Rewards Structure in Games: Learning a Compact Representation for Action Space

Margot Yann, Yves Lespérance, Aijun An
York University
lisayan@cse.yorku.ca

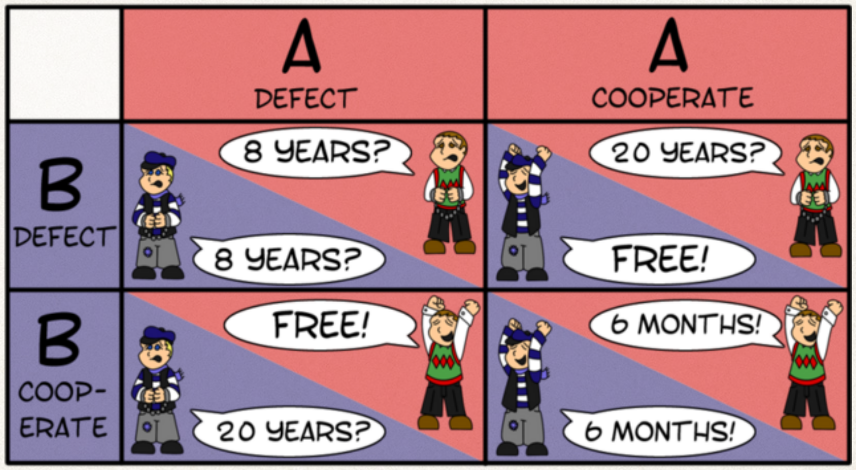
February 4, 2017

Introduction

- Computer Games: a key AI testbed

- Distinguish between games in:
"computer game" sense (e.g., Super Mario)

VS.



"game theoretical" sense (e.g., Prisoners Dilemma)

- Game Theory: how agents' strategies affect game outcomes/rewards

Exploring Action Space

In a game:

- While playing, players explore their individual action space combined with other players' action choices.

- Depending on the goal of each player, this action-choosing process is non-stationary and dynamic.

Motivation

- Common research problems exist in "computer games" and "game theoretic" sense, but there's a gap between them.
- Research problem: action space grows exponentially, when:
  - number of actions increases
  - number of players increases
- Many players may be irrelevant to a given player's payoff

Thus, a compact representation of payoff function is needed.

Our interest is to identify these irrelevant players through exploring players' payoff space, and to create a compact representation of player influence graph to eliminate irrelevant players from the search space of an individual's action choice.

Objectives

- Our approach comes from a machine learning perspective, and focuses on revealing the influence between all the action choices and the outcome utility;
- Directly learn structures of graphical games from payoff functions induced using regression models for normal-form games.

Why Graphical Games?

- Graphical Game Definition:

A graphical game is described as an undirected graph G in which players are represented as vertices, and each edge identifies influence between two vertices.

- In natural settings,
  - a player: represented as vertex **v**
  - payoffs: action of vertex **v** & neighbours of **v** who have influence over vertex **v**.

Each player's payoff is given by a matrix with all combinations of players' action choices using normal form representation.

Graphical games

Study game theoretic games: well defined & full information

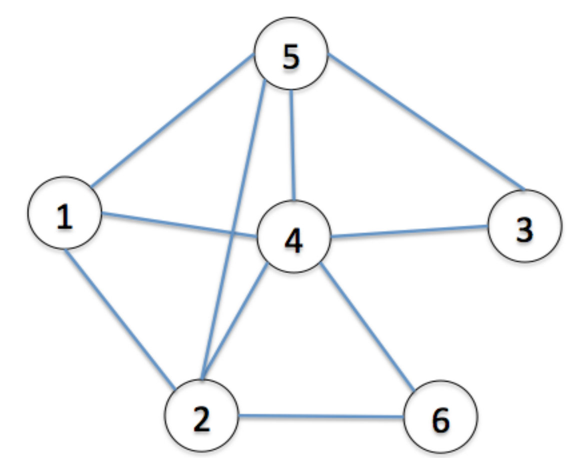
- We randomly generate multiplayer **Graphical Games** using GAMUT;
- Normal form representation:
  - Action profiles & the corresponding utilities for each player:

  a game with 6 players with 6 actions each has 46656 (6^6) action profiles.

