Control and Cybernetics

vol. **44** (2015) No. 1

A process algebraic form to represent extensive games*

by

Omid Gheibi[1] and Rasoul Ramezanian[2]

[1]Computer Engineering Department, Sharif University of Technology, Iran
gheibi@ce.sharif.edu
[2]Complex and Multi Agent System Lab,
Mathematical Sciences Department, Sharif University of Technology, Iran
ramezanian@sharif.edu

Abstract: In this paper, we introduce an agent-based representation of games, in order to propose a compact representation for multi-party games in game theory. Our method is inspired by concepts in process theory and process algebra. In addition, we introduce an algorithm whose input is a game in the form of process algebra (proposed in this paper) and as an output, the algorithm finds the Nash equilibrium of the game in linear space complexity.

Keywords: extensive games, Nash equilibrium, process theory, process algebra

1. Introduction

Extensive representation form of a game is a directed graph whose nodes are players, and edges are actions. Assuming that the game has n agents, and each agent has two actions available, the game can be represented by a graph of the size of $O(2^n)$.

However, by explaining the behavior of each agent individually using an adequate process (called *process-game*), and obtaining the whole game through *parallel composition* of these *process-game*s, it is possible to represent the same game in $O(n)$ space. We take advantage of process algebra to define process-games and the appropriate notion of parallel composition for them.

The word "process" refers to the behavior of a system which is a set of actions that are performed in the system and the order of their execution. The process theory makes it possible to model the behavior of a system with enormous complexity through modeling the behavior of its components (Middelburg and Reniers, 2005). By taking advantage of process algebra - automating calculations and running algorithms using parallel computing techniques - we can code the process theory terms and definitions (Fokkink, 2007; Tadjouddine, 2008).

*Submitted: October 2013; Accepted: January 2015.

On the other hand, a game is a system and its behavior is established by the behavior of all of its players (components). Hence, the game could be studied and formally modeled as an interactive process (Van Benthem, 2002).

Moving along this path in formal methods, in order to reduce the representation of games with lots of players, we modify the process theory in an appropriate manner to provide a model called "*process-game*" that encompasses both process theory and game theory notions. This proposed process algebraic model makes it possible to have a compact representation for extensive games - specially in social extensive games which have local interaction - via appropriate parallel composition (Section 3).

There exist also other attempts to reduce the representation of games with very high number of players. In comparison with the graph-based representation that is proposed to reach the same goal (Kearns, Littman and Singh, 2001) our proposed model facilitates the reduction of representation of games in the formal method.

Eventually, to manipulate the process-game model efficiently, we propose an algorithm to find the equilibrium path of games in linear space complexity by using a revision of depth first search and backward induction (Section 4).

2. Preliminaries

In this section, we review some preliminary concepts in game and process theory.

2.1. Game-theoretic concepts

In this part we briefly review definitions and concepts of strategic and extensive games with perfect information which appeared in the literature, using the same notations as in Osborne and Rubinstein (1994), page 89.

A strategic game is a model of decision-making such that decision-makers choose their plan simultaneously from their possible actions, once and for all.

Definition 1 (Strategic Game) *A strategic game consists of:*

- *A finite set of players N*
- *for each player $i \in N$ a nonempty set of actions A_i*
- *for each player $i \in N$ a payoff function $\Pi_i : A_1 \times \cdots \times A_n \to \mathcal{R}$.*

If for each player i the action set A_i is finite, the the game is finite.

Example 1 (prisoner's dilemma) *Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement with no means of speaking to or exchanging messages with the other. The police admit they do not have enough evidence to convict the pair on the principal charge. They plan to sentence both to a year in prison on a lesser charge. Simultaneously, the police offer each prisoner a Faustian bargain. Each prisoner is given the opportunity either to betray (B) the other, by testifying that the other committed*

the crime, or to cooperate (C) with the other by remaining silent. Here's how it goes:

- *If A and B both betray the other, each of them serves 2 years in prison*
- *If A betrays B but B remains silent, A will be set free and B will serve 3 years in prison (and vice versa)*
- *If A and B both remain silent, both of them will only serve 1 year in prison (on the lesser charge).*

The above situation is shown as a strategic game in Fig.1.

	B	C
B	(-2,-2)	(0,-3)
C	(-3,0)	(-1,-1)

Figure 1: Strategic representation of Example 1

The set of players is $N = \{1, 2\}$. *The possible actions for each player come from the set of* $A_1 = A_2 = \{B, C\}$. *Each cell in the above table corresponds to a* strategy profile *and shows the resulting payoffs. A strategy profile is a set of strategies for all players which fully specifies all actions in a game. A strategy profile must include one and only one strategy for every player. In the cell, the first value is the player 1's payoff and the player 2's payoff is the second one.*

One of the most common solution concepts in game theory is Nash equilibrium. This notion captures a steady state, in which no player wants to deviate from the current state if actions of the other players are fixed (therefore, all players choose their action in a rational manner).

NOTATION 1 *For each strategy profile* s, s_{-i} *shows the action of all players, except for player* i.

DEFINITION 2 (NASH EQUILIBRIUM) *A Nash equilibrium of a strategic game* $\Gamma = \langle N, (A_i), (\Pi_i) \rangle$ *is a strategy profile* s^* *with the property that for every player* $i \in N$ *we have*

$$\Pi_i(s^*_{-i}, s^*_i) \geq \Pi_i(s^*_{-i}, s_i), \forall s_i \in A_i.$$

EXAMPLE 2 *In Example 1, strategy profile* (B, B) *is a Nash equilibrium, because, if player 1 deviates from action* B *to* C, *his payoff decreases from* -2 *to* -3 *in the strategy profile* (C, B). *Therefore, he has no motivation to deviate from the state* (B, B). *Also, because of symmetry, we can specify the same reason for player 2 to have no motivation to deviate from the current state. Therefore, strategy profile* (B, B) *is a Nash equilibrium in this game.*

Another form of games, which is the pivot of discussion in this paper, is extensive game. In this form, decision-makers act sequentially (unlike in strategic game).

