The relation between protocols and games[*]

Johannes Reich

Gerbersruhstraße 147, 69168 Wiesloch
johannes.reich@sophoscape.org

Abstract: Both, games in a game theoretic sense and protocols in an informational sense describe rule based interactions between systems. Some similarities and differences of both approaches are explored and illustrated with the example of the well known game tic tac toe.

The main thesis of this article can be roughly states as “protocols, enriched by decisions are games without payoff evaluation”. Introducing decisions as an additional input alphabet to determine the usually nondeterministic transition relation of a protocol leads to a classification of decisions as being either spontaneous (or inducing) or selection decisions.

Relating protocols and games, the complementarity of the focus of current game theory and informatics becomes better visible: the focus of current game theory to find distinguished strategies within single interactions requires the introduction of some often quite arbitrary payoff function for optimization purposes. The focus of current informatics to solve the coordination problem for finite systems, that is to determine the nondeterminacies of single interactions by other interactions may contribute to an inappropriate disregard of the decision and thereby the strategy concept of game theory.

1 Introduction

Game theory as it was initiated by von Neumann (vNM90) is a mathematical theory of social interaction. It’s subject are decision situations, where the result for each participant may depend not only on their own, but usually also on the decisions of the other participants. It has become the dominant model in economic theory.

In theoretical informatics, rule based stateful nondeterministic interactions between information processing systems are described by so called “protocols” (e.g. Boc78). According to Holzmann (Hol91), Scantlebury and Bartlett (SB67) introduced the term pprotocolïn 1967 to denote a process-like data exchange.

Games and protocols obviously stem from different scientific domains. However, both describe the interaction of systems. The main thesis of this article could be roughly stated as “protocols, enriched by decisions are games without payoff evaluation”. As a consequence, the relation between informational protocols and game theoretic games allows the

[*]Published in S. Fischer, E. Maehle and R. Reischuk (Eds), GI Lecture Notes in Informatics, Proceedings of the 39. Annual Conference of the German Gesellschaft für Informatik e.V. 2009 in Lübeck, pp. 3453-3464.

study of the nature of decisions. As a byproduct, we get yet another interesting formalism for representing games.

The relation between games and informational entities like processes or protocols has already been explored by other researchers. van Benthem (Ben02) introduced games as a process model and investigated the question, "when are two games equal?" with means of bisimulation analysis, the hallmark of logical process theories.

Ghoulalmi-Zine and Arrar (GZA05) modeled net-banking systems as a game in the sense of game theory. They explain that "The protocol game of an exchange protocol is intended to model all the possible interactions of the (potentially misbehaving) protocol parties. The correct behavior of each party is represented by a particular strategy within the protocol game". However, they restricted their model on synchronous interactions, where the protocol participants interact with each other in rounds.

J.J. Kline and M. Kaneko (KK07) introduced a new mathematical representation of an extensive game situation, called "information protocol". Such an "information protocol" consists of a set of information pieces, a set of actions and and a causality relation, linking the processing of the information pieces by the actions together. In fact, they use the "protocol" term in a different sense than it is proposed in this article, where "protocols" are explicitly related to the interactions of systems.

In the first two parts of the article, the formal descriptions of games and protocols are introduced. The protocol structure is extended by decisions and thereby becomes mappable onto the game structure. All steps are illustrated by the example of the tic tac toe game. Last but not least, the result of a unified view on interactions is discussed.

By convention, the components of a mathematical structure may be denoted by the structure's symbol or index as subscript. For any alphabet set A, $A^\epsilon := A \cup \{\epsilon\}$ with ϵ is the empty word. For state value sets Q, $Q^\epsilon := Q \cup \{\epsilon\}$ with ϵ is the undefined value. $\vec{p}\left[\frac{q_k}{p_k}\right]$ denotes a vector $\vec{q}$ which is identical to $\vec{p}$ in all positions except position k where p_k is replaced by q_k. Because of the space restrictions, proofs are only sketched.

2 Games

One common way to describe games is the "extensive form" (vNM90; Kuh53; Sel75). Here, a game is specified as a tree where each node represents a state of play assigned to a unique player. Each edge represents a move in the sense of a state transition based either on a decision or on a transition probability. Every opportunity to move, the type of action as well as the available information has to be given. The course starts at a unique initial node and ends at a terminal node, which has a payoff assigned for the series of moves it represents. As a player may not be able to observe the choice of another player because of hidden or simultaneous moves, nodes belonging to one player can be aggregated to information sets. The player to whom the information set belongs cannot distinguish between the nodes of the set or in other words cannot base any of his decisions on any difference between these nodes.

Definition 1: The *extensive form representation* of a finite game is a structure $\mathcal{G} = (\mathcal{K}, N, X, \sim, R, C, u)$ where

1. $\mathcal{K} = (V, v_0, T, p)$ is a finite tree with a set of nodes V, a unique initial node $v_0 \in V$, a set of terminal nodes $T \subset V$ and an immediate predecessor function $p : V \setminus \{v_0\} \rightarrow D$ with the set of decision nodes $D = V \setminus T$. The set of edges is defined as $E = \{(a, b) \in V \times V | a = p(b)\}$

2. $N = \{1, 2, \ldots n\}$ is a finite set of players together with a partition $X = \{X_i\}_{i \in N}$ of the set of decision nodes D.

