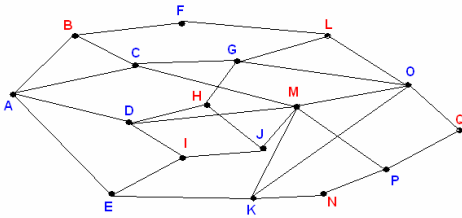


The Grundy theorem for combinatorial games
Matthieu Dufour, UQAM
Pace University, April 2007

1. Games as Graphs

Any combinatorial game can be viewed as a finite cycle-free oriented graph where two players alternately move from one vertex to another by following the edges in the direction of the arrows. When a player has to move from a terminal vertex (a vertex with no out degree) he loses the game.

The following graph represents a game, where the orientation of the edges is from left to right. The losing positions are labelled red on the graph and the winning ones are labelled blue.



Q is a terminal position of this game. The positions of the game and the vertices are interchangeable. A position Y is a successor of a position X if there is an edge going from X to Y . In our graph, the set of successors of D is $\{H, M, I\}$

2. Partitioning the game into the losing (L) positions and winning (W) positions

L and W stand for "losing" and "winning", from the point of view of the player who is about to play from this position.

The positions (vertices) of a game can be uniquely partitioned into two sets L and W with these properties:

- P1. From each position in W , there is a move that goes to a position in L ;
- P2. All the moves from a position in L lead to a position in W .

This partition is easily constructed by a recursive process:

R1: The terminal positions belongs to L

When all the successors of a position X are determined, then

R2: X is in L if all its successors are in W

R3: Otherwise X belongs to W .

The recursive construction process clearly shows that properties P1 and P2 uniquely determine the partition.

A player who is in a losing position is certain to lose, provided his opponent knows how to play, because each of his moves would put his opponent in a winning position and his opponent can always put him back in a losing position, and as the game must end, this player will eventually reach a terminal position where he will lose the game.

3. Combinatorial Sum of Games

If G_1, G_2, \dots, G_n are combinatorial games, then their **Sum** (denoted later as S) is a game where each player alternately plays by choosing one of the games G_i and playing a move in that particular game. As usual, the first player who is unable to play loses the game. This situation happens only when each of the games G_1, G_2, \dots, G_n , have reached a terminal position.

The vertices of S are all the possible vectors (v_1, v_2, \dots, v_n) , where v_i is a vertex of G_i . There is an edge from (v_1, v_2, \dots, v_n) to $(v'_1, v'_2, \dots, v'_n)$ if $v_i = v'_i$ for all i except one, (say v_k) and v'_k is a successor of v_k in the game G_k .

An example of a non-trivial sum of games is the game of **Nim** where piles of chips are initially put on a table and each player plays by alternately choosing a pile and then choosing to remove a positive number of chips from that pile. The first player who cannot remove any chip from any pile (because they are all gone!) loses.

Nim with one pile is trivial: the first player removes all the chips and his opponent loses immediately.

The game of **Nim** with n piles is simply the sum of n **Nim** games with one pile.

4. The Grundy Function

The "**Minimum Excluded Value**" function (MEX) maps each proper subset A of \mathbb{N} to the minimum integer belonging to A^c , i.e., to the smallest integer that is missing from the set A . For instance, if $A = \{0, 1, 2, 3, 5, 7, 8, 9, 10, 13, \dots\}$, then $\text{MEX}(A) = 4$.

The **Grundy Function** maps each position of the game to a non-negative integer in the following recursive way:

- G1. $\text{Gr}(X) = 0$ for every terminal position X ;
- G2. $\text{Gr}(X) = \text{MEX} \{ \text{Gr}(Y), \text{ for every successor } Y \text{ of } X \}$

Of course, this definition extends naturally to the digital sum of more than two integers. One also immediately sees that the digital sum is commutative, that is, for every x and y , one has $\dot{x} + \dot{y} = \dot{y} + \dot{x}$.

The following theorem may appear a bit technical but will prove very useful.