Definition 3 (Extensive Game with Perfect Information) *An extensive game with perfect information is defined by a four-tuple* $\langle N, H, P, (\Pi_i)\rangle$ *which has the following properties:*

- *A set* N *of* players
- *A set* H *of sequences (finite or infinite) that satisfies the following three properties:*
  - *The empty sequence* $\emptyset$ *(the* empty history *representing the start of the game) is a member of* H*.*
  - *If* $(a^k)_{k=1,\ldots,K} \in H$ *(where* K *may be infinite) and positive integer* $L < K$ *then* $(a^k)_{k=1,\ldots,L} \in H$*.*
  - *If an infinite sequence* $(a^k)_{k=1,\ldots,\infty}$ *satisfies* $(a^k)_{k=1,\ldots,L} \in H$ *for every positive integer* L *then* $(a^k)_{k=1,\ldots,\infty} \in H$*.*

  Each member of H *is a* history *and each term of a history is an* action *which is taken by a player. A history* $(a^k)_{k=1,\ldots,K} \in H$ *is* terminal *if it is infinite or there is no* a^{K+1} *such that* $(a^k)_{k=1,\ldots,K+1} \in H$*. The set of all terminal histories is denoted by* Z*.*
- *A function* P *(the* player function*) that assigns to each nonterminal history (each member of* $H\backslash Z$*) a member of* N*,* $P(h)$ *returns the player who takes an action after the history* h *(*$P : H\backslash Z \to N$*).*
- *A function* Π *(the* payoff function*) that assigns to each terminal history (each member of* Z*) a member of* $\mathbb{R}^{|N|}$ *(*$\Pi : Z \to \mathbb{R}^{|N|}$*,* $\Pi_i(z)$ *is player* i*'s payoff in terminal history* $z \in Z$*).*

An extensive game with perfect information is *finite* if and only if the set H of possible histories is finite. Throughout this paper, whenever we use the term *extensive games*, we mean *extensive games with perfect information.* In an extensive game, $P(h)$ chooses an action after any nonterminal history h from the set $A(h) = \{a : (h, a) \in H\}$ where (h, a) means a history h followed by an action a which is one of the actions available to the player who moves after h.

Definition 4 (Strategy) *A strategy of player* $i \in N$ *in an extensive game* $\langle N, H, P, (\Pi_i)\rangle$ *is a function that assigns an action from* $A(h)$ *to each* $h \in H\backslash Z$ *(nonterminal history) for which* $P(h) = i$*.*

The outcome of a strategy profile s is a terminal history, which is constructed by s, and is denoted by $O(s)$.

Example 3 *Two people ("husband" and "wife") are buying items for a dinner party. The husband buys either fish (F) or meat (M) for the meal; the wife buys either red wine (R) or white wine (W). Both people prefer red wine with meat and white wine with fish, rather than either of the opposite combinations. However, the husband prefers meat over fish, while the wife prefers fish over meat. Assume that the husband buys the meal and tells his wife what was bought; his wife then buys some wine. If we want to consider this problem as an extensive game with perfect information we can determine its component like*

- $N = \{Wife, Husband\}$
- *The possible actions for the husband are members of the set* $A_{Husband} = \{F, M\}$*, while wife's actions come from* $A_{Wife} = \{R, W\}$*. So, in this example,* Z *is a set of sequences which are started by the action* F *or* M *and are terminated by* R *or* W*. All possible histories* H *and terminal histories* Z *are shown below:*

$$H = \{(\emptyset), (F), (M), (F, R), (F, W), (M, R), (M, W)\}$$

$$Z = \{(F, R), (F, W), (M, R), (M, W)\}.$$

- *For each* $h \in H \backslash Z$ *,* $P(h)$ *is as follows:*

$$P((\emptyset)) = Husband, P((F)) = (Wife), P((M)) = (Wife).$$

- *We can represent the preferences as utility-based payoffs:*

$$\Pi_{Husband}(M, R) = 2,\ \Pi_{Husband}(F, W) = 1,$$

$$\Pi_{Husband}(F, R) = \Pi_{Husband}(M, W) = 0,$$

$$\Pi_{Wife}(M, R) = 1,\ \Pi_{Wife}(F, W) = 2,\ \Pi_{Wife}(F, R) = \Pi_{Wife}(M, W) = 0.$$

Nash equilibrium is a common solution concept for extensive games. This concept is defined below.

Definition 5 *The strategy profile* s^* *in an extensive game with perfect information is a Nash equilibrium if for each player* $i \in N$ *we have:*

$$\Pi_i(O(s^*)) \geq \Pi_i(O(s_i, s^*_{-i})) \text{ for every strategy } s_i \text{ of player } i.$$

The notion of a Nash equilibrium ignores the sequential structure of an extensive form game and treats strategies as if they were choices made once and for all. To refine the Nash equilibrium definition we introduce a new notion of subgame perfect equilibrium (see Narahari, 2014, Chapter 3).

Definition 6 *Let* $\Gamma = \langle N, H, P, (\Pi_i)\rangle$ *be an extensive game with perfect information. The subgame of* Γ *that follows the history* $h \in H \backslash Z$ *(a nonterminal history) is the extensive game* $\Gamma(h) = \langle N, H|_h, P|_h, \Pi|_h\rangle$ *where all sequences* h' *of actions for which* $(h, h') \in H$ *are in the set* $H|_h$ *and vice versa,* $P|_h$ *is the same as function* P*, but its domain comes from the set* $H|_h$*, and for each* $h', h'' \in Z|_h$ *(the set* $Z|_h$ *consists of all the terminal histories in the set* $H|_h$*) it is defined that* $\Pi_i|_h(h') \geqslant \Pi_i|_h(h'')$ *if and only if* $\Pi_i(h, h') \geqslant \Pi_i(h, h'')$ *(note that* $(h, h'), (h, h'') \in Z$*).*

In the light of the previous considerations, we can define the notion of subgame perfect equilibrium, as mentioned before, using the notion of subgame.

Definition 7 *The strategy profile s^* in an extensive game with perfect information is a subgame perfect equilibrium if for every player $i \in N$ and every nonterminal history $h \in H \backslash Z$ for which $P(h) = i$ we have:*

$$\Pi_i|_h(O_h(s^*|_h)) \geq \Pi_i|_h(O_h(s_i, s^*_{-i}|_h))$$

for every strategy s_i of player i in the subgame $\Gamma(h)$.

Most often, an extensive game is represented by a tree with marked the subgame perfect Nash equilibria (SPNE) of the game on the tree as in (Narahari, 2014) Chapter 3 (for a better understanding see Example 3 and Fig. 2).

