir þennan nýja yfirleik er  $g_{i-1} \oplus g_{n-j}$  ef  $i > 1$ , og  $g_{n-j}$  ef  $i = 1$ .

```

5 int mex(int* a, int n)
6 {
7     int b[n], i;
8     for (i = 0; i < n; i++) b[i] = 0;
9     for (i = 0; i < n; i++) if (a[i] < n) b[a[i]] = 1;
10    for (i = 0; i < n; i++) if (!b[i]) break;
11    return i;
12 }
13
14 int d[MAXN];
15 int dp_lookup(int x)
16 {
17     if (x == 0) return 0;
18     if (d[x] != -1) return d[x];
19     int i, j, a[x*x + 1];
20     for (a[0] = 0, i = 0; i < x; i++) for (j = i; j < x; j++)
21         a[i+a[0]] = dp_lookup(i)^dp_lookup(x - j - 1);
22     return d[x] = mex(a + 1, a[0]);
23 }

```

```

14 int d[MAXN];
15 int dp_lookup(int x)
16 {
17     if (x == 0) return 0;
18     if (d[x] != -1) return d[x];
19     int i, j, a[x*x + 1];
20     for (a[0] = 0, i = 0; i < x; i++) for (j = i; j < x; j++)
21         a[i+a[0]] = dp_lookup(i)^dp_lookup(x - j - 1);
22     return d[x] = mex(a + 1, a[0]);
23 }
24
25 void finna_besta_leik(int *a, int n)
26 {
27     int i, j, k, x, y;
28     for (i = 0; i < n; i++) for (j = i; j < n; j++)
29     {
30         for (k = i; k <= j; k++) if (!a[k]) break;
31         if (k <= j) continue;
32         for (k = i; k <= j; k++) a[k] = 0;
33         for (x = y = k = 0; k <= n; k++)
34         {
35             if (k == n || !a[k]) x ^= dp_lookup(y), y = 0;
36             else y++;
37         }
38         if (!x) return;
39         for (k = i; k <= j; k++) a[k] = 1;
40     }
41     for (k = 0; k < n; k++) if (a[k]) break;
42     a[k] = 0;
43 }

```

- ▶ Til að finna Grundy tölu á stöðu í einfalda leiknum þurfum við að reikna  $n$  gildi, og hvert gildi er reiknað í  $\mathcal{O}(n^2)$  tíma.
- ▶ Svo það tekur  $\mathcal{O}(n^3)$  tíma að reikna allar Grundy tölurnar.
- ▶ Til að finna best leikinn í hverri stöðu þrófum við alla mögulega leiki, sem tekur  $\mathcal{O}(n^3)$  tíma.
- ▶ Í heildina tekur þetta því  $\mathcal{O}(n^3)$  tíma.