3. $\sim = \{\sim_i\}_{i \in N}$ is a set of equivalence relations, indicating that a player i cannot distinguish between two nodes of an equivalence class, defined by the equivalence relation. Each $\sim_i \subseteq X_i \times X_i$ is required to fulfill: if $v, v' \in X_i$ and $v \sim_i v'$ then the number of immediate successors of v is equal to the number of immediate successors of v'. The equivalence classes of $\sim_i$ partition X_i and are called information sets. The set of all information sets of a player is denoted by $\mathbb{H}_i$.

4. $C = \{C_i\}_{i \in N}$ is the set of choice (or decisions) sets of each player $i = 1 \ldots n$. All C_i are finite.

5. $R = \{R_i\}_{i \in N}$ is a set of transition relations with $R_i \subseteq E \times C_i$. The transition relations satisfy the following two constraints: first, each choice selects a unique successor node, that is formally, if $(p, q, c) \in R_i$ and $(p, q', c) \in R_i$ then $q = q'$. Second, a choice has to be available to every node of an information set which formally means that if $(p, q, c) \in R_i$ and $p \sim_i p'$ then there exists a q' such that $(p', q', c) \in R_i$.

6. $u = (u_i)_{i \in N} : T \rightarrow \mathbb{R}^N$ is a set of payoff functions, assigning each terminal node a payoff value for each player.

2.1 Example: Tic tac toe as a game described in its extensive form

The game concept is illustrated by the game tic tac toe. As a game of perfect information, where all players know all moves that take place, tic tac toe does not illustrate the concept of information sets. Thus, the equivalence relation $\sim$ and the partition X can be disregarded.

As shown in Fig. 1, the tic tac toe game is about a board of 3×3 boxes where two players are allowed to mark a box with their sign, usually an x and an o, one at a move. The player who is first to have three of his symbols in a row, column or diagonal wins the game.

In the first move, player 1 can choose out of 9 positions. In the next move, player 2 can choose out of 8 remaining positions, and so forth.

To define the game, I use the following expressions relating to the board state: c_1 is true if player 1 has won, c_2 is true if player 2 has won, c_3 is true if there is a draw, that is if 5 x and 4 o are marked and c_1 and c_2 are false.

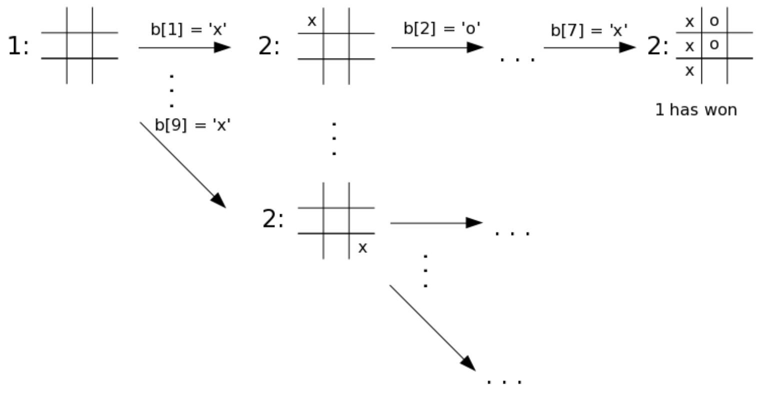

Figure 1: Tic tac toe game in its extensive or tree form.

Definition 2: A game $\mathcal{G} = ((V, v_0, T, p), N, R, C, u)$ is called *tic tac toe* if it can be described as following: $N = \{1, 2\}$. $V = \{1, 2\} \times \{\_, \mathrm{o}, \mathrm{x}\}^9$. $v_0 = (1, ((\_, \_, \_), (\_, \_, \_), (\_, \_, \_)))$. T is the set of nodes where c_1 is true and one o less then x are marked, or c_2 is true and as many o as x are marked, or c_3 is true. The transition relation R is defined by all triples (x, x', i), where a state $x' = (p', b')$ is a successor state of $x = (p, b)$, iff $p = 1$ and $p' = 2$ and $b' = b$, except in one place i where $b[i] = \_$ and $b'[i] = \mathrm{x}$ or $p = 2$ and $p' = 1$ and $b' = b$, except in one place i where $b[i] = \_$ and $b'[i] = \mathrm{o}$. The predecessor x of a state x' is then given by the first element of the triple of the successor relation with the second element being x'. $C = \{1, \ldots, 9\}$ indicate the chosen positions. R_1 is the subset of the successor relation S, where for all (x, x', i) is $player(x) = 1$. For R_2 respectively. u may be defined as $(1, -1)$ for all game winning sequences for player 1, $(-1, 1)$ for all game winning sequences for player 2 and $(0, 0)$ for a draw.