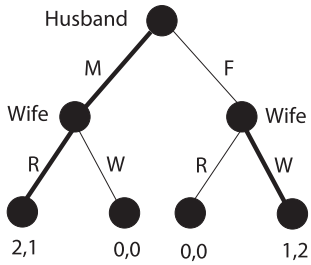


Figure 2: Game of Example 3 represented as a tree, showing the SPNEs with thick edges

The definition of extensive game with perfect information (Definition 3) can be generalized by allowing simultaneous moves of the players. This type of games is called *extensive game with imperfect information*, in this paper. An extensive game with imperfect information is determined by a quintuple $\langle N, H, P, (\mathcal{I}_i), (\Pi_i)\rangle$. Relative to the definition of an extensive game with perfect information, the new element is the collection $(\mathcal{I}_i)_{i\in N}$ of information partitions. For each player $i \in N$, $\mathcal{I}_i$ is a partition of $\{h \in H : P(h) = i\}$ with the property that $A(h) = A(h')$ whenever h and h' are in the same member of the partition. For $I_i \in \mathcal{I}_i$, we denote by $A(I_i)$ the set $A(h)$, and by $P(I_i)$ the player $P(h)$ for any $h \in I_i$ ($\mathcal{I}_i$ is the *information partition* of player i; a set $I_i \in \mathcal{I}_i$ is an *information set* of player i) (Osborne and Rubinstein, 1994).

We define a variant of the battle of sexes (BoS) game (Example 3) as an example of extensive games with imperfect information.

Example 4 *Assume in Example 3 that wife can decide whether to hold a dinner party or not. If she decides not to hold the dinner party, the game ends and nothing happens. On the other hand, there are games similar to Example 3 where players move simultaneously instead of moving sequentially. As shown in Fig. 3, simultaneous moving is specified by dashed line and means the wife does not know in which history she is.*

Formally, we have $P(\emptyset) = P(H, M) = P(H, F) =$ Wife, $P(H) =$ Husband, $\mathcal{I}_{\text{Wife}} = \{\{\emptyset\}, \{(H, M), (H, F)\}\}$, and $\mathcal{I}_{Husband} = \{\{H\}\}$.

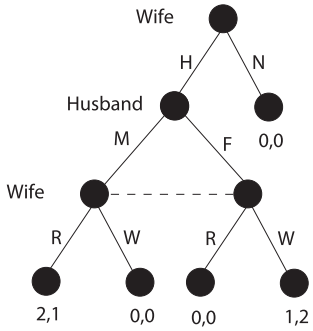


Figure 3: Game of the Example 4 represented as a tree with dashed line meaning that wife and husband move simultaneously on that level (as an example of extensive game with imperfect information).

Definition 8 *A pure strategy of player $i \in N$ in an extensive game with imperfect information $\langle N, H, P, (\mathcal{I}_i), (\Pi_i)\rangle$ is a function that assigns an action in $A(I_i)$ to each information set $I_i \in \mathcal{I}_i$.*

2.2. Process theory

The notion of a transition system can be considered to be the fundamental notion for the description of process behavior (Middelburg and Reniers, 2005). In this section we state some abstract formal definitions regarding transition systems and specify the notion of authentication using these definitions.

Definition 9 *A transition system T is a quintuple $(S, A, \rightarrow, \downarrow, s_0)$ where*

- *S is a set of states,*
- *A is a set of actions containing an internal action τ,*
- *$\rightarrow \subseteq S \times A \times S$ is a set of transitions,*
- *$\downarrow \subseteq S$ is a set of successfully terminating states,*
- *$s_0 \in S$ is the initial state.*

The set $\twoheadrightarrow \subseteq S \times A^\star \times S$ shows *generalized transitions* of T ($A^\star$ is the set of all possible chains of actions from A). A state $s \in S$ is called *reachable* state of T if there is $\sigma \in A^\star$ such that $s_0 \overset{\sigma}{\twoheadrightarrow} s$. The set of all reachable states of a transition system T is denoted by $reach(T)$. We define $act(T) = \{a \in A \mid \exists s, s' \in reach(T)\ (s, a, s') \in \rightarrow\}$. In the sequel, we assume that every transition system T is connected, i.e., $reach(T) = S$, and $act(T) = A$. If S and A are finite, T is called a finite transition system.

Notation 2 *We refer to $(s, a, s') \in \rightarrow$ by $s \overset{a}{\rightarrow} s'$.*

We define $Trace = \{\sigma \in A^\star \mid \exists s' \in S\ s_0 \overset{\sigma}{\twoheadrightarrow} s'\}$. If there exists $n \in \mathbb{N}$ such that $\forall \sigma \in Trace\ (|\sigma| \leq n)$, where $|\sigma|$ is the length of the sequence σ, then T

is called a finite-depth transition system. If for every $s \in S$, $\{(s, a, s') \in \rightarrow \mid a \in A, s' \in S\}$ is finite, then T is called a finite-branching transition system. By the notation τ, we refer to the *silent action*.

Proposition 1 *If T is both a finite-depth and a finite-branching transition system, then it is a finite transition system.*

Proof It is straightforward. □

Definition 10 *Let $T = (S, A, \rightarrow, \downarrow, s_0)$ be a transition system. Then T is deterministic if the following condition holds: whenever $s_0 \overset{\sigma}{\twoheadrightarrow} s$ and $s_0 \overset{\sigma}{\twoheadrightarrow} s'$, then $s = s'$.*

Definition 11 *Let A be a set of actions. A communication function on A is a partial function $\gamma : A \times A \rightarrow A$ such that for any $a, b \in A$: $\gamma(\tau, a)$ is not defined, and if $\gamma(a, b)$ is defined, then $\gamma(b, a)$ is defined and $\gamma(a, b) = \gamma(b, a)$. The image of γ is shown by C_γ. We define $H_\gamma = A - C_\gamma$. Assume that if $\gamma(a, b)$ is defined, then both $a, b \in H_\gamma$.*

Definition 12 (Parallel Composition) *Let $T = (S, A, \rightarrow, \downarrow, s_0)$ and $T' = (S', A', \rightarrow', \downarrow', s'_0)$ be two transition systems, and γ a communication function on a set of actions that includes $A \cup A'$. The parallel composition of T and T' under γ, written $T \parallel T'$, is the transition system $(S'', A'', \rightarrow'', \downarrow'', s''_0)$ where*

- $S'' = S \times S'$,
- $A'' = A \cup A' \cup \{\gamma(a, a') \mid a \in A, a' \in A'\}$
- $\rightarrow''$ *is the smallest subset of $S'' \times A'' \times S''$ such that:*
  - *if $s_1 \overset{a}{\rightarrow} s_2$ and $s' \in S'$, then $(s_1, s') \overset{a}{\rightarrow}'' (s_2, s')$,*
  - *if $s'_1 \overset{b}{\rightarrow} s'_2$ and $s \in S$, then $(s, s'_1) \overset{b}{\rightarrow}'' (s, s'_2)$,*
  - *if $s_1 \overset{a}{\rightarrow} s_2$, $s'_1 \overset{b}{\rightarrow} s'_2$, and $\gamma(a, b)$ is defined, then $(s_1, s'_1) \overset{\gamma(a,b)}{\rightarrow}'' (s_2, s'_2)$,*
- $\downarrow''{=}\downarrow \times \downarrow'$,
- $s''_0 = (s_0, s'_0)$.

