A Coalgebraic Framework for Games in Economics

Achim Blumensath
Department of Mathematics
Technical University of Darmstadt

Viktor Winschel
Department of Economics
University of Mannheim

June 6, 2013

Abstract

We present an abstract framework to compose economic games. It is sufficiently general to represent all types of games encountered in economics. Our framework is based on category-theoretical techniques and the notion of a coalgebra. Coalgebras have been successfully used to model the observable behavior of systems with unobservable state space. We introduce the fundamental notion of a *process* as a coalgebra of a certain kind, and we show how to use them to describe stage games, finitely, infinitely and potentially infinitely repeated games with imperfect and incomplete information based on deterministic, non-deterministic or probabilistic processes. Our framework allows us to compose games sequentially, in parallel and hierarchically and it provides a formal account of the aggregate behavior of networks of agents. The models are directly implementable in high level declarative functional programing languages. The abstract mathematics of our approach links economics to the latest developments in mathematical game theory, theoretical computer science and self-referential structures that underly the Lucas critique and many other economic modeling issues.

Contents

1 Introduction

[t]he idea of rational expectations is ... said to embody the idea that economists and the agents they are modeling should be placed on the equal footing: the agents in the model should be able to forecast and profit-maximize and utility-maximize as well as the economist - or should we say the econometrician - who constructed the model.

Thomas Sargent [100, 106]

I am claiming, that in any natural science, there is only as much genuine science as there is mathematics.

Immanuel Kant [62]

We present a new mathematical representation for games. The goal is to raise the abstraction level of the mathematical language that has been used so far. The methods we propose were developed in theoretical computer science in fields such as modeling and verification of processes, semantics of programming languages, logic, and automata theory [94]. A key idea that we want to exploit is that the behavior of systems is composed from the behavior of more basic ones [103, 65]. In our framework we can compose games uniformly over different types of agents, games, and networks.

The motivation for higher levels of abstraction is similar as the one in computer science and quantum physics [3, 25]. A level that is too low precludes to understand and synthesize complex structures from the analysis of substructures in specialized formalisms. We want to take serious the hard learned lessons of computer science for the need of compositionality, types and abstraction and logic and algebraic tools in order to design and understand complex informational systems [3].

We sketch in this introduction and further in the unification section below how our framework relates, integrates and unifies the major subfields of economics such as computational, network, agent based and macroeconomics and econometrics. The common thread in these fields is reflexivity or self-reference that is inherent in the interaction of content and context, or the object and its encoding (semantics and syntax) that arises in computer science when writing code that process its own or other code. In social science reflexivity arises since the modeling takes place in the modeled system and changes it as well [66, 117, 13, 15, 14].

The category-theoretical machinery we use is based on coalgebras [98, 58]. Category theory [77] provides a universal language for mathematics and certain parts of theoretical computer science [1]. Algebras support finite constructions, while coalgebras model the observation of infinite objects. In particular, coalgebras are well suited to model unobservable state-based systems [109], the observation of their behaviors at an interface and it provides a formal semantics to behavioral specifications. This approach has been developed for programing languages that are modeled as abstract unobservable state transition systems. The coalgebraic representation of games was introduced in [74, 5]. The proximity of our framework to those used in the theory of programing languages has the side effect for computational economics that games modeled in our framework can be implemented more or less directly in high level programing languages.

The goal of computational economics as we understand it is not to translate the low level mathematical language of economics by hand and long session in the debugger into even lower computer languages like Matlab, Fortran, C++ or Java but to raise the mathematical language of economics such that high level programing languages like Haskell, SML or Maude and their underlying compilers and interpreters that are based on category theory and coalgebras can do the translation job. In that way the powerful mathematical tools to design languages can be used to design economic theories as well.

In this paper we only take advantage of the very basic notions of category theory namely functors, natural transformations, and coalgebras. Functors are used to represent types of agents and games, while natural transformations translate between functors (i.e., types) for a formal account of composition. Each game will be represented by a coalgebra and gives rise to a corresponding final coalgebra, which can be regarded as a fixed point of the functor. Its domain allows us to define the behavior or the semantics of a system. The same kind of fixed points allows us to model reflexive structures [73, 118, 22, 2] that arise everywhere in social sciences like the infinite regress of "I know that you know that I know...". However, more powerful constructions await their interpretation in economics and we discuss some in the unification section below.