3 Protocols

Protocols can be viewed as descriptions of interactions between finite systems by iteratively linking the output of a state transition of one system to the input of a state transition of the other system (e.g. Hol91).

Definition 3: A *finite system* is defined by a tuple $\mathcal{S} = (T, succ, Q, I, O, x, in, out, f)$. T is the enumerable set of time values starting with 0 such that $succ : T \to T$ is the invertible time successor function. Q, I and O are the finite sets of state values for the internal, input and output states $(x, in, out) : T \to (Q^\epsilon, I^\epsilon, O^\epsilon)$. $f = (f^{ext}, f^{int}) : I^\epsilon \times Q^\epsilon \to O^\epsilon \times Q^\epsilon$ is a partial function describing the time evolution or system operation triggered by an update of its input parameters and updating the internal and output state in one time step with $(out(t'), x(t')) = \big(f^{ext}(in(t), x(t)), f^{int}(in(t), x(t))\big)$ for each $t \in T$ with $t' = succ(t)$.

Definition 4: A *behavior* or *trace* of a finite system (or parts of it) is described by the sequence of incoming and outgoing characters $(in_0, out_0, in_1, out_1, \ldots)$. ϵ-values do not contribute to the behavior.

The behavior of a finite system can be prescribed with nondeterministic finite I/O automata. In the sense that the unobservable initial state value is part of the automata structure, "prescribing" is more than just "describing" observable behavior.

Definition 5: A *nondeterministic finite I/O automaton (NFIOA)* is defined by a tuple $\mathcal{A} = (Q, I, O, q_0, Acc, \Delta)$. Q is the non-empty finite set of state values, I and O are the finite input and output alphabets where at least one of both is non empty, q_0 is the initial state value, Acc is the acceptance component and $\Delta \subseteq Q \times Q \times I^\epsilon \times O^\epsilon$ is the transition relation.

In case that for each $(p, i) \in Q \times I$ there is at most one transition $(p, q, i, o) \in \Delta$ then Δ defines a partial function $\delta : Q \times I \to Q \times O^\epsilon$ with $(q, o) = \delta(p, i)$. We then have a deterministic automaton or *DFIOA*.

Definition 6: A NFIOA $\mathcal{A} = (Q_\mathcal{A}, I_\mathcal{A}, O_\mathcal{A}, q_0, Acc, \Delta)$ *prescribes* the behavior of a projection of a system $\mathcal{S} = (T, succ, Q_\mathcal{S}, I_\mathcal{S}, O_\mathcal{S}, x, in, out, f)$, if a projection function $\pi = (\pi_Q, \pi_I, \pi_O) : Q_\mathcal{S} \times I_\mathcal{S} \times O_\mathcal{S} \to Q_\mathcal{A} \times I_\mathcal{A} \times O_\mathcal{A}$ exists such that for any point in time $t \geq 0$ in every possible sequence, $(\pi_Q(x(t)), \pi_Q(x(t')), \pi_I(in(t)), \pi_O(out(t'))) \in \Delta_\mathcal{A}$.

Definition 7: A *(pair) protocol* is defined by $\mathcal{P} = (S, Q, \vec{q_0}, I, O, Acc, \Delta)$. $S_\mathcal{P} = \{\mathcal{A}_1, \mathcal{A}_2\}$ is the set of participants described by NFIOAs, $Q_\mathcal{P} = Q_1 \times Q_2$ is the set of protocol state values, $\vec{q_0}_\mathcal{P}$ is the initial state value, $I_\mathcal{P} = I_1 \cup I_2$ and $O_\mathcal{P} = O_1 \cup O_2$ are the set of characters, and $Acc_\mathcal{P} = Acc_1 \wedge Acc_2$ is the common acceptance component where all acceptance conditions are combined by logical conjunction. The transition relation is $\Delta_\mathcal{P} \subseteq Q_\mathcal{P} \times Q_\mathcal{P} \times I^\epsilon_\mathcal{P} \times O^\epsilon_\mathcal{P} \times S_\mathcal{P}$. Its elements are determined inductively from the transition relations of the participants. If $\mathcal{A}_k$ is the one participant, $\mathcal{A}_{\bar{k}}$ is the other one.

1. Assuming that $\vec{p} \in Q_\mathcal{P}$ is a reachable state of the protocol and one of the participants $\mathcal{A}_k$ provides a spontaneous transition (p_k, q_k, ϵ, o) with $o \in O^\epsilon_k$, then $(\vec{p}, \vec{p}\left[\frac{q_k}{p_k}\right], \epsilon, o, \mathcal{A}_k) \in \Delta_\mathcal{P}$ is called a *spontaneous transition* of $\mathcal{P}$.
2. Be $(\vec{p}, \vec{p}\left[\frac{q_k}{p_k}\right], i, o, \mathcal{A}_k) \in \Delta_\mathcal{P}$ with $i \in I^\epsilon_k$, $o \in O_k$ (and therefore $o \neq \epsilon$). If the other participant $\mathcal{A}_{\bar{k}}$ provides an induced transition with $(p_{\bar{k}}, q_{\bar{k}}, o, o') \in \Delta_{\bar{k}}$, then $(\vec{p}\left[\frac{q_k}{p_k}\right], \vec{p}\left[\frac{q_k}{p_k}\right]\left[\frac{q_{\bar{k}}}{p_{\bar{k}}}\right], o, o', \mathcal{A}_{\bar{k}}) \in \Delta_\mathcal{P}$ is called an *induced transition* of $\mathcal{P}$.