Definition 13 (Encapsulation) *Let $T = (S, A, \rightarrow, \downarrow, s_0)$ be a transition system. Let H be a set of actions. The encapsulation of T with respect to H, written as $\delta_H(T)$, is the transition system $(S', A', \rightarrow', \downarrow', s'_0)$ where*

- $S' = S$, $A' = A$, $\downarrow'{=}\downarrow$, $s'_0 = s_0$ *and*
- $\rightarrow'{=}\rightarrow \cap (S \times (A - H) \times S)$.

Assume T_1 and T_2 are two processes, and execute them in parallel. Then for $H = A - C_\gamma$, the encapsulation of the process $T_1 \parallel T_2$ makes the processes communicate. It means that the difference between $T_1 \parallel T_2$ and $\delta_H(T_1 \parallel T_2)$ is that in the second process only communication actions exist.

Proposition 2 *If T and T' are two transition systems, and T is finite-depth, then $\delta_{H_\gamma}(T \parallel T')$ is finite-depth.*

PROOF It is straightforward. □

DEFINITION 14 *Assume two transition systems* $T = (S, A, \rightarrow, \downarrow, s_0)$ *and* $T' = (S', A', \rightarrow', \downarrow', s'_0)$*, and for* $s \in S$*, let* $l(s) = \{(s, a, t) \in \rightarrow | \ a \in A, t \in S\}$*, and for every* $s' \in S'$*,* $l'(s') = \{(s', b, t') \in \rightarrow' | \ b \in A', t' \in S'\}$*. We say that* T, T' *are* communication finite-branching *with respect to communication function* γ*, if for any* $(s, s') \in S \times S'$ *the set* $\{((s \xrightarrow{a} t), (s' \xrightarrow{b} 't)) \in l(s) \times l(s') \mid \gamma(a, b)\ is\ defined\}$ *is finite.*

PROPOSITION 3 *If two transition systems,*
$T = (S, A, \rightarrow, \downarrow, s_0)$ *and* $T' = (S', A', \rightarrow', \downarrow', s'_0)$
are communication finite-branching *with respect to a communication function* γ*, then* $\delta_{H_\gamma}(T \parallel T')$ *is finite-branching.*

PROOF It is straightforward. □

3. The processes of games

In this section, we introduce a *process-algebraic representation* for extensive games with perfect information in order to reduce the large social games into logarithmic (or polylog) size proportional to the size of *extensive representation* of the same games.

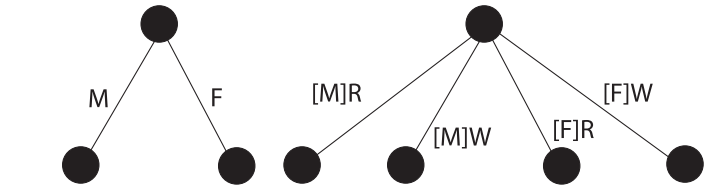
To analyse a game, we need to represent it with the game tree. If the number of agents is too large, then we may need to use a machinery to provide the game tree representation. We propose a machinery for this aim and call it *process-game*. The process game is a combination of process theory and game theory for solving games with large number of agents. We model the operation of each agent using a process-game. Then, we run all of these process-games in parallel to obtain the game tree.

DEFINITION 15 *Let* A *be a set of actions. The language* $L(A)$ *is the smallest superset of* $A^\star$ *such that*

$$\rho, \sigma \in L(A) \Rightarrow \neg\rho, (\rho \vee \sigma), mid(\sigma), pre(\sigma), pos(\sigma) \in L(A).$$

Let $T = (S, A, \rightarrow, \downarrow, s_0)$ *be a transition system. For* $s \in S$*, the history of* s*, denoted by* $h(s)$*, is the trace* $\sigma \in A^\star$ *such that* $s_0 \overset{\sigma}{\twoheadrightarrow} s$*. For a state* (T, s) *and a formula* $\sigma \in L(A)$*, we define the satisfaction* $(T, s) \models \sigma$*, as follows:*

for $\sigma \in A^\star$*,* $(T, s) \models \sigma$ *iff* $h(s) = \sigma$*,*
$(T, s) \models pre(\sigma)$ *iff* $\exists \rho \in A^\star (h(s) = \sigma\rho)$*,*
$(T, s) \models mid(\sigma)$ *iff* $\exists \rho, \varsigma \in A^\star (h(s) = \varsigma\sigma\rho)$*,*
$(T, s) \models pos(\sigma)$ *iff* $\exists \rho \in A^\star (h(s) = \rho\sigma)$*,*
$(T, s) \models (\rho \vee \sigma)$ *iff* $(T, s) \models \rho$ *or* $(T, s) \models \sigma$*,*
$(T, s) \models \neg\rho$ *iff* $(T, s) \not\models \rho$*.*



(a) husband process (b) wife process

Figure 4: Processes of Example 5

Definition 16 *Let A and B be two sets of actions. The set of conditional actions over A with conditions in B, denoted by $A_{con(B)}$ is defined as follows:*

for each $a \in A$, and $\sigma \in L(B)$, $[\sigma]a$ is a conditional action, i.e., $[\sigma]a \in A_{con(B)}$.

There is an injective mapping from A to $A_{con(B)}$ which maps each $a \in A$ to $[\top]a \in A_{con(B)}$.

Definition 17 (Encapsulation of Conditional Actions) *Let A and B be two sets of actions and $T = (S, A_{con(B)}, \rightarrow, \downarrow, s_0)$ be a transition system over conditional actions $A_{con(B)}$. The encapsulation of conditional actions of T, written $\delta^c(T)$, is the transition system $(S', A', \rightarrow', \downarrow', s'_0)$ where*

- *$S' = S$, $A' = A$, $\downarrow'=\downarrow$, $s'_0 = s_0$ and*
- *for $s, t \in S'$ and $a \in A$, $s \xrightarrow{a}{}' t$ iff for some $\sigma \in L(B)$, $s \xrightarrow{[\sigma]a} t$ and $(T, s) \models \sigma$.*

Now we give an example to illustrate the above definitions.