  - Action combinations: also called a "joint strategy"
  - Graphical game structure: example of a 6-player game

```
# Players:     6
# Actions:     6 6 6 6 6 6
# players:     6
# actions:     [6]
# graph:       RandomGraph
# graph_params:[ -nodes 6 -edges 10 ]
# Graph Params:
# { nodes: 6, edges: 10, sym_edges: true, reflex_ok: false }
[1  1  1  1  1  1] :   [ 0.9889 0.1452 0.3467 0.0380 0.2359 0.8738 ]
[2  1  1  1  1  1] :   [ 0.6493 0.4155 0.3467 0.6591 0.4223 0.8738 ]
[3  1  1  1  1  1] :   [ 0.1519 0.1217 0.3467 0.0664 0.6214 0.8738 ]
[4  1  1  1  1  1] :   [ 0.2684 0.4025 0.3467 0.3414 0.7824 0.8738 ]
[5  1  1  1  1  1] :   [ 0.2793 0.9420 0.3467 0.1562 0.4247 0.8738 ]
[6  1  1  1  1  1] :   [ 0.1645 0.1241 0.3467 0.3051 0.4003 0.8738 ]
[1  2  1  1  1  1] :   [ 0.8218 0.0256 0.3467 0.1831 0.5240 0.7784 ]
[2  2  1  1  1  1] :   [ 0.9571 0.1945 0.3467 0.1309 0.5542 0.7784 ]
[3  2  1  1  1  1] :   [ 0.2123 0.1948 0.3467 0.3615 0.2782 0.7784 ]
[4  2  1  1  1  1] :   [ 0.3500 0.3485 0.3467 0.3791 0.2542 0.7784 ]
[5  2  1  1  1  1] :   [ 0.7256 0.0974 0.3467 0.7004 0.0661 0.7784 ]
[6  2  1  1  1  1] :   [ 0.0774 0.8020 0.3467 0.3862 0.7296 0.7784 ]
                       …
                       …
```

Figure 1: Data sample from a 6-player random graphical game



Objectives & Approach

Goal: learn an approximate player influence graph
(the influence between paired actions as a connection, an edge)

Multi-Descendent Regression Learning Structure Algorithm (MDRLSA):

- 1) use linear regression methods to learn a player's utility function;
- 2) use the payoff functions to identify independence among players and further generate a graphical game structure representation.

Contribution

MDRLSA successfully achieves the stated goal to learn an approximate player influence network, and

- performs better in terms of time and accuracy compared with a state-of-art graphical game model learning method;
- the running time of MDRLSA increases linearly with respect to the number of strategy profiles of a game.

MDRLSA Design

- Given a set of data points (x,y):

x describes an instance where players choose a pure strategy profile and realized value $y = (y_1, \cdots, y_{n_p})$

- For deterministic games of complete information, y is simply $f(x)$.
- We address payoff-function learning as a standard regression problem: selecting a function f to minimize some measure of deviation from the true payoff y.

δ-*independent*

- Definition "δ-*independent*":

Consider a game [I, (*x*), *y*(s)], player p and q are δ-*independent,* if for every x_p, $x_p \in X_p$, and for any available joint strategy of x_{-pq},

$$\max_{i \in [1, N_a]} y_p(x_p, x_q^i, x_{-pq}) - \min_{j \in [1, N_a]} y_p(x_p, x_q^j, x_{-pq}) \leq \delta$$

- We define an influence graph as a $n_p \times n_p$ binary matrix:

$$\text{graph\_param}(i, j) = \begin{cases} 0, & \text{if } i \text{ \& } j \text{ are } \delta\text{-independent;} \\ 1, & \text{otherwise.} \end{cases}$$

MDRLSA - Step 1

- Modelling: fit parameters θ to all players' utility profiles y
- **Action mapping:**