A protocol has to guarantee that every exchanged character is indeed processed:

Definition 8: A (pair) protocol $\mathcal{P}$ is called *well formed* (e.g. BZ83) if for every transition $(\vec{p}, \vec{p}\left[\frac{q_k}{p_k}\right], i, o, \mathcal{A}_k) \in \Delta_\mathcal{P}$ with $i \in I^\epsilon_k$, $o \neq \epsilon$ there exists an induced transition of $\mathcal{A}_{\bar{k}}$

Additionally, a protocol has to guarantee that the acceptance condition can really be met from each reachable state:

Definition 9: A well formed protocol $\mathcal{P}$ is called *consistent* if for each reachable protocol state value $\vec{p} \in Q_\mathcal{P}$ there exists a (finite) path such that its acceptance condition hold.

In the case of finite automata with the acceptance component of a set of final states $F_{\mathcal{P}} = F_1 \times F_2$, a consistent protocol always provides a finite path leading to a final protocol state.

3.1 Partitioned I/O automata

For describing more complex rules, ordinary I/O automata become awkward because of the so-called “state explosion”. A state capable of representing an arbitrary 32 bit integer value already introduces roughly 4.3 billion different state values. To make complex I/O automata better comprehensible, the transition relation Δ can be partitioned into N disjoint sub-relations $\Delta_l \subseteq \Delta$ with $\Delta = \bigcup \Delta_l$:

Definition 10: A *nondeterministic partitioned finite I/O automaton (NPFIOA)* is defined by $\mathcal{A} = (Q, I, O, q_0, Acc, \{\Delta_l\}_{l \in N})$. Q, I, O, q_0 and Acc are defined as in definition 5.

The sub-relations allow a representation of complex rules by corresponding expressions which become true for each member and only each member of the sub-relations. In case of a deterministic automaton (DPFIOA), each sub-relation defines its own transition function.

One way to find an appropriate partition of a given transition relation is to introduce a “principal state component” together with “structured” I/O characters. As principal state component the first component of the internal state can be chosen. A structured character is actually a string structured according to a given grammar $\mathcal{G}$. Each Δ_l is then determined by the value of a principal start and target state component $\hat{p}$ and $\hat{q}$ and the grammars $\mathcal{G}_i$ and $\mathcal{G}_o$ of its input and output character: $\Delta_l = \Delta_l(\hat{p}, \hat{q}, \mathcal{G}_i, \mathcal{G}_o)$.

Finding an appropriate partition therefore is a bit arbitrary. For example, a confirmation and a rejection can be modeled as two different document types or as two different instantiations of a single document type.

3.2 Example: Tic tac toe described as an interaction

I now describe the tic tac toe game of section 2.1 as a protocol interaction where two players tell each other their moves as is illustrated in Fig. 2.

A class of states is denoted by $[p, b]_c$ where p is the principal state component, b is the board state and c is one of the board related conditions. A class of transitions is also indicated by square brackets '[]'. A single character encoding the value k is denoted as \$k.

For convenience, I introduce two additional conditions relating to the board state: c_4 is true if the k-th position is empty, c_5 is true if the '\$k'-th position is empty.

Definition 11: Two participants 1 and 2 are involved in a *tic tac toe protocol* if they can be characterized by $(Q, I, O, q_0, F, \{\Delta_l\})_{1,2}$ with:

$$\begin{array}{lcl}
Q_{1,2} & = & \{\text{won}, \text{lost}, \text{draw}, \text{mt}, \text{ot}\} \times \{\_, \text{o}, \text{x}\}^9\}, \\
I_{1,2} & = & O_{1,2} = \{1, \ldots, 9\}, \\
q_{0\,1} & = & (\text{mt}, ((\_, \_, \_), (\_, \_, \_), (\_, \_, \_))),
\end{array}$$