Theorem: If $\dot{x}_1 + \dot{x}_2 + \dot{x}_3 + \dots + \dot{x}_n = c > 0$ then, for every d such that $0 \leq d < c$, there is at least one $x'_i < x_i$ such that $\dot{x}_1 + \dots + \dot{x}_{i-1} + \dot{x}'_i + \dot{x}_{i+1} + \dots + \dot{x}_n = d$.

Let's look at an example to illustrate how the proof works:

digital sum							
powers of 2						base 10	
	32	16	8	4	2	1	
+	1	0	1	1	0	1	45
+	1	0	0	1	1	1	39
+	0	1	1	0	0	1	25
+	0	1	1	1	1	0	30
	0	0	1	1	0	1	13

Here, $c = 13$. Let's put $d = 6 < 13$, the "target" value for our digital sum and put it in our table :

digital sum							
Powers of 2						base 10	
	32	16	8	4	2	1	0
	1	0	1	1	0	1	45
	1	0	0	1	1	1	39
	0	1	1	0	0	1	25
	0	1	1	1	1	0	30
	0	0	1	1	0	1	13
	0	0	0	1	1	0	6 target

One takes the leftmost binary digit of d that is different from the corresponding digit of c . Here, it is in the "8" column and there is a "1" in the 13 row and a "0" in the 6 row. Observe that one will always have the "1" in the c row and the "0" in the d row because $d < c$.

As there is a "1" in the "8" column of the digital sum, there is at least one of the x_i for which there is a "1" in the "8" column (actually, there is a odd number of these x_i).

Here, we have three choices: 45, 25 and 30. Choose arbitrarily one of them, say 45. Change the "1" in the "8" column of 45 to a "0", and change, if necessary, the digits in the columns in the "45" row that are at the right of the "8" column in such a way that the resulting digital sum of the numbers equals the target 6.

Here is the result, where the gray shading shows where there has been a change:

digital sum							
powers of 2						base 10	
	32	16	8	4	2	1	
+	1	0	0	1	1	0	38
+	1	0	0	1	1	1	39
+	0	1	1	0	0	1	25
+	0	1	1	1	1	0	30
							6
	0	0	0	1	1	0	6
	0	0	0	1	1	0	6
							Target

As the left most modified cell will change a 1 to a 0, the resulting x_i' will always be smaller than the initial x_i , as was required in the theorem. The general proof of this theorem is absolutely similar to this illustration, only the notation would be more tedious.

6. The main theorem

Theorem (Grundy): Let S be the sum of the games G_1, G_2, \dots, G_n , and (v_1, v_2, \dots, v_n) a position of this game. Then $\text{Gr}((v_1, v_2, \dots, v_n)) = \text{Gr}(v_1) + \text{Gr}(v_2) + \dots + \text{Gr}(v_n)$, that is, the Grundy function of a position is the digital sum of the Grundy functions of all the positions in the games contributing to the sum S .

Proof: Lets Gr' be the function defined by the statement of the theorem. If we prove that Gr' has the recursive properties G1 and G2 of section 4, then we will be done.

G1. $\text{Gr}'((v_1, v_2, \dots, v_n)) = 0$ for every terminal position (v_1, v_2, \dots, v_n) :

If (v_1, v_2, \dots, v_n) is a terminal position, this means that each v_i is a terminal position of the game G_i . Therefore, $\text{Gr}(v_i) = 0$ for each i , and then

$$\text{Gr}'((v_1, v_2, \dots, v_n)) = \text{Gr}(v_1) + \text{Gr}(v_2) + \dots + \text{Gr}(v_n) = 0 + 0 + \dots + 0 = 0$$

G2. $\text{Gr}'((v_1, v_2, \dots, v_n)) = \text{MEX} \{ \text{Gr}'((w_1, w_2, \dots, w_n)) \}$, for every successor (w_1, w_2, \dots, w_n) of (v_1, v_2, \dots, v_n)

Let $c = \text{Gr}(v_1) + \text{Gr}(v_2) + \dots + \text{Gr}(v_n)$ and consider d such that $0 \leq d < c$. By the theorem of the previous section, there is a $x_i' < G(v_i)$ such that

