2. Games of Strategy

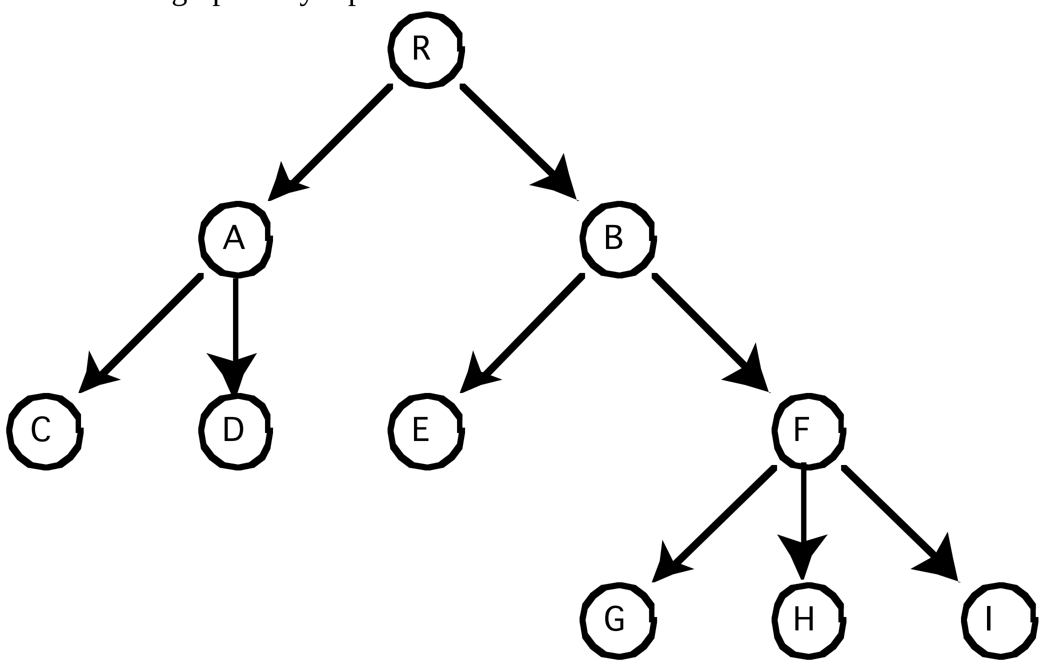
2.1. Game Trees

It should be clear from the preceding example that the mathematical formulation of games will involve trees. There are two possible ways to define trees. The first definition is based upon graphs:

2.1.1. Definition

A tree t with a root R is a directed graph so that for every node N of T different from R there is exactly one path leading from R to N

Trees can be graphically represented as follows:



There are two different ways to represent trees formally:

1. As an Incidence Matrix. Assume that the tree has n nodes. We label the rows and the columns of an $n \times n$ matrix with the nodes and write a 1 in the entry of the row labeled with X and the column labeled with Y if there is an arrow from X to Y. All other entries will be set to 0. For the preceding graph, we would obtain the following matrix A.

$$A = \left(\begin{array}{c|cccccccccc} & A & B & C & D & E & F & G & H & I & R \\ \hline A & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ C & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ G & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ H & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ R & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

We see that this is a very wasteful method.

2. Using sets and relation: Each edge e is represented as an ordered pair of nodes, where the first coordinate is the start point of the edge and the second coordinate is the end point of the edge. A tree then is represented by a pair $t = (T, \rho)$, where T is the set of nodes of t, and where ρ is a set of ordered pairs of nodes representing the edges of t. In our example,

$$T = \{A, B, C, D, E, F, G, H, I, R\}$$
$$\rho = \{(R,A),(R,B),(A,C),(A,D),(B,E),(F,G),(F,H),(F,I)\}$$

Widely used is also a third method of representing trees. It is based upon a recursive definition of trees:

2.1.2. Definition.

Trees are defined recursively as follows:

1. *If X is any variable, then* X *is a tree. The only node of* X*is X, which is also the root of* X *and its only leaf. This tree has no edges*
2. *If* $t_1, \ldots t_n$ *are n trees that do not have any nodes in common, and if R is a new variable, not occurring in any of n given trees, then* $t = [t_1 \oplus \cdots \oplus t_n]$ *is a tree.*
3. *The set of nodes of t is the set of nodes of the* t_i *'s together with the new node R.*
4. *R is the root of the new tree t.*
5. *The edges of t are all the edges in one of the trees* t_i *together with all edges connecting R to a root of one of the* t_i
6. *The set of leaves of t is the union of the leaves of the* t_i *'s*

If we use this definition of trees, then our example can be represented as

$$t = [C \oplus D] \oplus [E \oplus [G \oplus H \oplus I]]$$

This is the most economical way to present trees. It also can be used for programming purposes. However, it involves a lot of counting of square brackets.