$$\begin{array}{lll}
q_{02} & = & (\text{ot}, ((\_,\_,\_), (\_,\_,\_), (\_,\_,\_))), \\
\Delta_1 & = & \{[[\text{mt}, b]_{\neg c_2 \wedge c_4}, [\text{ot}, b \begin{bmatrix} \text{x} \\ \hline - \end{bmatrix}_k]_{\neg c_2}, \epsilon, \$\text{k}], \\
 & & [[\text{ot}, b]_{\neg c_1 \wedge \neg c_3 \wedge c_5}, [\text{mt}, b \begin{bmatrix} \text{o} \\ \hline - \end{bmatrix}_k]_{\neg c_1 \wedge \neg c_3}, \$\text{k}, \epsilon], \\
 & & [[\text{ot}, b]_{c_1}, [\text{won}, b]_{c_1}, \epsilon, \epsilon], [[\text{ot}, b]_{c_3}, [\text{draw}, b]_{c_3}, \epsilon, \epsilon], [[\text{mt}, b]_{c_2}, [\text{lost}, b]_{c_2}, \epsilon, \epsilon] \}, \\
\Delta_2 & = & \{[[\text{mt}, b]_{\neg c_1 \wedge \neg c_3 \wedge c_4}, [\text{ot}, b \begin{bmatrix} \text{o} \\ \hline - \end{bmatrix}_k]_{\neg c_1 \wedge \neg c_3}, \epsilon, \$\text{k}], \\
 & & [[\text{ot}, b]_{\neg c_2 \wedge c_5}, [\text{mt}, b \begin{bmatrix} \text{x} \\ \hline - \end{bmatrix}_k]_{\neg c_2}, \$\text{k}, \epsilon], \\
 & & [[\text{ot}, b]_{c_2}, [\text{won}, b]_{c_2}, \epsilon, \epsilon)], [[\text{mt}, b]_{c_3}, [\text{draw}, b]_{c_3}, \epsilon, \epsilon], [[\text{mt}, b]_{c_1}, [\text{lost}, b]_{c_1}, \epsilon, \epsilon]\} \}, \\
F_1 & = & \{[\text{won}, b]_{c_1}, [\text{lost}, b]_{c_2}, [(\text{draw}, b]_{c_3}\}, \\
F_2 & = & \{[\text{won}, b]_{c_2}, [\text{lost}, b]_{c_1}, [\text{draw}, b]_{c_3}\},
\end{array}$$

3.3 Introducing decisions - the game automaton and its simplification

Decisions determine the actions of a player. Any extension of a NFIOA which leaves the original I/O-behavior invariant and results in a deterministic automaton could therefore be interpreted as introducing decisions.

Definition 12: A corresponding *decision automaton* $\mathcal{D}$ of a NFIOA $\mathcal{A}$ (of a protocol $\mathcal{P}$) is a DFIOA with an input alphabet $I_{\mathcal{D}} = I_{\mathcal{A}} \cup D$ ($I_{\mathcal{A}} \cap D = \{\}$) and a state $Q_{\mathcal{D}} = Q_{\mathcal{A}} \times Q'$ such that its generated behavior restricted to the input character set $I_{\mathcal{A}}$ still prescribes the same behavior as $\mathcal{A}$.

Proposition 1: For each NFIOA a corresponding decision automaton can be constructed.

Proof: A possible construction mechanism is to add an additional NFIOA with the decision alphabet as input and no output to the original NFIOA as an unsynchronized product (see definition below) and synchronize this product afterwards in a way that the original behavior is retained and a deterministic automaton results.

Definition 13: The *unsynchronized product* of two NFIOAs $\mathcal{A}_1$ and $\mathcal{A}_2$ is defined by NFIOA $\mathcal{P}$ with $Q_{\mathcal{P}} = Q_1 \times Q_2$, $I_{\mathcal{P}} = I_1 \cup I_2$, $O_{\mathcal{P}} = O_1 \cup O_2$, $\vec{q}_0 = (q_{01}, q_{02})$, The common acceptance condition $Acc_{\mathcal{P}}$ is the logical conjunction of the individual acceptance conditions, $\Delta_{\mathcal{P}} := \{(\vec{p}, \vec{q}, i, o) |\ \vec{p}$ is a reachable state and $\mathcal{A}_{k \in \{1,2\}}$ provides a transition (p_k, p'_k, i_k, o_k) with $\vec{q} = \vec{p} \begin{bmatrix} p'_k \\ \hline p_k \end{bmatrix}$ and $i = i_k$ and $o = o_k\}$.

Synchronization which leaves the original I/O-behavior invariant can then be achieved by two mechanisms: transition elimination and ϵ-merge. As is illustrated in Fig. 3, transitions of the product automaton which contribute to nondeterministic ambiguities but whose omission don't affect the projected original behavior can be eliminated. In an $\epsilon - merge$ a spontaneous transition without input is merged onto the preceding transition.

Proposition 2: Be $\mathcal{P}$ a consistent protocol with participants $\mathcal{A}_1$ and $\mathcal{A}_2$. Be furthermore

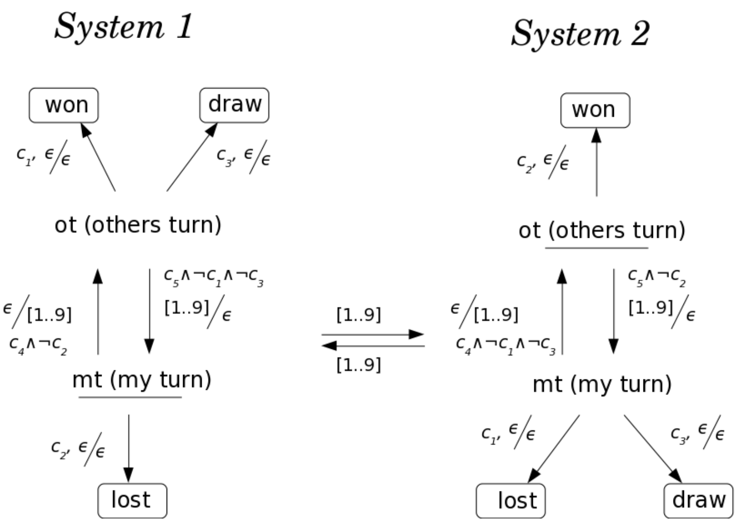

Figure 2: Two systems playing tic tac toe. System 1 makes the initial move.