We show how to formulate some standard games in our framework [44, 87, 79, 107]. The novel and abstract representation provides us with the following immediate advantages.

Firstly, we can formally compose simple games into complicated ones, sequentially or in parallel. The types of agents, games and strategies can be deterministic, non-deterministic or probabilistic and can be based on imperfect or incomplete information. Any finite and possibly very irregular game constructed in this way can be repeated infinitely or potentially infinitely. It is worth noting that we have a formal way to set up potentially or actually infinite games. In economics the usual approach is to form a limit on finite games which precludes this choice. The implied behavior (a possibly infinite tree) of such corecursively formulated games can be defined as a coalgebra morphism into the final coalgebra, which contains all behaviors of a certain type (or functor) of agents and games.

Secondly, compositionality allows for a formal account of aggregation of games. We aggregate along the horizontal and the vertical dimension. The horizontal structures are groups of agents combined into a single agent and the vertical structures aggregate over hierarchies of agents in networks. Hierarchies may be spatial or, as in our example, informational networks for imperfect monitoring games. This approach to aggregation or composition contributes to a formal microfoundation of macroeconomics [7].

Thirdly, our approach can be taken as a formal semantics for agent based economics [111] based on ideas in [112] and complements its hypothesis generating simulation approach to emergent properties. This meets the often raised objection to agent based modeling namely that simulation and statistical inference of systematic properties can not replace a formal account of synthesis. We do not rely on simulation, but on the approach of the semantics of programing languages instead that derives by compositional methods the behavior of systems from the behavior of their subparts. This is a major analogy of our approach that we exploit. Our agents can be arbitrary heterogenous and combined with a synthetic approach we unify agent based economics with the

macroeconomic literature of heterogenous agents [49].

Fourthly, our framework generalizes network economics [56] where games in networks can be played over the structure of the network. However, we do not want to formalize such a game in this paper within our framework but it is important that in principle it would be possible to do so. The interaction of the network structure and the games played in the network is the realm of institutional economics and the general field of the reflexivity of the interaction of the context and the content or the object and its encoding. Coalgebras are well prepared to model this characteristic of societies that also underlie the Lucas critique where the model and the modeled interact. We will discuss reflexivity in some more details in the unification section.

Fifthly, the application of methods from computer science and language design suggest a division of labor in economics. Theoretical economics develops the interpreter for the economic language while applied economics uses the language to answer economic questions. Our formulation of games is directly programable in high level declarative functional programing languages [34, 55, 88, 24]. The types or functorial definitions of our framework form the core of a simulation engine or the interpreter of our language while the specification of concrete games is the runnable code as type transformations. Under the Curry-Howard isomorphism [1] types are propositions and computations are proofs. This duality opens the way to proof assistants and logical specification languages for economics. Coalgebras come with appropriate logical systems to define and proof properties [70] of the systems like coinduction for all kind of infinite as opposed to induction for finite structures or modal logic for the specification and verification of complicated real world systems. Not only can we compute the games by the identity of code and model but we can also analyze the models by powerful logical and category theoretical tools.

Last but not least the current development of the information society moves computer sciences into the traditional realms of social science such that both overlap [39]. Computers communicate in networks and allocate decentralized computational resources and also steer human societies with more or less explicit human-computer interfaces. It is therefore of paramount importance that economics and computer science interact at the theoretic level. This means most of all that as much as possible of the informal semantics and methodology of economics formulated in natural languages [36, 26, 18] need to be formalized. But this presupposes that the formal structures are sufficiently expressive or high level in order to be able to accommodate the expressivity of economics formulated in natural languages without eschewing decidability [19, 48].

We base our approach essentially on the deepest commonalities of both sciences namely reflexivity and functorial fixed points. Composability, types and abstraction are needed as urgent in economics as in computer science in order to understand and design the interacting complex information systems of humans, computers and nature.