Example 5 *Consider Example 3. We may consider two processes, one for the husband and one for the wife*

$husband := M + F$ (Fig. 4(a)), and
$wife := [M]R + [M]W + [F]R + [F]W$ (Fig. 4(b)).

Finally, the transition system of the process $\delta^c(husband \parallel wife)$ is exactly the tree of the game (Fig. 2). The process algebraic form of the tree of the game is $M.(R + W) + F.(R + W)$.

Definition 18 *Let $T = (S, A, \rightarrow, \downarrow, s_0)$. The transition system $\delta_{\cup}(T)$ is a transition systems obtained from T by cutting some of its transitions in the following way:*

$$s \xrightarrow{a} t \;\; and \;\; b \subsetneq a \;\; then \;\; s \not\xrightarrow{b}.$$

Assume that we are given a $\Gamma = \langle N, H, P, (\Pi_i)\rangle$, which is an extensive game with perfect information. Now we model it using process theory as a process-game by mapping each player (in N) to a process, each terminal state to a member of $\downarrow$, and payoff function (Π) to profit value for each member of $\downarrow$.

Definition 19 (Process-game) *Let* $\Gamma = \langle N, H, P, (\Pi_i)\rangle$ *be an extensive game with perfect information. A process-game model for* Γ *is a tuple*

$$\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N}\rangle$$

where each $T_i = (S^i, A^i_{con(B)}, \rightarrow^i, \downarrow^i, s^i_0)$ *is a transition systems and each*

$$\pi_i : A^\star \rightarrow \mathbb{R}$$

is a profit function. $A_1, A_2, \ldots, A_n$ *are conditioned by* B $(A^1_{con(B)}, \ldots, A^n_{con(B)})$ *so that*

1. *if* $i \neq j$ *then* $A_i \cap A_j = \emptyset$,
2. $B = (\bigcup_{i\in N} A_i) \cup A_1 \times A_2 \times \cdots \times A_n$.

and the game Γ *is mapped to process-game* $\mathfrak{P}$ *so that*

- S^i *is a set of states where at each state player* $i \in N$*, decides to perform one of his/her possible actions, which is determined by* P *for each node on the game tree,*
- A^i *is a set of actions containing an internal action* τ *that represents possible actions of player* i*,*
- $\rightarrow^i \subseteq S^i \times A^i_{con(B)} \times S^i$ *is a set of transitions that represents what happens when a player chooses one of his actions to do (using* P*),*
- $\downarrow^i \subseteq S^i$ *is a set of successfully terminating states that represents terminal states on the game tree, which can be defined by terminal histories* $(O(s))$*,*
- $\pi_i : A^\star \rightarrow \mathbb{R}$ *is a payoff function that represents payoff* (Π_i) *for each action of the player* i *in each* subprocess *(like a subgame) which is started by* i*.*
- $s^i_0 \in S^i$ *is the set of initial states, which a player* i *can choose to start his game from.*

Now we can construct the process tree of the game using $\delta_\cup(\delta^c(T_1 \parallel T_2 \parallel ... \parallel T_n))$.

The sizes of A_i and N (set of players or agents) in T_i and π_i are denoted by $|A_i|$, n, and $|\pi_i|$ respectively. Let $d = \max_i(|A_i|)$, which is called the branching factor of the process/game tree.

Theorem 1 *Suppose a process-game* $\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N}\rangle$ *with*

$$T_i = (S^i, A^i_{con(B)}, \rightarrow^i, \downarrow^i, s^i_0)$$

is given. The size of the equivalent extensive representation of $\mathfrak{P}$*, which is denoted by* $|ERT|$*, is* $O\left(d^{n+1}\right)$.

Proof The size of the extensive representation is equal to the size of the tree of the game. As players act sequentially, the maximum number nodes in the first level of the tree is $|A_1|$, in the second level would be $|A_1| \times |A_2|$ and so on. Therefore, we have

$$\begin{aligned} |ERT| &\leq |A_1| + |A_1| \times |A_2| + \cdots + |A_1| \times |A_2| \times \cdots \times |A_n| \\ &\leq d + d^2 + \cdots + d^n = O\left(d^{n+1}\right). \end{aligned}$$

□

Theorem 2 *The size of a given process-game* $\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N}\rangle$ *with* $T_i = (S^i, A^i_{con(B)}, \rightarrow^i, \downarrow^i, s^i_0)$ *is* $O(nd|B| + \sum_{i=1}^{n} |\pi_i|)$.

Proof The size of $\mathfrak{P}$ is equal to the sum of $|T_i|$ and $|\pi_i|$ for all players. The size of T_i is $O(|A^i_{con(B)}|)$. Therefore,

$$\sum_{i=1}^{n} |T_i| \leq \sum_{i=1}^{n} |A^i_{con(B)}| \leq \sum_{i=1}^{n} |A_i| \times |B| \leq nd|B|, \tag{1}$$

$$|\mathfrak{P}| = \sum_{i=1}^{n} |T_i| + \sum_{i=1}^{n} |\pi_i|. \tag{2}$$

We can conclude from equations (1) and (2) that

$$|\mathfrak{P}| = O(nd|B| + \sum_{i=1}^{n} |\pi_i|). \qquad \square$$

Assume that all of the actions of players are in the form of a or $[b]a$ (without condition or just with one condition), therefore the size of B for each player is $O(d)$. According to Theorem 2, it is easy to see that the size of the process-game representation would be $O(nd^2 + \sum_{i=1}^{n} |\pi_i|)$. Based on the assumption, the size of the process-game can be logarithmic proportional to the size of the extensive representation (as d is the maximum number of plsyers' actions, it is a constant). The following equation shows this fact.

$$|\mathfrak{P}| = O(nd^2 + \sum_{i=1}^{n} |\pi_i|) = O(\log(d^n)\frac{d^2}{\log(d)} + \sum_{i=1}^{n} |\pi_i|) = O(\log(|ERT|) + \sum_{i=1}^{n} |\pi_i|).$$

Extended version

We can extend the definition of process-game for extensive games with imperfect information in the following manner.

