Library for extensive-form games

ABSTRACT

In the literature different algorithms have been developed to solve extensive-form games. However, these algorithms are not comparable because no test dataset provides a benchmark. In this document the reader can find the first classification of extensive-form games with perfect recall and perfect information, together with a dataset of games which covers a large set of possible combinations from the categories given. The attached files include a Python code to read and manipulate the games and a text file from which to upload the dataset.

State of the art. The two main softwares that allow to build extensive-form games are *Gambit* [1] and *Game Theory Explorer* (GTE) [2]. Both softwares are open source. In the last 30 years *Gambit* has been the most established software for studying game theory and now it comes also with a Python package, *pygambit*. On the other hand *GTE* is more accessible for the great public, as it is also available via web browser. These softwares have different features, that result to be cumbersome for a specific application to extensive-form games with perfect recall and perfect information. The code included in the library presents the same features of *pygambit*, but those that are not necessary for extensive-form games with perfect recall and perfect information.

Code. The code includes a Python class *Game* that allows to create an extensive-form game. As in *pygambit*, the attributes available at every node are:

- *player*, the player acting at the node;
- *parent*, the parent node;
- *children*, a dictionary having as keys the actions available at the node and as items the corresponding child nodes;
- *outcomes*, a list of the outcomes of the subgame;
- *depth*, the depth of the node in the tree;
- *utility*, a vector of the values of the utility of the node (if the node is an outcome).

Dataset. Every game has a reference code like $C2R3R-1532$, which shows the properties of the game. The reference code is to be interpreted by the following classification scheme:

- The first letter and the first number ($C2$ in the example) identify the *structure* of the game;
- The second letter and the number ($R3$) identify the *players* of the game;
- The third letter (R) shows the properties of the *utility* function;
- The last number (1532) is the *size*, i.e. the number of outcomes, of the game.

.

The coding for every category will be explained in the following paragraphs. Two datasets of games with two players are provided. Since most of the algorithms are made for two-player games, the dataset is limited to two-player games. The first dataset varies on the structure and the size, while the second dataset varies on the structure and the utility. We believe that the datasets include a significant range of options for every category. The library is available on GitHub.[1] Here follows the list of all the categories and respective codings.

Structure. The coding for the structure includes a letter and a number (n_S). The possible codings for the structure of the games are:

- *Random* (R), the number of actions available at every node are picked from the discrete uniform distribution $\mathcal{U}(1, n_S)$.
- *Complete* (C), the number of actions available at every node is equal to n_S. Every outcome has the same depth.
- *Totally Unbalanced* (B), the number of actions available at every node is equal to n_S. Out of n_S child nodes, $n_S - 1$ are outcomes.

Players. The number of the players is identified by the number of this category n_P. The letter attached to this category identifies the order with which such players are chosen:

- *Random* (R), the player acting at a node is chosen randomly among all players but the one acting at the parent node;
- *Ordered* (D), the players act one after another starting from the first.

Utility. The possible codings for this category are:

- *Random* (R), the utility of an outcome for a player is drawn by a uniform distribution $U(0, 1)$;
- *Discrete* (D), the utility of an outcome for a player is drawn by a discrete uniform distribution $\mathcal{U}(1, 10)$;
- *Zero-sum* (Z), the utility of an outcome for a player chosen randomly is 1, for the other players is 0;
- *Asymmetric* (A), the utility of an outcome for a player chosen randomly is $\mathcal{U}(0, 1)$, for the other players is 0;
- *Indifferent* (F), the utility of an outcome for every player has the same value and it is drawn by a uniform distribution $U(0, 1)$;
- *Equal* (E), the utility for every outcome for every player has the same value equal to 1.