In the next section we describe our framework. It is based on the building block of processes as coalgebras. We define the operators to compose processes from more simple ones. We then define agents, strategies and games as processes. Final coalgebras are the semantical domains where we can find the behaviors of our systems. For the types of our framework in the current stage final coalgebras contain sets of potentially infinite labelled trees. We show how these trees arise in the semantic domain. We end with the definition of outcomes

and equilibria for games.

In the following section we will present some examples how to formulate games within our framework. These examples are not meant to take advantage of the full power of our framework. We do aim neither to formally proof the representational power of our framework nor its formal relation to the representation of games so far. But we want to suggest by some examples that the present state of game theory can be encoded in our framework. More complex examples and more mathematics is left for the work to come.

We want to mention some of these directions in the section on unification. The unification and synthesis of various strands of literatures in economics is a major strength and goal of our abstract framework. It extends beyond economics and opens the door to powerful mathematical and computer scientific methods that have been developed outside of economics.

2 The coalgebraic framework

Definition 2.1. A *category* $\mathcal{C}$ consists of a class $\mathcal{C}^{\mathrm{obj}}$ of *objects*, a class $\mathcal{C}^{\mathrm{arr}}$ of *arrows,* and a *composition operation* $\circ$ on arrows. Each arrow $f \in \mathcal{C}^{\mathrm{arr}}$ has a *domain* $X \in \mathcal{C}^{\mathrm{obj}}$ and a *codomain* $Y \in \mathcal{C}^{\mathrm{obj}}$. We write $f : X \to Y$ to indicate that f is an arrow with domain X and codomain Y. The composition operation is assumed to satisfy the following two conditions:

1. The composition $f \circ g$ of two arrows is defined if, and only if, the domain of f is equal to the codomain of g.

2. The composition operation is associative, i.e., for all arrows $f : X \to Y$, $g : Y \to Z$, $h : Z \to W$,

$$(h \circ g) \circ f = h \circ (g \circ f)\,.$$

3. For each object X, there is an *identity arrow* $\mathrm{id}_X : X \to X$ such that

$$f \circ \mathrm{id}_X = f \quad \text{and} \quad \mathrm{id}_X \circ g = g\,,$$

for all arrows $f : X \to Y$ and $g : Z \to X$.

The only category we will use in this paper is the category of all sets, where the objects are sets and the arrows are total functions. Other examples of categories contain sets and relations, measurable spaces and measurable functions, or topological spaces and continuous functions.

In order to keep the category-theoretical overhead to a minimum, we will introduce all category-theoretical notions only in the special case of the category of sets. The main concepts we will need are that of a functor and a natural transformation.

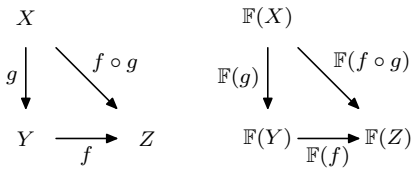
Definition 2.2. A *functor* $\mathbb{F}$ (from the category of sets to itself) is an operation assigning

- to each set X a new set $\mathbb{F}(X)$ and
- to each function $g : X \to Y$ a function $\mathbb{F}(g) : \mathbb{F}(X) \to \mathbb{F}(Y)$

such that

$$\mathbb{F}(\mathrm{id}_X) = \mathrm{id}_{\mathbb{F}(X)} \quad \text{and} \quad \mathbb{F}(f \circ g) = \mathbb{F}(f) \circ \mathbb{F}(g)\,,$$

for all sets X and all functions $f : Y \to Z$ and $g : X \to Y$.



As an example, let us introduce three functors that will be used below.