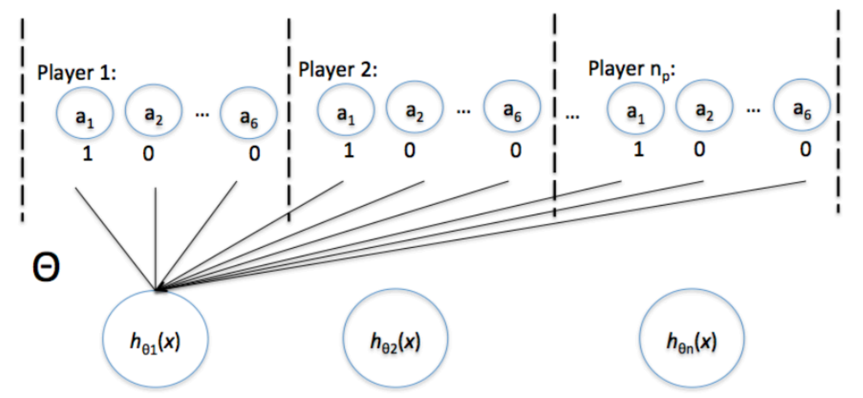


Figure 2: n_p player' action mapping

- $h_{\theta_k}(x)$: approximate of utility y_k, given as the Eq. 1 linear model:

$$\begin{aligned} h_{\boldsymbol{\theta_k}}(x) &= \boldsymbol{\theta}_k^\intercal \boldsymbol{x} + \varepsilon_k \\ &= \theta_{1k}x_1 + \cdots + \theta_{ik}x_j + \cdots + \theta_{n_a k}x_{n_p * n_a} + \varepsilon_k \end{aligned} \tag{1}$$

$x_j = \{0,1\}$ $\quad i \in [1, n_a], j \in [1, n_p * n_a], k \in [1, n_p]$.

MDRLSA - Step 2

- We define the cost function as,

$$J(\boldsymbol{\theta}_k) = \frac{1}{2m}\sum_{l=1}^{m}\left(h_{\boldsymbol{\theta}_k}(x^{(l)}) - y_k^{(l)}\right)^2. \quad (2)$$

- when the matrix XX is invertible, we have

$$\hat{\boldsymbol{\theta}}_k = (X^\intercal X)^{-1} X^\intercal \boldsymbol{y_k},$$

- Map Θ onto player action-influential relationships, based on the given utilities.

$$\Theta = [\theta_1 \ldots \theta_k \ldots \theta_{np}]$$

Parameter δ

- δ is set as a parameter to control the tolerance level for the influence among players.
- The larger we set the delta parameter, the coarser the approximation of the game; but the smaller the number of connections in the graphical game, resulting in larger computational gains.

Linearity Assumption

- Objective of our model: **identify independence**
- **Simplicity** of linear approximation: can be fitted efficiently.
- **Evaluate the validity** of our linearity assumption:

we use cost functions to measure how well the linear models correctly capture the functions.

Notes:

More complex relationships, which may not be perfectly modelled using linear functions, also **imply the players influence each other and are not independent.**

Thus, simple fitting of a linear model is used to identify the independence.

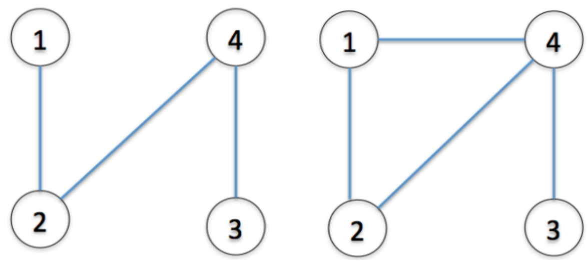
Empirical Results

We tested MDRSLA on a set of random graphical games generated from GAMUT:

Game	Player Number	Action Number	Edge Number	Runtime (Seconds)	Accuracy (%)	Normal Form Profile Entries
a.	4	3	3	0.0063	100	81
b.	4	4	4	0.0091	100	256
c.	5	3	5	0.0136	100	243
d.	5	4	6	0.0156	100	1024
e.	5	5	7	0.0193	100	3125
f.	6	4	5	0.0340	100	4096
g.	6	5	8	0.1029	100	15625
h.	6	6	10	0.2999	100	46656