$\mathcal{D}_1$ and $\mathcal{D}_2$ some corresponding decision automata with their additional input alphabet D_1 and D_2. Then, $\mathcal{D}_1$ and $\mathcal{D}_2$ interacting together as $\mathcal{A}_1$ and $\mathcal{A}_2$ is a deterministic finite automaton with the input alphabet $D_1 \cup D_2$ and. Such an automaton is also called a *game automaton*.

Proof: Because the interaction fulfills the consistency condition, all transitions within the game automaton beside the decision-induced now occur automatically. Thus, with respect to the input alphabet $D_1 \cup D_2$, the interacting decision automata represent a deterministic finite automaton.

As the decision automata interaction occurs automatically, it can actually be eliminated in analogy to the well known ϵ-elimination (e.g. HMU02, section 2.5). I define:

Definition 14: The *decision-closure* of a state q and a decision d of a game automaton $\mathcal{D}$ are all states reachable from $\delta(q, d)$ without any further decision.

Partitioning the set of states into decision closures in the sense of $\epsilon-$ elimination, we can reach a *simplified game automaton*, relating only to these decision closures. For such a simplified game automaton, the following proposition holds:

Proposition 3: Be $\mathcal{A} = (\bigcup D_i, Q, q_0, F, \Delta, N)$ a simplified game automaton with a non-cyclical transition relation $\Delta_{\mathcal{A}}$, a set of finite states $F_{\mathcal{A}}$ as acceptance component and supplemented by the set of players N. And be $f : Q \to V$, $g : \bigcup D_i \to C$ two bijections. Then $\mathcal{A}$ defines a game $\mathcal{G} = ((V, v_0, T, p), N, X, \sim, R, C, u)$ according to definition 1 without its payoff functions u.

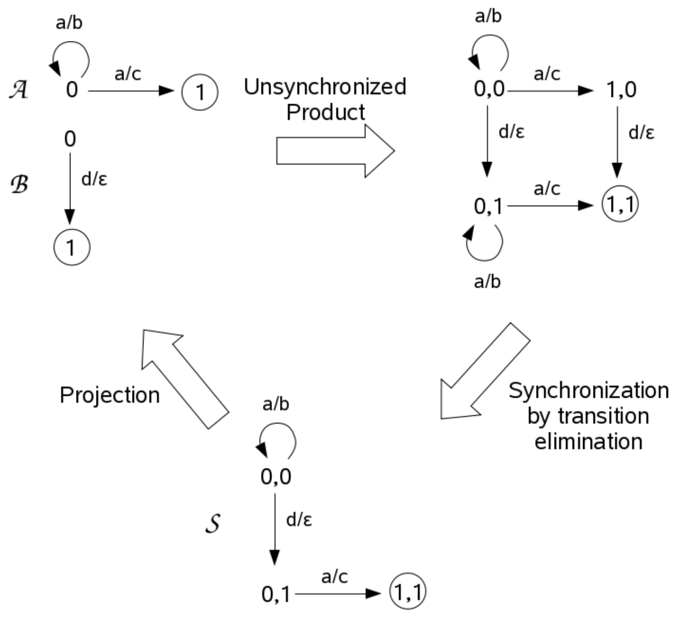

Figure 3: Synchronization by transition elimination. Please note the symmetry breaking effect of transition elimination.

Proof: $V_{\mathcal{G}} = f(Q_{\mathcal{A}})$, $v_0 = f(q_0)$, and $T_{\mathcal{G}} = f(F_{\mathcal{A}})$, $C = g(\bigcup D_i)$. Because of its determinacy, the transition relation $\Delta_{\mathcal{A}}$ fulfills the first condition, that each decision character "selects" an immediate successor node. The second condition that a choice has to be available to every node of an information set of a player i can be used to find the information sets. Thus, Δ_A is transformed onto $R_{\mathcal{G}}$ and the predecessor function p can be extracted from $R_{\mathcal{G}}$. The partition of the set $Q_{\mathcal{A}} \setminus F_{\mathcal{A}}$ into disjoint subsets which can be attributed to each player is possible because of the construction of the simplified game automaton, leading to $X_{\mathcal{G}}$.

In a game of perfect information, it should be possible to show that any relevant internal 'board' states of both players become identical before any further decision is made by either of them.