[1] https://github.com/paolozapp/gtlibrary

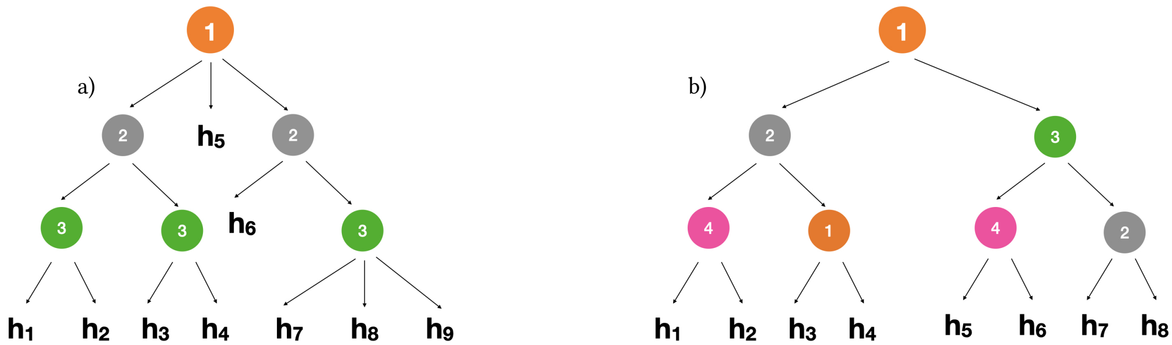

Figure 1: a) Game $R3D3R - 9$ **with random structure (**R**) and** $n_S = 3$**;**
b) Game $C2R4R - 8$ **with complete structure (**C**) and** $n_S = 2$**.**

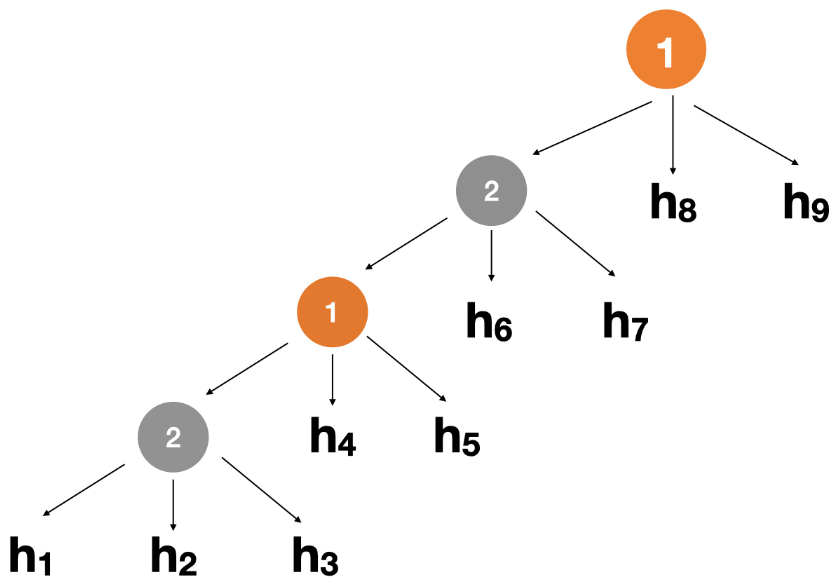

Figure 2: Game $B3D2R - 9$ **with totally unbalanced structure (**B**) and** $n_S = 3$**.**

Reference	Structure	n_S	Utility	Size
$R4D2R-100$	Random	4	Random	100
$C10D2R-100$	Complete	10	Random	100
$B4D2R-100$	Unbalanced	4	Random	100
$R4D2R-216$	Random	4	Random	216
$C6D2R-216$	Complete	6	Random	216
$B6D2R-216$	Unbalanced	6	Random	216
$R5D2R-324$	Random	5	Random	324
$C18D2R-324$	Complete	18	Random	324
$B18D2R-324$	Unbalanced	18	Random	324
$R5D2R-400$	Random	5	Random	400
$C20D2R-400$	Complete	20	Random	400
$B4D2R-400$	Unbalanced	4	Random	400
$R6D2R-512$	Random	6	Random	512
$C2D2R-512$	Complete	2	Random	512
$B8D2R-512$	Unbalanced	8	Random	512
$R6D2R-625$	Random	6	Random	625
$C5D2R-625$	Complete	5	Random	625
$B4D2R-625$	Unbalanced	4	Random	625
$R7D2R-729$	Random	7	Random	729
$C3D2R-729$	Complete	3	Random	729
$B14D2R-729$	Unbalanced	14	Random	729

Table 1: First dataset for games with 2 players.