1. The *identity functor* id maps every set X and every function $f : X \to Y$ to itself.

2. The *finite power-set functor* $\mathcal{P}_{\mathrm{fin}}$ maps every set X to the set $\mathcal{P}_{\mathrm{fin}}(X)$ of its finite subsets, and it maps a function $f : X \to Y$ to the function

$$\mathcal{P}_{\mathrm{fin}}(f) : \mathcal{P}_{\mathrm{fin}}(X) \to \mathcal{P}_{\mathrm{fin}}(Y) : S \mapsto \{\, f(s) \mid s \in S \,\}\,.$$

3. The *finite probability functor* $\mathbb{D}_{\mathrm{fin}}$ maps a set X to the set of all finite probability distributions on X, i.e., all maps $d : X \to [0,1]$ such that only finitely many elements of X are mapped to non-zero values. For a function $f : X \to Y$, it returns the function

$$\mathbb{D}_{\mathrm{fin}}(f) : \mathbb{D}_{\mathrm{fin}}(X) \to \mathbb{D}_{\mathrm{fin}}(Y) : d \mapsto d_f\,,$$

where

$$d_f(y) := \sum_{x \in f^{-1}(y)} d(x)\,.$$

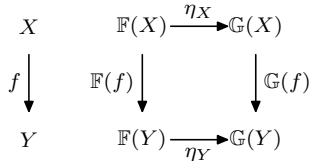
Besides the notion of a functor, we also need those of a natural transformation and a distributive law.

Definition 2.3. (a) A *natural transformation* $\eta : \mathbb{F} \Rightarrow \mathbb{G}$ from a functor $\mathbb{F}$ to a functor $\mathbb{G}$ is a family $\eta = (\eta_X)_X$ of functions

$$\eta_X : \mathbb{F}(X) \to \mathbb{G}(X)\,,$$

indexed by sets X, satisfying

$$\eta_Y \circ \mathbb{F}(f) = \mathbb{G}(f) \circ \eta_X\,, \quad \text{for every function } f : X \to Y\,.$$



(b) A *distributive law* between two functors $\mathbb{F}$ and $\mathbb{G}$ is a natural transformation $\eta : \mathbb{F} \circ \mathbb{G} \Rightarrow \mathbb{G} \circ \mathbb{F}$.

Examples of natural transformations will appear in Section 2.2 below.

2.1 Processes

Before introducing games, let us define the simpler notion of a process, which corresponds to a game with a single player. Processes will provide the technical machinery our framework is based on.

A process is a state-based system transforming an input sequence into an output sequence. In each step it receives an input value and, depending on its current state, it produces an output value and changes its state. Alternatively, a process can decide to terminate. Formally, a *process* is given by

- a set S of *states,*
- a set I of *inputs,*
- a set O of *outputs,*
- a set R of *results,* and
- a function $\pi : S \times I \to \mathbb{C}(R + S \times O)$, for some functor $\mathbb{C}$.

The function π describes one step of the process. When in state $s \in S$ and given the input $i \in I$, the process chooses a possible continuation that consists in either terminating with a result $r \in R$, or in continuing in a state $s' \in S$ and producing an output value $c \in O$.

In the above definition, the *choice functor* $\mathbb{C}$ determines which kind of process we are dealing with. Important examples for choice functors are the following ones.

1. The *deterministic choice functor* $\mathbb{C}_{\text{det}} = \text{id}$ is the identity functor. It can be used if the input uniquely determines what happens next.

2. The *non-deterministic choice functor* $\mathbb{C}_{\text{ndet}} = \mathcal{P}_{\text{fin}}$ is the finite power-set functor. It can be used if, for a given input, there might by several possible continuations of the process.

3. The *probabilistic choice functor* $\mathbb{C}_{\text{prob}} = \mathbb{D}_{\text{fin}}$ is the finite probability functor. It can be used if the continuation of the process is random.

To apply the category-theoretical machinery it will be convenient to write the function π in the form

$$\pi : S \to (\mathbb{C}(R + S \times O))^I.$$

In category-theoretical terms, such functions can be seen as so-called coalgebras.

Definition 2.4. Let $\mathbb{F}$ be a functor. An $\mathbb{F}$-*coalgebra* is a function $h : X \to \mathbb{F}(X)$, for some set X.

Hence, a process π becomes a Π_0-coalgebra $\pi : S \to \Pi_0(S)$, where Π_0 is the *process* functor

$$\Pi_0(X) := \mathbb{C}(R + X \times O)^I.$$

We denote by

$$\Pi(S; I, O, R) := \Pi_0(S)^S$$

the set of all processes with states S, inputs I, outputs O, and results R.