$d = \text{Gr}(v_1) + \dots + \text{Gr}(v_{i-1}) + x_i' + \text{Gr}(v_{i+1}) + \dots + \text{Gr}(v_n)$. As $x_i' < G(v_i)$, there is in the game

G_i a successor w_i of v_i such that $x_i' = G(w_i)$. By letting $w_j = v_j$ for all $j \neq i$, one gets a

successor (w_1, w_2, \dots, w_n) of (v_1, v_2, \dots, v_n) such that $d = \overset{\cdot}{\text{Gr}}(w_1) + \overset{\cdot}{\text{Gr}}(w_2) + \dots + \overset{\cdot}{\text{Gr}}(w_n)$. So $\overset{\cdot}{\text{Gr}}'((v_1, v_2, \dots, v_n)) \geq \text{MEX} \{ \overset{\cdot}{\text{Gr}}'((w_1, w_2, \dots, w_n)) \}$, for every successor (w_1, w_2, \dots, w_n) of (v_1, v_2, \dots, v_n) .

One establishes equality by showing that if (w_1, w_2, \dots, w_n) is a successor of (v_1, v_2, \dots, v_n) , then $\overset{\cdot}{\text{Gr}}'(w_1, w_2, \dots, w_n) \neq \overset{\cdot}{\text{Gr}}'(v_1, v_2, \dots, v_n)$.

If (w_1, w_2, \dots, w_n) is a successor of (v_1, v_2, \dots, v_n) , then there is a $w_i \neq v_i$ and $w_j = v_j$ for every $j \neq i$, that is $\overset{\cdot}{\text{Gr}}(w_i) \neq \overset{\cdot}{\text{Gr}}(v_i)$ and $\overset{\cdot}{\text{Gr}}(w_j) = \overset{\cdot}{\text{Gr}}(v_j)$ for every $j \neq i$.

Then,

$$\overset{\cdot}{\text{Gr}}'((v_1, v_2, \dots, v_n)) = \overset{\cdot}{\text{Gr}}(v_1) + \overset{\cdot}{\text{Gr}}(v_2) + \dots + \overset{\cdot}{\text{Gr}}(v_n) \neq \overset{\cdot}{\text{Gr}}(w_1) + \overset{\cdot}{\text{Gr}}(w_2) + \dots + \overset{\cdot}{\text{Gr}}(w_n) = \overset{\cdot}{\text{Gr}}'((w_1, w_2, \dots, w_n))$$

And the proof is completed.

7. Application: the game of Nim

Let's say that (x_1, x_2, \dots, x_n) is a position of the game of Nim if the first heap has x_1 chips, the second one has x_2 chips, and so on.

Theorem: The position (x_1, x_2, \dots, x_n) of the game of Nim is losing if and only if

$$x_1 \oplus x_2 \oplus \dots \oplus x_n = 0.$$

Proof: We already know that the game of Nim is the sum of n Nim games with one pile.

It is easy to see that if there is a heap of size x in the one-pile-Nim, then $\overset{\cdot}{\text{Gr}}(x) = x$. The theorem then follows by the main theorem and the lemma.

Example: How should a player play in the Nim with 4 piles with heaps of sizes 39, 35, 25, and 14 chips? Let's see:

digital sum						
powers of 2						base 10
	32	16	8	4	2	1
+	1	0	0	1	1	1
+	1	0	0	0	1	1
+	0	1	1	0	0	1
+	0	0	1	1	1	0
						<u>14</u>

0	1	0	0	1	1	19	
0	0	0	0	0	0	0	target

digital sum

powers of 2						base 10	
32	16	8	4	2	1		
1	0	0	1	1	1	39	
1	0	0	0	1	1	35	
0	0	1	0	1	0	10	
0	0	1	1	1	0	14	
0	0	0	0	0	0	0	
0	0	0	0	0	0	0	target

The unique correct move is therefore to remove 15 chips of the third heap. Whatever move the opponent makes, the Grundy function of the resulting position will be greater than zero. Now we redo the process to figure out the next winning move.