We will use pictures whenever this is possible.

Next, we have to discuss game trees. If we have n players, we might think of them either as colors or as numbers. If we think of them as colors, then a game tree is simply a colored tree: We color the nodes with the colors representing the players. If we think of the players as numbers, then we label the nodes of the tree with the numbers representing the players.

Games also have outcomes. The outcome might be different for each player. What is desirable or relevant for one person might be not desirable relevant for the next. So each player will have a set of possible outcomes E_i. Later on, we will add an order to each E_i. For now, we define

2.1.3. Definition.

An n-player game G with outcomes E_1 ,…, E_n consists of

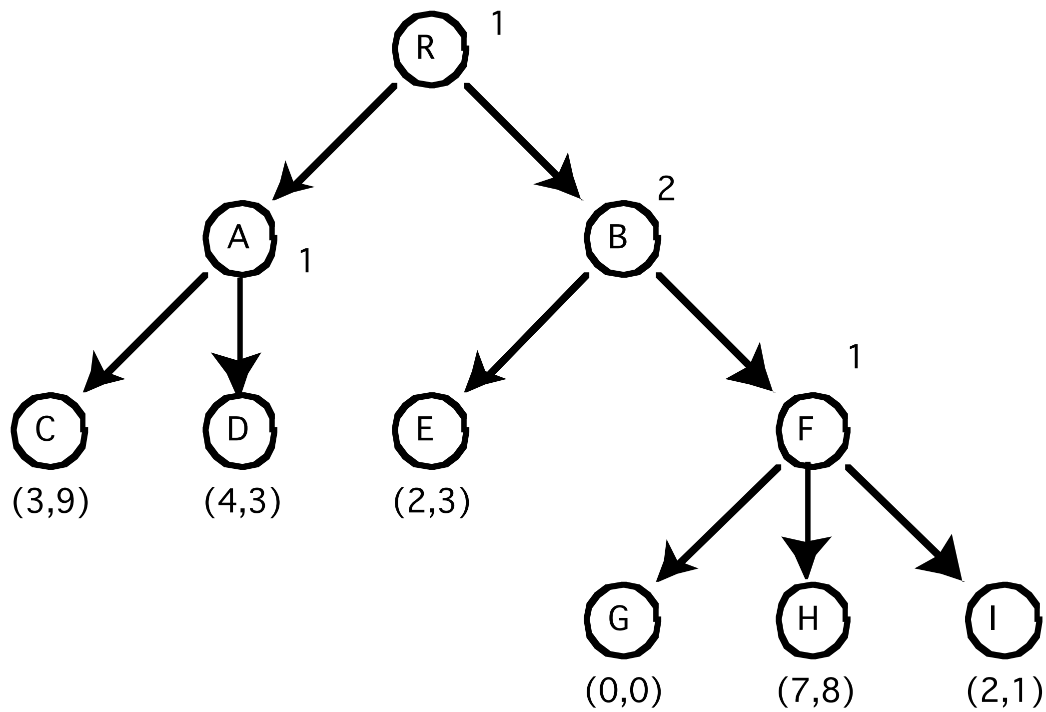
1. A tree t with nodes E and leaves L.

2. A function $p : E \setminus F \to \{1, \ldots, n\}$ that labels each node that is not also a leaf of the tree t with one of the numbers 1,…,n.

3. A function $\omega : L \to E_1 \times \cdots \times E_n$ that labels each leaf of the tree t with one possible combination of outcomes.