Notation 3 *Let* A_1 *and* A_2 *be two disjoint sets. For* $(a, b) \in (A_1 \cup 2^{A_1}) \times (A_2 \cup 2^{A_2})$, *we define* $a\dot{\cup}b :=$

$$\begin{array}{ll} \{a\} \cup \{b\} & \text{if } (a, b) \in A_1 \times A_2, \\ \{a\} \cup b & \text{if } (a, b) \in A_1 \times 2^{A_2}, \\ \{b\} \cup a & \text{if } (a, b) \in 2^{A_1} \times A_2, \\ a \cup b & \text{if } (a, b) \in 2^{A_1} \times 2^{A_2}. \end{array}$$

Using the above notation, we modify Definition 11 over n disjoint sets of actions in the following definition.

Definition 20 *Let* $A_1, A_2, \ldots, A_n$ *be* n *disjoint sets of actions.*

$$\gamma : (A_1 \cup 2^{A_1}) \times (A_2 \cup 2^{A_2}) \times \cdots \times (A_n \cup 2^{A_n}) \rightarrow 2^{A_1 \cup A_2 \cup \cdots \cup A_n}$$

is a communication function, which is defined as

$$\gamma(a_1, a_2, \ldots, a_n) := a_1 \dot{\cup} a_2 \dot{\cup} \cdots \dot{\cup} a_n$$

for all $(a_1, a_2, \ldots, a_n) \in (A_1 \cup 2^{A_1}) \times (A_2 \cup 2^{A_2}) \times \cdots \times (A_n \cup 2^{A_n})$.

We may extend γ to the set of conditional actions in the manner as that given in the following definition:

DEFINITION 21 *Let* A_1, A_2, B *be three disjoint sets of actions.* γ *is a communication function over conditional actions* $A^1_{con(B)}$ *and* $A^2_{con(B)}$ *for* $([\sigma]a, [\sigma]b) \in A^1_{con(B)} \times A^2_{con(B)}$ *,which is defined as follows:*

$$\gamma([\sigma]a, [\rho]b) := [\sigma \wedge \rho]\gamma(a, b).$$

This communication function helps to model simultaneous players' moves in extensive games with imperfect information (See Example 4).

EXAMPLE 6 *The process of each player in Example 4 is as given below:*

$wife := (H + N).(R + W)$,

$husband := [H]M + [H]F$.

and γ *is defined over* $A_w = \{R, W\}$, $A^h_{con(B)} = \{[H]M, [H]F\}$, *and* $B = \{H\}$. $\gamma([H]M, R)$, *for instance, is equal to* $[H]\{M, R\}$. *The transition system of the process over* γ *communication function is* $\delta^c(wife \parallel husband)$ *which is exactly the tree of the game. The tree is shown in Fig. 5, which is equivalent to Fig. 3.*

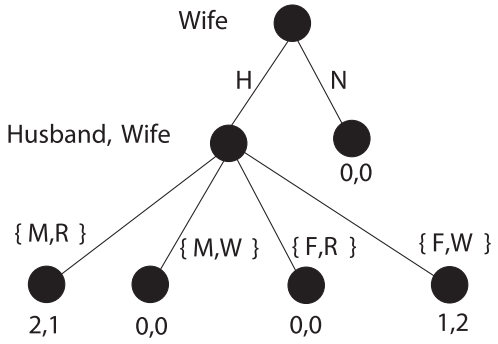


Figure 5: This figure shows simultaneous moves of wife and husband in the second level of the game and is equivalent to Fig. 3

Now, we can define the *extended process-game* by a tuple

$$\mathfrak{P} = \langle (T_i)_{i \in N}, (\pi_i)_{i \in N}, \gamma \rangle$$

for a given extensive game with imperfect information $\Gamma = \langle N, H, P, (\mathcal{I}_i), (\Pi_i) \rangle$. Relative to the definition of a process-game, the new element is the γ as a

communication function over $A_1, A_2, \ldots, A_n$ which are conditioned by B. The definition of γ is based on $\mathcal{I}_i$.

In the extended version, the process/game tree is constructed with the same method as in process-game ($\delta_{\cup}(\delta^c(T_1 \parallel T_2 \parallel ... \parallel T_n))$). Also, if the size of γ is denoted by $|\gamma|$, we have

$$|\mathfrak{P}| = \sum_{i=1}^{n} |T_i| + \sum_{i=1}^{n} |\pi_i| + |\gamma|. \tag{3}$$

Similar theorems as before can be proven for the extended process-game by introducing equation (3) into those theorems.

Applications

One of the applications of process-game in representing the real social extensive games in log or polylog size could be in modeling of social systems where agents have *local interaction* and *local competition* (these two terms might be used interchangeably, conform to the convention adopted here). A significant approach which is founded on the assumption of local interaction is generative social science.

Generative social science (Epstein, 2006) is an approach to study social systems via agent-based computational models. Its aim is to answer the question as to how the regularity of the society emerges from the local interaction of heterogeneous autonomous agents. One of the main assumptions of generative social science is called "local interaction" (see Epstein, 2006, page 6). According to this assumption, agents interact with their neighbors and compete in the social network, in which they are involved.

On the other hand, social extensive games - each player is viewed as being involved in a local competition with the players in geographically neighboring regions - can be modeled as a graph G (Kearns, Littman, and Singh, 2001). In the graph-based model, each player is represented by a vertex in G. The interpretation of edges is that each player is in a game only with the respective neighbors in G.

In the above vein, it can be proposed that lots of social systems are based on local interactions and competitions. The observation forwarded below stipulates that in considering local interaction in the form of extensive games, the size of the process-game can be logarithmic in comparison with the extensive representation.

Observation 1 Suppose there is an extensive game with perfect information and the interaction given by the graph-based model G. The maximum degree of G is denoted by Δ. Actions of each player are limited just to the player's neighbors in G (the size of B in the equivalent process-game would be $O(d^{\Delta})$). Hence, the size of the process-game representation would be $O(nd^{\Delta+1} + \sum_{i=1}^{n} |\pi_i|)$. As

a consequence, we have

$$|\mathfrak{P}| = O(nd^{\Delta+1} + \sum_{i=1}^{n} |\pi_i|) = O(\log(d^n)\frac{d^{\Delta+1}}{\log(d)} + \sum_{i=1}^{n} |\pi_i|) =$$

$$O(\frac{d^{\Delta+1}}{\log(d)}\log(|ERT|) + \sum_{i=1}^{n} |\pi_i|).$$

As $d, \Delta \ll n$ is a common situation in social extensive games, the size of the process-game can be logarithmic (or polylog, depending on d and Δ) with respect to the size of the extensive representation. For instance, considering local competition property, we have $\Delta = O(\log(n))$, then $\frac{d^{\Delta+1}}{\log(d)} = O(n) = O(\log(|ERT|))$. Therefore, $|\mathfrak{P}| = O(\log^2(|ERT|) + \sum_{i=1}^{n} |\pi_i|)$.

