Nim e Outros Jogos Combinatórios

Tiago Reis

9 de agosto de 2017

Jogos Combinatórios

- Jogos de dois jogadores.
- Determinísticos: não dependem de eventos aleatórios, como o rolar de dados.
- Informação completa: todas as informações sobre o estado do jogo são públicas para todos os jogadores.
- Exemplos: xadrez, damas, go, nim...

Nim

Problema "Last Year at Marienbad", modificado. Existem várias fileiras com palitos, jogadores se alternam retirando um número positivo de palitos de uma delas.

Condição *normal* de vitória: o primeiro jogador a ficar sem jogadas, perde. O oposto é conhecido por condição *misère*.

Nim

Devemos encontrar as *configurações vencedoras*: estados em que o próximo jogador consegue garantir a vitória independentemente das jogadas do oponente.

Solução com programação dinâmica:

- A configuração com todas as fileiras vazias é perdedora.
- Se existe um movimento que leve o estado atual a uma configuração perdedora, ele está em uma configuração vencedora.
- Se todos os movimentos a partir do estado atual levam a uma configuração vencedora, ele está em uma configuração perdedora.

Nim - Solução com PD

```
int dp[2][4][6][8][10][12];

bool win(int a, int b, int c, int d, int e, int f) {
    if ((a|b|c|d|e|f) == 0) return true;
    if (dp[a][b][c][d][e][f] != -1)
        return dp[a][b][c][d][e][f];

    bool winning = false;
    for (int i=0; i<a; i++) winning |= !win(i,b,c,d,e,f);
    for (int i=0; i<b; i++) winning |= !win(a,i,c,d,e,f);
    for (int i=0; i<c; i++) winning |= !win(a,b,i,d,e,f);
    for (int i=0; i<d; i++) winning |= !win(a,b,c,i,e,f);
    for (int i=0; i<e; i++) winning |= !win(a,b,c,d,i,f);
    for (int i=0; i<f; i++) winning |= !win(a,b,c,d,e,i);

    return dp[a][b][c][d][e][f] = winning;
}
```

Nim - Solução Eficiente

Teorema

Uma configuração de nim normal em que os números de palitos nas pilhas é dado por x_1, x_2, ..., x_n é uma configuração vencedora se, e somente se, a soma nim $x_1 \oplus x_2 \oplus ... \oplus x_n$ é diferente de zero.

Para demonstrar esse teorema, basta mostrar a equivalência entre os movimentos permitidos no jogo e as operações sobre o grupo dos inteiros não-negativos dotados da operação de XOR bit a bit.

- A posição terminal do jogo é perdedora e tem soma nim igual a zero.
- De uma posição com soma nim igual a zero, todas as posições alcançáveis têm soma nim não nula.
- De uma posição com soma nim não nula, é possível alcançar uma posição com soma nim igual a zero.

Nim Misère

- Uma posição em que o número de fileiras com mais de um palito é exatamente 1 é vencedora, tanto no jogo normal quanto no misère.
- As duas versões podem ser jogadas da mesma forma até esse ponto.
- A partir daí, o jogo se reduz a filas de tamanho um, e a paridade do número dessas filas determina o vencedor do jogo.

Nim Misère

```
int main() {

    int T;
    scanf("%d", &T);

    for(int ncase=1; ncase<=T; ncase++) {
        int sz, x=0, notGreaterThanOne=1;
        for(int i=0; i<6; i++) {
            scanf("%d", &sz);
            x^=sz;
            if (sz > 1) notGreaterThanOne = 0;
        }
        printf("Instancia %d\n", ncase);
        printf("%s\n\n", (x^notGreaterThanOne)?"sim":"nao");
    }

    return 0;
}
```

Teorema de Sprague-Grundy

Teorema

Todo jogo imparcial finito sob a condição normal de vitória é equivalente a uma configuração de nim.