Reference	Structure	n_S	Utility	Size
$R5D2R - 256$	Random	5	Random	256
$R5D2D - 256$	Random	5	Discrete	256
$R5D2Z - 256$	Random	5	Zero-sum	256
$R5D2A - 256$	Random	5	Asymmetric	256
$R5D2F - 256$	Random	5	Indifferent	256
$R5D2E - 256$	Random	5	Equal	256
$C2D2R - 256$	Complete	2	Random	256
$C2D2D - 256$	Complete	2	Discrete	256
$C2D2Z - 256$	Complete	2	Zero-sum	256
$C2D2A - 256$	Complete	2	Asymmetric	256
$C2D2F - 256$	Complete	2	Indifferent	256
$C2D2E - 256$	Complete	2	Equal	256
$B2D2R - 256$	Unbalanced	2	Random	256
$B2D2D - 256$	Unbalanced	2	Discrete	256
$B2D2Z - 256$	Unbalanced	2	Zero-sum	256
$B2D2A - 256$	Unbalanced	2	Asymmetric	256
$B2D2F - 256$	Unbalanced	2	Indifferent	256
$B2D2E - 256$	Unbalanced	2	Equal	256
$R3D2R - 729$	Random	3	Random	729
$R3D2D - 729$	Random	3	Discrete	729
$R3D2Z - 729$	Random	3	Zero-sum	729
$R3D2A - 729$	Random	3	Asymmetric	729
$R3D2F - 729$	Random	3	Indifferent	729
$R3D2E - 729$	Random	3	Equal	729
$C3D2R - 729$	Complete	3	Random	729
$C3D2D - 729$	Complete	3	Discrete	729
$C3D2Z - 729$	Complete	3	Zero-sum	729
$C3D2A - 729$	Complete	3	Asymmetric	729
$C3D2F - 729$	Complete	3	Indifferent	729
$C3D2E - 729$	Complete	3	Equal	729
$B5D2R - 729$	Unbalanced	5	Random	729
$B5D2D - 729$	Unbalanced	5	Discrete	729
$B5D2Z - 729$	Unbalanced	5	Zero-sum	729
$B5D2A - 729$	Unbalanced	5	Asymmetric	729
$B5D2F - 729$	Unbalanced	5	Indifferent	729
$B5D2E - 729$	Unbalanced	5	Equal	729

Table 2: Second dataset for games with 2 players (first part).

Reference	Structure	n_S	Utility	Size
$R4D2R - 1296$	Random	4	Random	1296
$R4D2D - 1296$	Random	4	Discrete	1296
$R4D2Z - 1296$	Random	4	Zero-sum	1296
$R4D2A - 1296$	Random	4	Asymmetric	1296
$R4D2F - 1296$	Random	4	Indifferent	1296
$R4D2E - 1296$	Random	4	Equal	1296
$C6D2R - 1296$	Complete	6	Random	1296
$C6D2D - 1296$	Complete	6	Discrete	1296
$C6D2Z - 1296$	Complete	6	Zero-sum	1296
$C6D2A - 1296$	Complete	6	Asymmetric	1296
$C6D2F - 1296$	Complete	6	Indifferent	1296
$C6D2E - 1296$	Complete	6	Equal	1296
$B6D2R - 1296$	Unbalanced	6	Random	1296
$B6D2D - 1296$	Unbalanced	6	Discrete	1296
$B6D2Z - 1296$	Unbalanced	6	Zero-sum	1296
$B6D2A - 1296$	Unbalanced	6	Asymmetric	1296
$B6D2F - 1296$	Unbalanced	6	Indifferent	1296
$B6D2E - 1296$	Unbalanced	6	Equal	1296
$R6D2R - 2401$	Random	6	Random	2401
$R6D2D - 2401$	Random	6	Discrete	2401
$R6D2Z - 2401$	Random	6	Zero-sum	2401
$R6D2A - 2401$	Random	6	Asymmetric	2401
$R6D2F - 2401$	Random	6	Indifferent	2401
$R6D2E - 2401$	Random	6	Equal	2401
$C7D2R - 2401$	Complete	7	Random	2401
$C7D2D - 2401$	Complete	7	Discrete	2401
$C7D2Z - 2401$	Complete	7	Zero-sum	2401
$C7D2A - 2401$	Complete	7	Asymmetric	2401
$C7D2F - 2401$	Complete	7	Indifferent	2401
$C7D2E - 2401$	Complete	7	Equal	2401
$B21D2R - 2401$	Unbalanced	21	Random	2401
$B21D2D - 2401$	Unbalanced	21	Discrete	2401
$B21D2Z - 2401$	Unbalanced	21	Zero-sum	2401
$B21D2A - 2401$	Unbalanced	21	Asymmetric	2401
$B21D2F - 2401$	Unbalanced	21	Indifferent	2401
$B21D2E - 2401$	Unbalanced	21	Equal	2401

Table 3: Second dataset for games with 2 players (second part).

REFERENCES

[1] Richard D McKelvey, Andrew M McLennan, and Theodore L Turocy. 2006. Gambit: Software tools for game theory. (2006).

[2] Rahul Savani and Bernhard Von Stengel. 2015. Game Theory Explorer: software for the applied game theorist. *Computational Management Science* 12, 1 (2015), 5–33.