Note that local competition is a kind of local interaction. Recalling the assumption of local interaction in generative social science, we can come to the conclusion that it is obvious that in lots of social systems $\Delta = O(\log(n))$.

There have been various representations of extensive games aiming to represent the large games compactly, such as the graphical model (Kearns, Littman, and Singh, 2001) and MAIDs (Koller and Milch, 2003). However, our proposal (originating from the process theory) of producing a compact representation, is quite different. Also, we bring the advantages of process theory (in execution and management) to the game theory environment.

Our model can be applied in management of complex systems too, but how? Each process has its own profit and each profit is a function of some inputs. Suppose a process game is defined with some initial values and may deliver some equilibria path as a result (using the algorithm of the following section). However, for a manager, it would be critical to control the path as he wants it to be. Therefore he can try changing the inputs (so the profits will be changed, leading also to some new equilibria path) up to getting the best one.

4. The algorithm

The notion of process-game has been explained completely in the previous section. As strategies and agents in the process-game are the same as in its equivalent extensive game, therefore the subgame perfect equilibrium (or equilibrium path) is a solution concept for the process-game, too. Now, in this section, we propose an algorithm to find the equilibrium path in a process-game. Algorithm 1 illustrates the finding of equilibria path under the assumption that there exists only one equilibrium path for each *subprocess* (like a subgame).

REMARK 1 *Equilibrium path in a process-game* $\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N} \rangle$ *is a sequence of actions from* s_0^1 *(suppose player* 1 *starts the entirety of the game) to* $\downarrow^n$ *(suppose the last action is for player* n*).*

Algorithm 1 Depth-First Finding Equilibria

Input: A Process-Game $\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N}\rangle$
Output: Equilibria path Λ

1: **for** $s^i_j \leftarrow$ each state visited in depth-first expansion of $\mathfrak{P}$ from s^1_0 using $A^i_{con(B)}$ **do**
2: $isVisited[s^i_j] \leftarrow false$
3: $ratPath[s^i_j] \leftarrow \varnothing$
4: **if** $s^i_j \in{\downarrow^n}$ **then**
5: $isVisited[s^i_j] \leftarrow true$
6: **else**
7: **if** $\forall s', a : (s^i_j, a, s') \in{\rightarrow^i} \wedge isVisited[s']$ **then**
8: $a \leftarrow \arg\max_a \pi_i(a.ratPath[s']) \triangleright s^i_j \xrightarrow{a} s' \in{\rightarrow^i}$
$\triangleright$ choose a possible action $a \in A^i_{con(B)}$ from state $s^i_j \in S^i$ which maximizes profit of player i in the state s^i_j
9: $ratPath[s^i_j] \leftarrow a.ratPath[s']$
10: $isVisited[s^i_j] \leftarrow true$
$\triangleright$ the assumption is that just one equilibrium path exists for each *subprocess*
11: delete the $ratPath$ for all s'.
$\triangleright$ by deleting $ratPath$ for s' the space complexity will remain linear and $ratPath$ is propagated toward s^1_0 states.
12: **end if**
13: **end if**
14: **end for**
15: **return** $ratPath[s^1_0]$

In line 1 of Algorithm 1, expansion takes place by virtue of conditional actions. As in Algorithm 1 each player's payoff value is calculated bottom-up, it is sufficient to save players' payoff in the subprocess equilibria at each level. To reuse space and keep the space required by the algorithm linear, we delete all process nodes which are expanded in that subprocess previously in line 11 of Algorithm 1. We present below an example, meant to clarify how this algorithm works.

Example 7 *Two people select a policy that affects them both by alternately vetoing the available (voted) policies until only one remains. First person 1 vetoes a policy. If more than one policy remains, person 2 then vetoes a policy. If more than one policy still remains, person 1 then vetoes another policy. The process continues until only one policy has not been vetoed. Suppose there are three possible policies, X, Y, and Z, person 1 prefers X to Y to Z, and person 2 prefers Z to Y to X (Osborne, 2004). Now, we want to represent this situation through the process-game, as a compact representation proportional to the extensive model. To define a process-game $\mathfrak{P} = \langle (T_i)_{i\in\{1,2\}}, (\pi_i)_{i\in\{1,2\}}, \gamma\rangle$,*

we should first determine the transition system T_i. Players' process model is specified below, with $P1$ and $P2$ being the processes of person 1 and person 2, respectively. T_1 and T_2 can be obtained by decoding these process models.

$P1 := X + Y + Z$

$P2 := [X]Y + [X]Z + [Y]X + [Y]Z + [Z]X + [Z]Y$.

Definitions of π_1 and π_2 are based on the players' preferences. The payoff function of each player on each subprocess is equal to the player's payoff over the complete path from the root to the leaf, passing through the subprocess. This path for each subprocess p is denoted by path(p). Therefore,

$$\pi_1(p) = \begin{cases} 2 & \textit{if there is no } X \textit{ in the } path(p) \\ 1 & \textit{if there is no } Y \textit{ in the } path(p) \\ 0 & \textit{if there is no } Z \textit{ in the } path(p) \end{cases},$$

$$\pi_2(p) = \begin{cases} 2 & \textit{if there is no } Z \textit{ in the } path(p) \\ 1 & \textit{if there is no } Y \textit{ in the } path(p) \\ 0 & \textit{if there is no } X \textit{ in the } path(p) \end{cases}.$$

Actually, we define the above functions compactly. Thus, their space complexity is much lower than when defining the function for each path separately. Now, let us compute the Nash equilibrium. We know that the process tree is constructed completely by $\delta_{\cup}(\delta^c(P1 \parallel P2))$. However, using Algorithm 1, the process tree is expanded step by step to save the space. The steps of the DFS expansion of the process tree are sketched in Fig.6.

As it is not necessary that there exist any pure equilibrium path for an extensive game with imperfect information, this algorithm does not work, in general, for the extended process-game. However, we can detect such a situation in the bottom-up calculation and report "no pure equilibrium path".