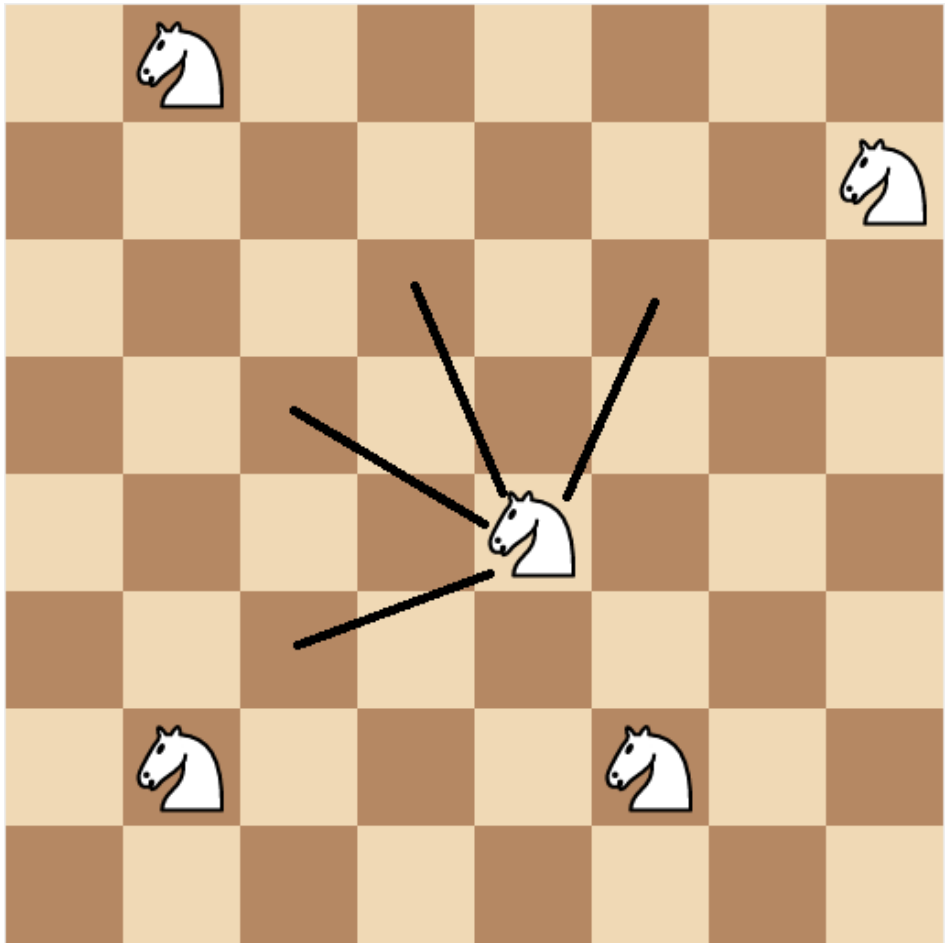
- Cada jogo é associado a um Número de Grundy ("Grundy number", "nim-value" ou "nimber").
- Número de Grundy: *mex* ("minimum excluded") do conjunto de Números de Grundy alcançáveis em uma jogada.
- Soma de subjogos independentes: XOR.

Takeaway

- O jogo começa com n fichas em uma pilha.
- Dois jogadores se alternam retirando 1, 2 ou 3 fichas.
- O último a jogar ganha.
- Quais são os Números de Grundy?

All the King's Horses

Dois jogadores se alternam escolhendo um cavalo e movendo-o de acordo com um dos 4 movimentos possíveis. Pode haver mais de um cavalo na mesma casa.



All the King's Horses

Podemos analisar cada cavalo individualmente. Os Números de Grundy para cada posição no tabuleiro são:

0	0	1	1	0	0	1	1
0	0	2	1	0	0	1	1
1	2	2	2	3	2	2	2
1	1	2	1	4	3	2	3
0	0	3	4	0	0	1	1
0	0	2	3	0	0	2	1
1	1	2	2	1	2	2	2
1	1	2	3	1	1	2	0

Variações: e se cada jogador pudesse escolher um subconjunto não vazio dos cavalos e fazer um movimento com cada um deles?

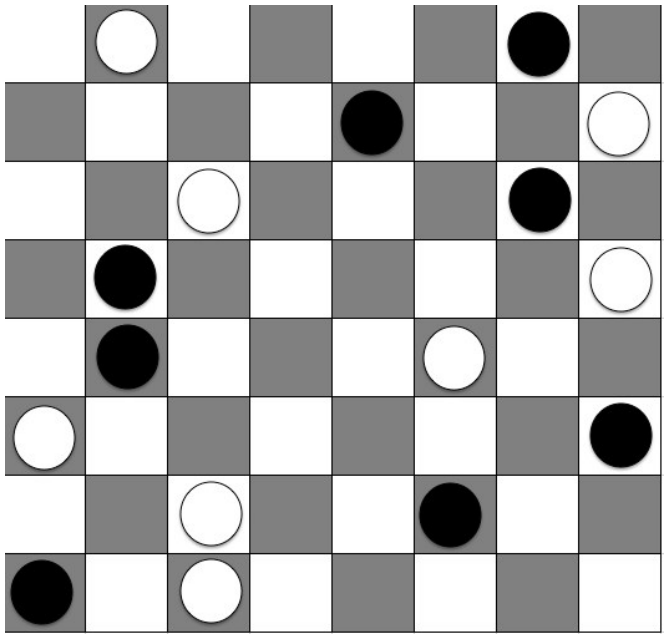
All the King's Horses

Outras formas de composição de jogos:

- União de jogos (ou): jogador escolhe um dos jogos e deve fazer um movimento nele.
- Composição seletiva de jogos (quase ou): o jogador deve escolher pelo menos um dos jogos para fazer um movimento, mas não pode escolher todos.
- Composição conjuntiva de jogos (e): o jogador deve fazer um movimento em cada jogo.
- Composição conjuntiva continuada de jogos (também): o jogador joga com todos os jogos que ainda têm um movimento válido.