2.2 Transformations of choice functors

In this section we present several natural transformations between choice functors that will be needed in the next section.

1. For two choice functors $\mathbb{C}_1$ and $\mathbb{C}_2$, we define a natural transformation

$$\mu_{1,2} : \mathbb{C}_1 \circ \mathbb{C}_2 \Rightarrow \mathbb{C}_{1,2}$$

   that combines a choice of $\mathbb{C}_1$ followed by a choice of $\mathbb{C}_2$ into a single choice with respect to a combined functor $\mathbb{C}_{1,2}$.

2. For a choice functor $\mathbb{C}$ and fixed sets A, B, we define a distributive law

$$\delta : A + B \times \mathbb{C}(X) \Rightarrow \mathbb{C}(A + B \times X)\,.$$

3. For two choice functors $\mathbb{C}_1$ and $\mathbb{C}_2$, we define a natural transformation

$$\lambda_{1,2} : \mathbb{C}_1(X) \times \mathbb{C}_2(Y) \Rightarrow \mathbb{C}_{1,2}(X \times Y)\,.$$

The definitions of all three natural transformations are the ones you would expect from looking at the respective types. We encourage the reader to skip the formal definitions below, which are only included for sake of completeness.

(1.) For $\mathbb{C}_2 = \mathbb{C}_{\text{det}}$, we can use $\mathbb{C}_{1,2} := \mathbb{C}_1$ and the identity function

$$\mu_{1,\text{det}} : \mathbb{C}_1(X) \to \mathbb{C}_1(X)\,.$$

Analogously, we can define $\mu_{1,2}$ for $\mathbb{C}_1 = \mathbb{C}_{\text{det}}$. For $\mathbb{C}_1 = \mathbb{C}_2 = \mathbb{C}_{\text{ndet}}$, we use $\mathbb{C}_{1,2} := \mathbb{C}_{\text{ndet}}$ and the functions

$$\mu_{\text{ndet,ndet}} : \mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(X)) \to \mathcal{P}_{\text{fin}}(X) : U \mapsto \textstyle\bigcup_{Z \in U} Z$$

mapping a set $U \subseteq \mathcal{P}_{\text{fin}}(X)$ to its union. For $\mathbb{C}_1 = \mathbb{C}_2 = \mathbb{C}_{\text{prob}}$, we use $\mathbb{C}_{1,2} := \mathbb{C}_{\text{prob}}$ and the functions

$$\mu_{\text{prob,prob}} : \mathbb{D}_{\text{fin}}(\mathbb{D}_{\text{fin}}(X)) \to \mathbb{D}_{\text{fin}}(X)$$

mapping a distribution d over $\mathbb{D}_{\text{fin}}(X)$ to the distribution

$$x \mapsto \sum_{d' \in \mathbb{D}(X)} d(d') \cdot d'(x)\,.$$

The case where one of $\mathbb{C}_1$ and $\mathbb{C}_2$ equals $\mathbb{C}_{\text{ndet}}$ and the other one equals $\mathbb{C}_{\text{prob}}$ is more involved. We omit the definitions.

(2.) We define δ as follows. If $\mathbb{C} = \mathbb{C}_{\text{det}}$, we can use the identity map

$$\delta_{\text{det}} : A + B \times X \to A + B \times X\,.$$

If $\mathbb{C} = \mathbb{C}_{\text{ndet}}$, we use the map

$$\delta_{\text{ndet}} : A + B \times \mathcal{P}_{\text{fin}}(X) \to \mathcal{P}_{\text{fin}}(A + B \times X)\,,$$

defined by

$$\delta_{\mathrm{ndet}}(x) := \begin{cases} \{x\} & \text{for } x \in A\,, \\ \{\,(b,u) \mid u \in U\,\} & \text{for } x = (b,U) \in B \times \mathcal{P}_{\mathrm{fin}}(X)\,. \end{cases}$$