If node n is labeled with i, then we say that it is the ith *player's turn to move at node n.*

If we reuse our previous tree, then this tree can be turned into a two-player game in various ways. Here are three possibilities:



Here player 1 moves first. He might be tempted to go to the left, because this would exclude player 2 completely from the game. He draws again, and gains 4 units, while player 2 can only watch and gain 3 units. Player 1 wins.

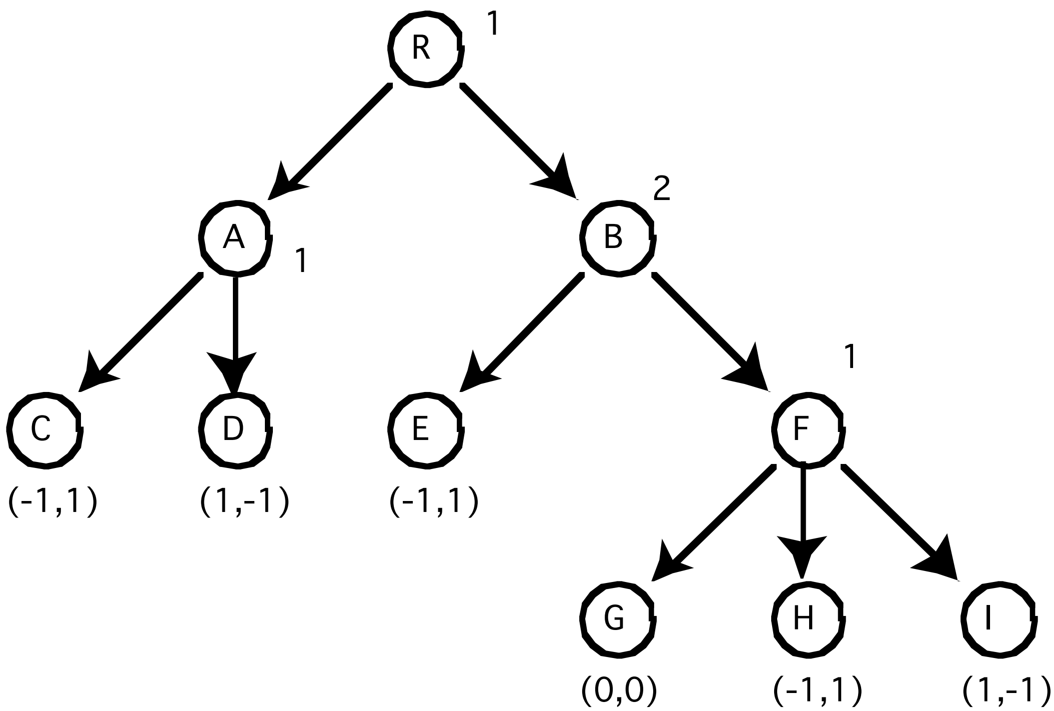
However, this might not be optimal for player 1. He could get 7 units, if he would trust player 2 move to the right. Now player 2 has the option. If she in turn would trust player 1, then she would also move to the right. Now player 1 gets to draw again. He now could really stick it to player 2 and move to the right. Then player 2 gets only 1, while player 1 laughs and wins. However, he has to pay a prize: He gets only two units. If we really would be nice and smart, he would move to the middle, gain 7 units – and loose to player 2!

So player 2 might opt not to trust player 1, move to the left when it is her turn and gain 3, while player 1 gets 2 units.

A lot of words with this small game. Mathematics cannot handle this kind of psychology. We better say precisely what our assumption and goals are and how our results are to be interpreted. There are of course dangers in that approach – see Friedrich Dürrenmatt's play "The Physicist." [1]

If we would use Zermelo's algorithm as discussed before for the game of Reversi, we would assume that both players behave rationally in some sense. The value of the game would be (7,8), player 1 would move to the right, player 2 would also move to the right and player 1 finally would move to the middle.

However we could modify this game by changing the possible outcomes. Both sets of outcome are {-1,0,1}, where -1 stands for loosing, 1 stands for winning and 0 stands for a draw. Then we would arrive at the following game tree:



Now the situation changes. Zermelo's algorithm dictates that the first player first moves to the left and then to the right, while the second player does not get to move at all.

The second game tree is also a typical example of a zero-sum game. Player 1 gains at much as player 2 looses. The sum of all outcomes is equal to 0. This is different for the first game tree.