Complexity

Time complexity of Algorithm 1 in the worst case would be in the NP-complete complexity class, like for backward induction (Nisan, Roughgarden, Tardos, and Vazirani, 2007) (because we want to find pure Nash equilibria). Its space complexity is linear in the size of the game, which is given as an input, i.e., linear in the depth of the process-game, maximum number of actions which can be performed by a player, and the size of the payoff function.

In the extended version, if the number of simultaneous moves grows, the extensive game will be transforming to the pure strategic form, so that in the worst case, its space complexity would be exponential.

The algorithm is like a depth-first search and the space complexity of depth-first search is $O(hd)$, where h is the height of the tree and d is the branching factor that is the maximum number of actions which a player can perform. However, there is a bottleneck, which is caused by the size of the payoff function. If it has exponential size, as space complexity is linear with respect to the size

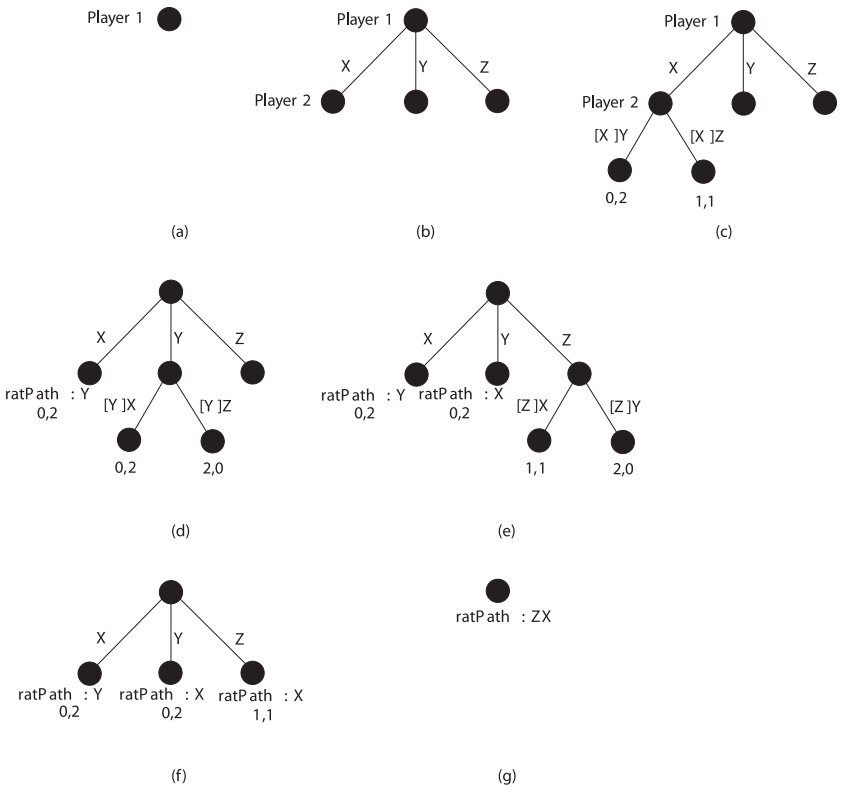


Figure 6: Graphs (a) to (g) show the steps of Algorithm 1 in finding the Nash equilibrium for the process model of Example 7. At step (g) the rational path of the s_0^1 (ZX) is the Nash equilibrium

of the payoff function too, it will be exponential too. Therefore, the algorithm will be better than backward induction or using strategic form algorithms in terms of space complexity under two circumstances. First, the size of the payoff function is polynomial in n and d. Second, the size regarding the simultaneous moves (size required to represent the function of γ) is polynomial in n and d (in the case of extended version).

Theorem 3 *For a given extended process-game* $\mathfrak{P} = \langle (T_i)_{i\in N}, (\pi_i)_{i\in N}, \gamma \rangle$ *as an input, the space complexity of Algorithm 1 is* $O(nd + |\mathfrak{P}|)$.

Proof At each step of Algorithm 1, one process node is expanded and finally collapsed when all its subnodes are visited. Therefore, when Algorithm 1 is running, at all levels of the tree, at most one node is expanded. We know that the height of the tree is at most n and the branching factor of the tree is d. Therefore, the allocated space for expanding the process tree during the running in all steps would be $O(nd)$ in the worst case. Hence, the space complexity of

the algorithm is $O(nd + \text{size of input}) = O(nd + |\mathfrak{P}|)$. □

Actually, the process-game is a simple case of the extended process-game with $|\gamma| = 0$. Hence, the above theorem is in a general case true for the process-game, too.

5. Conclusions and future work

We introduced a new model to represent large extensive games in a compact representation (especially the social extensive games with local competition). In addition, the model is defined in algebraic terms and can be ran in parallel mode. Further, we provide an algorithm to find the Nash equilibrium for this representation having linear space complexity, with respect to the size of the input.

As we mentioned, one of the applications of the model can be in management of complex systems. However, there is no software to facilitate the management process for the manager. Therefore, the next step of the work may be develop a software to approach this particular goal.

Acknowledgment

We would like to thank anonymous referees for their helpful comments and suggestions which resulted in a better presentation of the ideas in our paper.

6. References

EPSTEIN, J. M. (2006) *Generative Social Science: Studies in Agent-based Computational Modeling*. Princeton University Press.

FOKKINK, W. (2007) *Introduction to Process Algebra*. Springer-Verlag, 2nd edition.

KEARNS, M., LITTMAN, M. L., and SINGH, S. (2001) Graphical models for game theory. In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufman Pubs. Inc., 253–260.

KOLLER, D. and MILCH, B. (2003) Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45(1):181–221.

MIDDELBURG, C. A. and RENIERS, M. A. (2005) *Introduction to Process Theory*. Technische Universiteit Eindhoven.

NARAHARI, Y. (2014) *Game Theory and Mechanism Design*. World Scientific Publishing Company.

NISAN, N., ROUGHGARDEN, T., TARDOS, E., and VAZIRANI, V. V. (2007) *Algorithmic Game Theory*. Cambridge University Press.

OSBORNE, M. J. (2004) *An Introduction to Game Theory*. Oxford University Press.

OSBORNE, M. J. and RUBINSTEIN, A. (1994) *A Course In Game Theory*. The MIT Press, 1st edition.

TADJOUDDINE, E. M. (2008) Automated mechanism design using process algebra. In: *AISB 2008 Convention Communication, Interaction and Social Intelligence. Journal of AISB*, 1, 8.

VAN BENTHEM, J. (2002) Extensive games as process models. *Journal of Logic, Language and Information*, **11**(3):289–313.