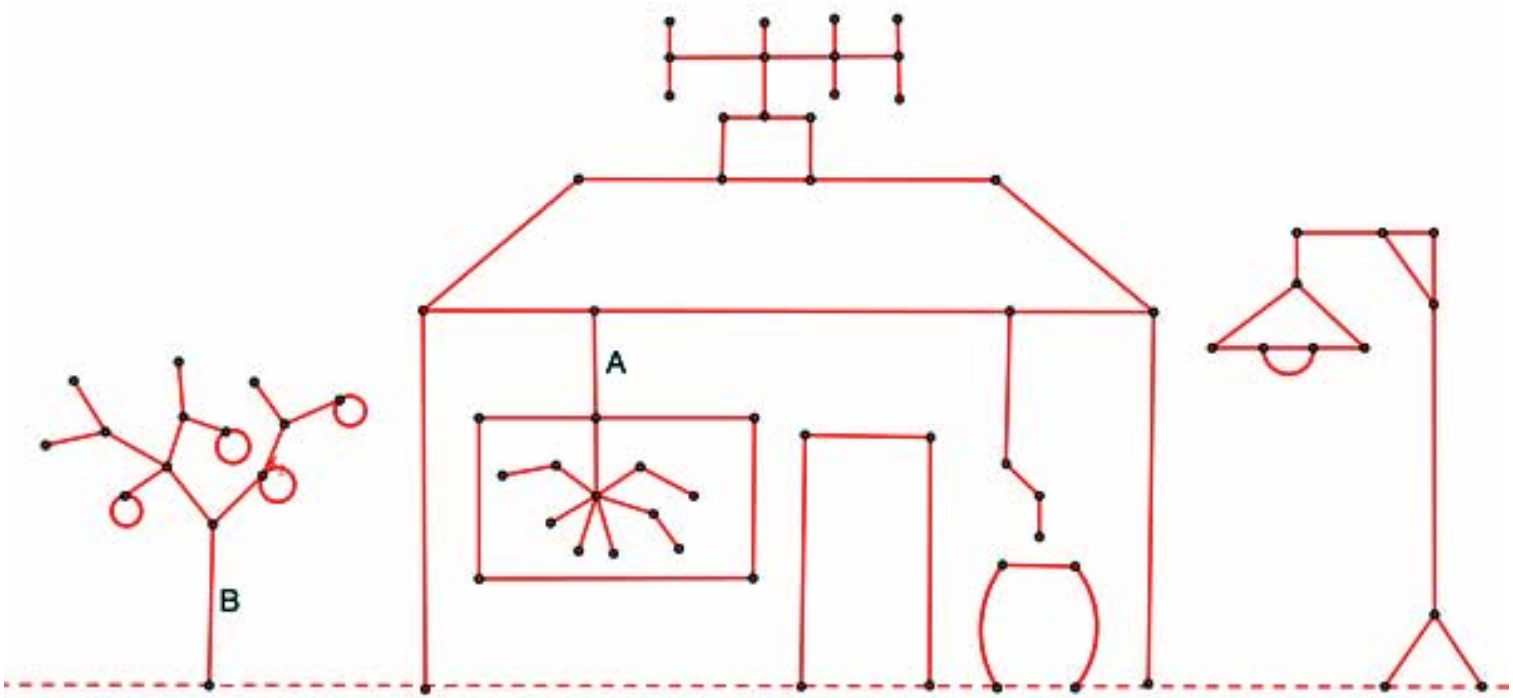
Northcott

- Cada jogador move peças de uma determinada cor.
- No seu turno, mova exatamente uma peça para qualquer outra posição na mesma linha, sem pular sobre uma peça do oponente.
- Quem não puder mover, perde.
- Não é um jogo imparcial nem necessariamente finito!



Outros jogos

Existem vários outros jogos combinatórios interessantes: kayles, rayles, rims, sprout, hackenbush...



Referências


E. R. Berlekamp, J. H. Conway e R. K. Guy. Winning Ways for your Mathematical Plays. Academic Press, New York.


T. S. Ferguson. Game Theory Class Notes. Disponível em: `http://www.cs.cmu.edu/afs/cs/academic/class/15859-f01/www/notes/comb.pdf`


TopCoder Tutorial: Algorithm Games. Disponível em: `https://www.topcoder.com/community/data-science/data-science-tutorials/algorithm-games/`