If $\mathbb{C} = \mathbb{C}_{\mathrm{prob}}$, we use the map

$$\delta_{\mathrm{prob}} : A + B \times \mathbb{D}_{\mathrm{fin}}(X) \to \mathbb{D}_{\mathrm{fin}}(A + B \times X)\,,$$

defined by

$$\delta_{\mathrm{prob}}(x) := \begin{cases} d_x & \text{for } x \in A\,, \\ d_{b,e} & \text{for } x = (b,e) \in B \times \mathbb{D}_{\mathrm{fin}}(X)\,, \end{cases}$$

where

$$d_x(y) := \begin{cases} 1 & \text{for } y = x\,, \\ 0 & \text{otherwise}\,. \end{cases} \quad \text{and} \quad d_{b,e}(y) := \begin{cases} e(c) & \text{for } y = (b,c)\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

(3.) For $\mathbb{C}_1 = \mathbb{C}_{\mathrm{det}}$, we can use for

$$\lambda_{\mathrm{det,det}} : X \times \mathbb{C}_2(Y) \Rightarrow \mathbb{C}_2(X \times Y)$$

the distributive law δ from (2.) (setting $A := \emptyset$ and $B := X$). The case where $\mathbb{C}_2 = \mathbb{C}_{\mathrm{det}}$ is handled symmetrically. For $\mathbb{C}_1 = \mathbb{C}_2 = \mathbb{C}_{\mathrm{ndet}}$, we define

$$\lambda_{\mathrm{ndet,ndet}} : \mathcal{P}_{\mathrm{fin}}(X) \times \mathcal{P}_{\mathrm{fin}}(Y) \to \mathcal{P}_{\mathrm{fin}}(X \times Y) : (U,V) \mapsto U \times V\,.$$

For $\mathbb{C}_1 = \mathbb{C}_2 = \mathbb{C}_{\mathrm{prob}}$, we define

$$\lambda_{\mathrm{prob,prob}} : \mathbb{D}_{\mathrm{fin}}(X) \times \mathbb{D}_{\mathrm{fin}}(Y) \to \mathbb{D}_{\mathrm{fin}}(X \times Y) : (d,d') \mapsto e_{d,d'}$$

where

$$e_{d,d'}(x,y) := d(x) \cdot d'(y)\,.$$

Again, we omit the cases mixing $\mathbb{C}_{\mathrm{ndet}}$ and $\mathbb{C}_{\mathrm{prob}}$.

2.3 Operations on processes

Before introducing games, let us present several operations to construct processes from simpler ones. We start with sums and products of processes.

(a) The *sum* of two processes is a processes where, depending on the state, either the first process takes a step, or the second one does. We only support the case where both processes use the same choice functor. Formally, the sum $+$ is the operation

$$+ : \Pi(S_0; I, O_0, R_0) + \Pi(S_1; I, O_1, R_1) \to \Pi(S_0 + S_1; I, O_0 + O_1, R_0 + R_1)$$

defined by

$$(\pi_0 + \pi_1)(s) := \begin{cases} \pi_0(s) & \text{if } s \in S_0\,, \\ \pi_1(s) & \text{if } s \in S_1\,. \end{cases}$$

(b) The *product* $\pi_1 \times \pi_2$ of two processes is a process where both components take steps simultaneously. We support the case where π_1 and π_2 use different choice functors. Suppose that π_1 uses $\mathbb{C}_1$, while π_2 uses $\mathbb{C}_2$. Formally, the product $\times$ is the operation

$$\begin{aligned}\times : \Pi(S_0; I_0, O_0, R_0) + \Pi(S_1; I_1, O_1, R_1) \to \\ \Pi(S_0 \times S_1;\ I_0 \times I_1,\ O_0 \times O_1,\ R_0 \times R_1 + R_0 + R_1)\end{aligned}$$

defined by

$$(\pi_0 \times \pi_1)(s_0, s_1)(i_0, i_1) := (\mathbb{C}_{1,2}(f) \circ \lambda_{1,2})(\pi_0(s_0)(i_0), \pi_1(s_1)(i_1))\,,$$

