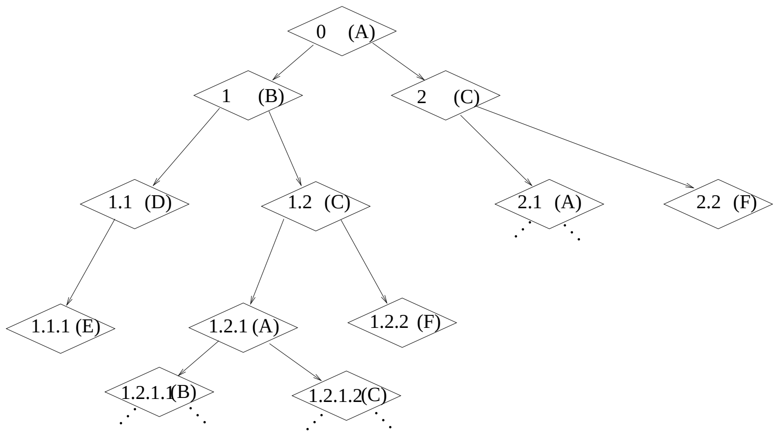
Computation, Information, and Intelligence (COMS/ENGRI/INFO/COGST 172), Fall 2005

9/7/05: **Lecture aid – DFS and BFS; games**

Agenda: More on DFS and BFS; games; perhaps also evaluation functions and minimax values.
Announcements:

- Prof. Lee will not be holding her regular office hours on Monday 9/12. (She will be out of town. There will still be lecture.)
- Specification #3 on the 9/2/05 handout has a repeated action type (the 10-11-12 schedule type appears twice, whereas it should only appear once).

That path tree again



Self-test questions: (understanding path-tree/specification relationships)
Q1: What must the initial state of the corresponding problem-space specification (CPS) be?
Q2: Which (single finite) path in the CPS must node 1.2.1 represent?
Q3: How many ways are there to reach state F from the initial state in the CPS?

Informal descriptions of our search algorithms: Note that the "least-Gorn-numbered" child of a given node n is the left-most one, since the Gorn numbers for the children of a given node differ only in the last "digit". (The children of a tree node are those that the node has edges to.) Thus, DFS can be *roughly* thought of as:

> **DFS:** By following edges, go deeper and to the left unless logic dictates otherwise, in which case, backtrack until you can explore new territory deeper and to the left again.

Also, note that the largest Gorn number of a set of nodes in a tree would be the one that is deepest (most "digits") and, in case of depth ties, the one that is most to the "right" (largest last "digit"). So, BFS can be *roughly* thought of as:

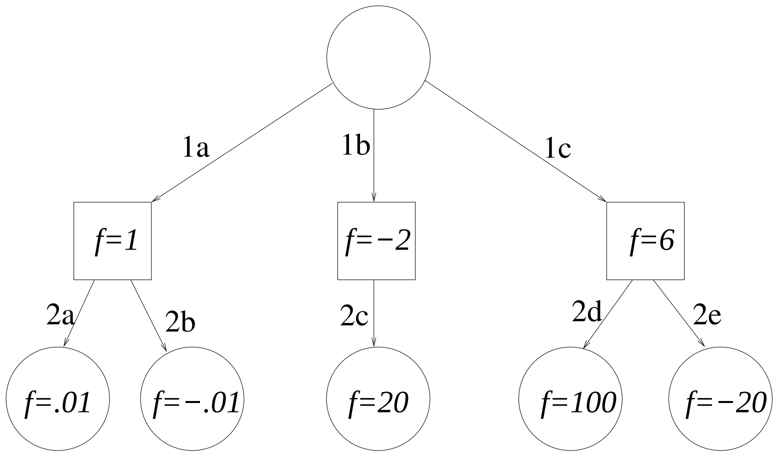
> **BFS:** Consider the tree to consist of "layers" of nodes depending on their distances to the root. Head rightwards through each layer unless logic dictates otherwise, in which case skip to the next available node to the right (wrapping around to the next layer if necessary).

These are *informal* descriptions: make sure you understand and follow the rules given in the previous class's lecture aid for removing and deleting nodes.

Two-player zero-sum games: assuming optimal (selfish) play, player one tries to maximize their benefit, which corresponds to minimizing the benefit to player two; conversely, player two aims to maximize the benefit to themselves, which corresponds to minimizing the benefit to player one.

For now, assume the existence of an *evaluation function* that tells us what the benefit to player 1 is of every possible state.

Example full game tree There is only one round in this very simple game. We've marked relevant nodes with the value of evaluation function f on the state that the node corresponds to. $\bigcirc$ and $\square$ indicate player 1's and player 2's turn to move, respectively. The edge labels indicate the name of the corresponding action.



Should player 1 make move 1c, since the game tree shows that it would yield a state with $f = 6$,[1] as opposed to the worse values 1 (node 1) and -2 (node 2)?

Minimax value of a (game tree) node: for now, think of this as the "true" eventual benefit to player 1 of taking the path that the node corresponds to, assuming player 1 always tries to force a *maximum* value of f for the situation that the game ends in, whereas player 2 tries to force a *minimum* value of f for the situation that the game ends in.

[1]Not necessarily a big deal, but for those keeping track of terminology at home: the game tree shows the fact mentioned above by depicting an edge labeled "1c" out of node 0 and into a node whose corresponding state has a value of 6 according to f. That is, strictly speaking, it is incorrect to speak of the node itself as having f-value 6, or of action 1c applying to node 0 (actions apply to the states of the problem-space specification). In short, the game tree depicts nodes, which are different from the states of the underlying problem-space specification.