3.4 Tic tac toe as a (simplified) game automaton

For the tic tac toe game, we get the following decision-closures: '1 has won' if c_1 holds for player 1, '2 has won' if c_2 holds for player 2, 'draw' if c_3 holds for 1. '1's turn' if the only next possible move is a choice of 1 and 2 has not yet won and '2's turn' if the only next possible move is a choice of 2 and 1 has not yet won. Formally:

1 has won := $\{[[\mathrm{ot}_1, b_1]_{c_1}, [\mathrm{ot}_2, b_2]_{\neg c_2}, \$\mathrm{k}_1]\ ,\ [[\mathrm{won}_1, b_1]_{c_1}, [\mathrm{ot}_2, b_2]_{\neg c_2}, \$\mathrm{k}_1]\ ,$

$[[\text{ot}_1, b_1]_{c_1}, [\text{mt}_2, b_2]_{c_1}, \epsilon]\,,\ [[\text{won}_1, b_1]_{c_1}, [\text{mt}_2, b_2]_{c_1}, \epsilon]\,,\ \underline{[[\text{won}, b]_{c_1}, [\text{lost}, b_{c_1}], \epsilon]}\}$
2 has won $:= \{[[\text{ot}_1, b_1]_{\neg c_1}, [\text{ot}_2, b_2]_{c_2}, \$\text{k}_2]\,,\ [[\text{ot}_1, b_1]_{\neg c_1}, [\text{won}_2, b_2]_{c_2}, \$\text{k}_2]\,,$
$[[\text{mt}_1, b_1]_{c_2}, [\text{ot}_2, b_2]_{c_2}, \epsilon]\,,\ [[\text{mt}_1, b_1]_{c_2}, [\text{won}_2, b_2]_{c_2}, \epsilon]\,,$
$[[\text{lost}_1, b_1]_{c_2}, [\text{ot}_2, b_2]_{c_2}, \epsilon]\,,\ \underline{[[\text{lost}_1, b_1]_{c_2}, [\text{won}_2, b_2]_{c_2}, \epsilon]}\}$
draw $:= \{[[\text{ot}_1, b_1]_{c_3}, [\text{ot}_2, b_2]_{\neg c_2}, \$\text{k}_1]\,,\ [[\text{draw}_1, b_1]_{c_3}, [\text{ot}_2, b_2]_{\neg c_2}, \$\text{k}_1]\,,$
$[[\text{ot}_1, b_1]_{c_3}, [\text{mt}_2, b_2]_{c_3}, \epsilon]\,,\ [[\text{draw}_1, b_1]_{c_3}, [\text{mt}_2, b_2]_{c_3}, \epsilon]\,,$
$[[\text{ot}_1, b_1]_{c_3}, [\text{draw}_2, b_2]_{c_3}, \epsilon]\,,\ \underline{[[\text{draw}_1, b_1]_{c_3}, [\text{draw}_2, b_2]_{c_3}, \epsilon]}\}$
1's turn $:= \{[[\text{ot}_1, b_1]_{\neg c_2}, [\text{ot}_2, b_2]_{\neg c_2}, \$\text{k}_2]\,,\ \underline{[[\text{mt}_1, b_1]_{\neg c_2}, [\text{ot}_2, b_2]_{\neg c_2}, \epsilon]}\}$
2's turn $:= \{[[\text{ot}_1, b_1]_{\neg c_1}, [\text{ot}_2, b_2]_{\neg c_1}, \$\text{k}_1]\,,\ \underline{[[\text{ot}_1, b_1]_{\neg c_1}, [\text{mt}_2, b_2]_{\neg c_1}, \epsilon]}\}$

Each decision-closure has an element which is the endpoint of the deterministic chains (the underlined states). It can be seen that for each endpoint element $b_1 = b_2$ holds. That is, before any decision is made by either of the two players, the internal board states always become identical. Thus, referring to these closures, it makes sense to refer to a "single" board. Such a tic tac toe game with a single board is illustrated in Fig. 4 and is formally given by the DPFIA $(Q, D, q_o, \{\Delta_k\}, F)$ with

$Q = \{(p, b) | p \in \{\text{1 has won, 2 has won, 1's turn, 2's turn, draw}\}, b \in \{\_, \text{o}, \text{x}\}^9\},$
$D = \{1, \ldots, 9\}$
$q_0 = (\text{1's turn}, ((\_, \_, \_), (\_, \_, \_), (\_, \_, \_))),$
$\Delta = \{[[\text{2's turn}, b]_{\neg c_2 \wedge c_4}, [\text{1's turn}, b\left[\frac{\text{o}}{\_}\right]]_{\neg c_2}, k_2],$
$[[\text{2's turn}, b]_{\neg c_2 \wedge c_4}, [\text{2 has won}, b\left[\frac{\text{o}}{\_}\right]]_{c_2}, k_2],$
$[[\text{1's turn}, b]_{\neg c_1 \wedge \neg c_3 \wedge c_4}, [\text{2's turn}, b\left[\frac{\text{x}}{\_}\right]]_{\neg c_1 \wedge \neg c_3}, k_1],$
$[[\text{1's turn}, b]_{\neg c_1 \wedge \neg c_3 \wedge c_4}, [\text{1 has won}, b\left[\frac{\text{x}}{\_}\right]]_{c_1}, k_1],$
$[[\text{1's turn}, b]_{\neg c_1 \wedge \neg c_3 \wedge c_4}, [\text{draw}, b\left[\frac{\text{x}}{\_}\right]]_{c_3}, k_1]\}$
$F = \{[\text{1 has won}, b]_{c_1}, [\text{2 has won}, b]_{c_2}, [\text{draw}, b]_{c_3}\}$