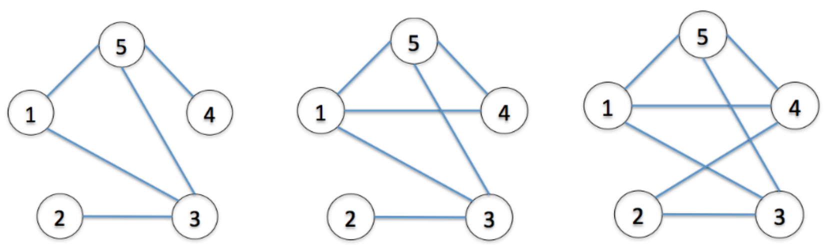
Table 1: MDRLSA performance on random graphical games experiments

Experiment Results-1



(a) 4 players, 3 actions, 3 edges

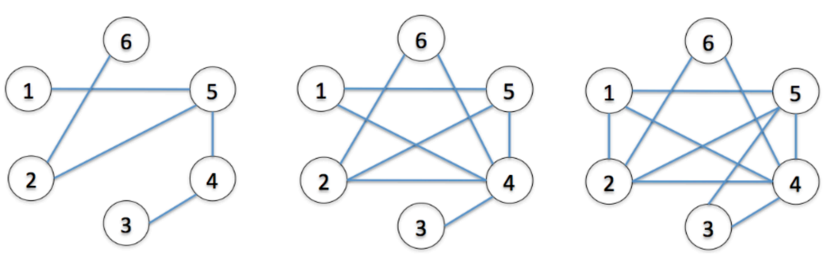
(b) 4 players, 4 actions, 4 edges



(c) 5 players, 3 actions, 5 edges

(d) 5 players, 4 actions, 6 edges

(e) 5 players, 5 actions, 7 edges



(f) 6 players, 4 actions, 5 edges

(g) 6 players, 5 actions, 8 edges

(h) 6 players, 6 actions, 10 edges

Figure 5: MDRLSA learned graphical structures

Experiment Results-2

Game	Players Number	Actions Number	Number of Edges	Runtime (Seconds)	Accuracy (%)
h1.	6	6	5	0.3255	100
h2.	6	6	10	0.3066	100
h3.	6	6	15	0.3111	100

Table 3: MDRLSA time performance on different number of influential edges: from a random graphical game 6-player, 6-action with same size of strategy profiles of 46656

Comparison

- Accuracy:

compared with Duong et al., on a random generated maximum of 6 edges is allowed for any player:

  - Duong et al.'s structural similarity ≈ 90%
  - MDRLSA's accuracy: 100%

- Time efficiency:

for a maximum of 5 edges each player [see Figure 5 (h)], running time of MDRLSA is approximately 0.3 seconds (written in Matlab), which is significantly faster than previous models (Duong et al.) above 500 seconds (written in Java)

Concluding Remarks

- Objective of MDRLSA is to be useful and practical;
- To extract influence graphs and achieve some reduction in the search space.

1. Learning the structure of the game is important.
2. Separating the structure learning and the strategy learning can be advantageous.

MDRSLA successfully achieves the stated goal to learn an approximate player influence network. Using a learned compact representation, it can:

- speed up search in the action space
- estimate the payoff for global strategy planning
- then utilize standard methods for game playing

Discussion & Future work

- Scale up MDRLSA and extend it to deal with a large number of actions or a large number of players in computer games where this abstraction technique is practical.
- Adjust the parameter δ to balance the tradeoff between the amount of computation of a game and approximation: to handle incomplete information & noise.
- Extend MDRLSA to other types of games.

Related Game Theory Models

- **Action Graph Games**: "dual" representation

a directed graph with nodes A (action choices)

Each agent's utility is calculated according to an arbitrary function of the node she chose and the numbers placed on the nodes that neighbor the chosen node in the graph.

- **Congestion Games**: by Rosenthal (1973)

Definition of players and resources, where the payoff of each player depends on the resources it chooses and the number of players choosing the same resource.

Can be represented as a graph, e.g. traffic routes from point A to point B

Related Research on Abstraction

Related abstraction techniques for game playing:

- Using Bayesian networks to represent non-linear relations / influence among players' actions (Artificial Life)

- Vorobeychik's work on learning payoff functions in infinite games

Apply to Practical Games

Settlers of Catan

Thank you.