Definition of a Game

(material mostly taken from [1]. Terms in **boldface** are being defined at that point.)

A Tree

A **graph** $G = (V, E)$ is a *countable* set of **vertices** V together with a *countable* set of edges E. Each **edge** is a pair of two vertices. If the graph is directed we consider every edge to be directed and $e = (v_1, v_2) \in V \times V$ is an ordered pair. If the graph is not directed then there is no ordering on the vertices that describe each edge.

A **path** is a set of edges of the form:

$$\{\{x_1, x_2\}, \{x_2, x_3\}, ..., \{x_{m-1}, x_m\}\} = \{\{x_k, x_{k+1}\} \mid k = 1, ..., m-1\}$$

where $m \geq 2$ and each x_k is a (not necessarily uniquely named) vertex in the graph. We allow edges from a vertex to itself and paths that include cycles, *e.g.* if $x_1 = x_4$ we have a cycle. We may say that such a path **connects** vertices x_1 and x_m. A **tree** is a graph in which each pair of vertices is connected by *exactly one* path of edges in the graph. A **rooted tree** is a tree in which one special vertex is designated as the **root** of the tree. When we speak of **the path to a given vertex**, we mean the unique path connecting this vertex to the root. **An alternative at a vertex** in a rooted tree is any edge that connects it with another vertex and is not in the path to this vertex. A vertex or edge x **follows** another vertex or edge y iff y is in the path to x. A vertex or edge x **immediately follows** another vertex or edge y iff x follows y and there is an alternative at y that *connects* x to y. A **terminal vertex** or **leaf** in a rooted tree is a vertex with no alternatives following it. A **non-terminal vertex** is a vertex that is not terminal. The set of all non-terminal vertices is denoted by V_{NT} the set of all terminal vertices by V_T .

A Game (in extensive form)

Definition: For any positive integer n, an **n-person extensive-form game** Γ^e is a 8-tuple $\Gamma^e = (T, \mathcal{P}, \mathcal{M}, \mathcal{I}, f, g, \theta, o)$ where:

- T is a rooted tree
- $\mathcal{P} = \{1, ..., n\}$ is the the set of players
- $\mathcal{M}$ is the set of move labels, *e.g.* $\mathcal{M} = \{up, down, left, right, shoot, bet, fold, raise\}$. Note that $\mathcal{M}$ includes all the possible labels for all players and for all the moves of the game.
- $\mathcal{I} = \{I_0, I_1, ...\}$ is the set of information states. ***All** the knowledge a player has when making a move is in which information state he is in at that vertex.*

- $f : V_{NT} \longmapsto \{0, 1, ..., n\}$ is a function that assigns to each non-terminal vertex in the rooted tree T a player i that should play at that vertex or the value 0. If f assigns zero to a vertex, the vertex is called a **chance vertex**. The set of all chance vertices is denoted by V_C. If f assigns a number in $\{1, ..., n\}$ to a vertex (*i.e.* the vertex is in V_{NT} and not in V_C) then the vertex is called a **decision vertex**. The set of all decision vertices is denoted by V_D.

- $g : (V_{NT}, V) \longmapsto \mathcal{M} \cup [0, 1]$ is a function that assigns to each alternative at a *decision vertex* a move label and to each alternative at a *chance vertex* a probability called the **chance probability**. The sum of the chance probabilities over all alternatives stemming from the same vertex is 1. When playing the game each player starting from the root of the tree chooses a move at the vertices f assigns to them. Thus, g must assign a *different* move label to each alternative of a decision vertex. We say that g *must be unambiguous.* The **path of play** is the path from the root to the current vertex that corresponds to the move labels chosen by each player and the chance events that occurred at each chance vertex along the way.

- $\theta : V_D \longmapsto \mathcal{I}$ is function that specifies at each decision vertex the *information state* that the player would have if the path of play reached this vertex. "When the path of play reaches a vertex controlled by a player, the player knows only the information state of the current vertex. That is, two vertices that belong to the same player should have the same information state *if and only if* the player would be unable to distinguish between the situations represented by these nodes when either occurs in the play of the game."[1] Thus, for any two vertices x and y that f assigns to the same player and θ assigns to the same information state, and for any alternative at vertex x , there must be exactly one alternative at node y that has the same move label. We say that θ *must be information compatible with* g (and f).

- $o : V_T \longmapsto \mathbb{R}^{\mathrm{n}}$ is a function that assigns to each terminal vertex a payoff vector which specifies the payoff to each player in some utility scale when this vertex is the outcome of the game. A terminal vertex is the outcome of the game when the path of play ends at that vertex.

Definition: "We say that a game has *perfect information* when no two vertices have the same information state.

That is, in a game with perfect information whenever a player moves, he knows the past moves of all other players and chance, as well as his own past moves."

Definition: We say that a player i has *perfect recall* if for any vertices x, y and z that are controlled by i, and for any alternative b at x, if y and z have the same information state and if y follows x and b, then there exists some vertex w and some alternative c at w such that:

- z follows w and c,
- $f(w) = i$, *i.e.* w is controlled by player i
- $\theta(w) = \theta(x)$, *i.e.* w has the same information state as x
- $g(c) = g(b)$, *i.e.* has the same move label as b

(Of course it may be that $w = x$ and $c = b$.)

This condition asserts that whenever a player moves, *he remembers all the information that he knew earlier in the game*, including all of his own past moves. However, he may not know his opponents' moves.

Definition: A pure strategy is a function $s : \mathcal{I}' \longmapsto \mathcal{M}$ that maps information states into moves. A pure strategy for a player i to play a game Γ assigns a move to each information state the player may face during the game. Furthermore, the moves assigned must be available moves for that state.

More formally, We define $\mathcal{I}'$ to be the subset of $\mathcal{I}$ that the player may face, *i.e.* $\mathcal{I}'$ is the smallest subset of $\mathcal{I}$ such that for all vertices v in V_D, if $f(v) = i$ then $\theta(v) \in \mathcal{I}'$. The requirement that the move must be available for that information state means that, for each v where $f(v) = i$ there exists an alternative e such that $s(\theta(v)) = g(e)$.

[1] Roger B. Myerson, *Game Theory: Analysis of conflict,* Harvard Press 1991