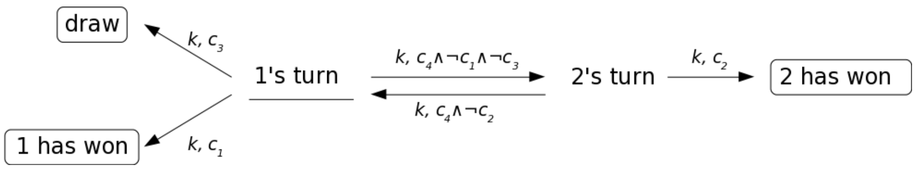

Figure 4: Tic tac toe game with a single board as deterministic decision automaton.

4 Discussion

With this article I illustrate the relation between games and protocols. It is interesting to review the differences of both formalisms which describe very similar aspects of interactions. The fundamental significance of both concepts is emphasized by directly attributing the protocol concept to the very general system concept. However, while the traditional extensive formalism of games focuses on the move of a participating system, protocols focus more on the separation of the systems during the interaction.

According to both approaches, games specify interactions where participants can make decisions not determined by the interactions. From this point of view, we can speak of decisions only if we assume nondeterministic interaction relations. If the interactions determine the actions then there is no room left for any decisions in this sense.

This study also indicates to classify decisions into spontaneous (or inducing) and selection decisions as undetermined transitions of a protocol occur either spontaneously or induced. Hence, the characters of the decision alphabet of the corresponding decision automaton either trigger a spontaneous transition or select an otherwise undetermined induced transition.

In game theory much weight is currently put on the question of strategy or how to decide in a single sort of game. Blanking out everything else, identifying a distinguished strategy depends on additional assumptions like payoff optimization.

In informatics, the focus to describe interactions is actually not so much the optimized behavior in a single interaction, but to assure that the player - or process, as it is called - coordinates all interactions it is involved in at least correctly. Thus, in informatics the correctness of deterministic systems, involved in many different interactions, each of which is nondeterministic by itself, is key. And optimization of single interactions becomes subordinate to performing "good enough" over all.

Here, my hope is that both disciplines could learn from each other: on the one hand, game theory could look more at the player and on its necessity to coordinate different interactions in the sense of "satisficing" (Sim56) for identifying distinguished strategies. On the other hand, informatics could focus more on systems which effectively have some real degrees of freedom to make their decisions and have to figure out real strategies to fill this gap.

References

[Ben02] Johan Van Benthem. Extensive Games as Process Models. *J. of Logic, Lang. and Inf.*, 11(3):289–313, 2002.

[Boc78] Gregor V. Bochmann. Finite State Description of Communication Protocols. *Computer Networks*, 2:361–372, 1978.

[BZ83] Daniel Brand and Pitro Zafiropulo. On Communicating Finite-State Machines. *J. ACM*, 30(2):323–342, 1983.

[GZA05] N. Ghoualmi-Zine and A. Araar. Net-banking system as a game. In *Proceedings of the world academy of science, engineering and technology*, volume 6, pages 26–29, June 2005.

[HMU02] J. E. Hopcroft, R. Motwani, and J. D. Ullmann:. *Einführung in die Automaten-*

theorie, formale Sprachen und Komplexitätstheorie. Addison-Wesley, Pearson Studium, 2002.

[Hol91] Gerard J. Holzmann. *Design and validation of computer protocols*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1991.

[KK07] Jeffrey J. Kline and Mamoru Kaneko. Information Protocols and Extensive Games an Inductive Game Theory. 2007.

[Kuh53] H.W. Kuhn. Extensive games and the problem of information. In H.W. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games, Vol. II*, volume 28 of *Annals of Mathematics Studies*, pages 193–216. Princeton University Press, Princeton, NJ, 1953. Reprint in Kuhn(1997), 46-68.

[SB67] R. A. Scantlebury and K. A. Bartlett. A Protocol for Use in the NPL Data Communications Network. Technical Memorandum, 1967.

[Sel75] R. Selten. Re-examination of the perfectness concept for equilibrium points in extensive games. *International Journal of Game Theory*, pages 25–55, 1975. Reprinted in Kuhn(1997), 317-354.

[Sim56] A. Simon, Herbert. Rational choice and the structure of the environment. *Psychological Review*, 63:129–138, 1956.

[vNM90] Johann von Neumann and Oskar Morgenstern. *Spieltheorie und oekonomisches Verhalten*. University Press, 3 edition, 1990.