where λ is the natural transformation from Section 2.2 and

$$\begin{aligned}f : (R_0 + S_0 \times O_0) \times (R_1 + S_1 \times O_1) \to \\ (R_0 \times R_1 + R_0 + R_1 + S_0 \times S_1 \times O_0 \times O_1)\end{aligned}$$

is the function

$$f(x_0, x_1) := \begin{cases} (x_0, x_1) & \text{if } x_0 \in R_0 \text{ and } x_1 \in R_1\,, \\ x_0 & \text{if } x_0 \in R_0 \text{ and } x_1 \notin R_1\,, \\ x_1 & \text{if } x_0 \notin R_0 \text{ and } x_1 \in R_1\,, \\ (s_0, s_1, c_0, c_1) & \text{if } x_0 = (s_0, c_0) \text{ and } x_1 = (s_1, c_1)\,, \end{cases}$$

(c) We also introduce two operations to modify the inputs and outputs. Given a process π and a function f, we define new processes $\pi \triangleright f$ and $f \triangleright \pi$ as follows.

For a function $f : S \times O \to S \times O'$ and a process $\pi \in \Pi(S; I, O, R)$, the process $\pi \triangleright f$ applies, after each step, the function f to the returned state-output pair. Formally, we define $\pi \triangleright f \in \Pi(S; I, O', R)$ by

$$(\pi \triangleright f)(s)(i) := \mathbb{C}(\mathrm{id} + f)\big(\pi(s)(i)\big)\,.$$

For a function $f : I' \to I$ and a process $\pi \in \Pi(S; I, O, R)$, the process $f \triangleright \pi$ applies, before each step, the function f to the given input value. Formally, we define $f \triangleright \pi \in \Pi(S; I', O, R)$ by

$$(f \triangleright \pi)(s)(i) := \pi(s)(f(i))\,.$$

(d) Finally, we introduce two more complicated operation on processes. The *feedback operation* takes a process π and feeds back its output as additional input. That is, given a process $\pi \in \Pi(S; I \times O, O, R)$ we construct a new process $\pi^{\circlearrowleft} \in \Pi(S \times O; I, O, R)$ which, at each step, calls the process π with its current input value and the output of the previous turn. We define

$$\pi^{\circlearrowleft}(s, c)(i) := \mathbb{C}(\mathrm{id}_R + f)(\pi(s)(i, c))\,,$$

where

$$f : S \times O \to (S \times O) \times O : (s, c) \mapsto ((s, c), c)\,.$$

(e) The *cascading operation* takes two processes π and ϱ, runs them in parallel, and uses the outputs of the first process as inputs of the second one. We

support the case where π and ϱ use different choice functors. Suppose that π uses $\mathbb{C}_1$, while ϱ uses $\mathbb{C}_2$. Given $\pi \in \Pi(S; I, M, P)$ and $\varrho \in \Pi(T; M, O, R)$, we define $\pi \triangleright \varrho \in \Pi(S \times T; I, O, P + R)$ as follows. Let

$$\varrho' : T \times M \to \mathbb{C}_2(R + T \times O) : (t, m) \mapsto \varrho(t)(m)\,,$$
$$\pi' : S \times T \times I \to \mathbb{C}_1(P + S \times T \times M) : (s, t, i) \mapsto \mathbb{C}_1(\mathrm{id}_P + f_t)\big(\pi(s)(i)\big)\,,$$

where

$$f_t : S \times M \to S \times T \times M : (s, m) \mapsto (s, t, m)\,.$$

We set

$$(\pi \triangleright \varrho)(s, t)(i) := \big(\mathbb{C}_{1,2}(g) \circ \mu \circ \delta \circ \mathbb{C}_1(\mathrm{id}_P + \mathrm{id}_S \times \varrho')\big)\big(\pi'(s, t, i)\big)\,,$$

where μ and δ are the natural transformations from Section 2.2 and

$$g : P + S \times (R + T \times O) \to P + R + S \times T \times O$$

is the function

$$g(x) := \begin{cases} x & \text{if } x \in P\,, \\ r & \text{if } x = (s, r) \in S \times R\,, \\ (s, t, c) & \text{if } x = (s, (t, c)) \in S \times T \times O\,. \end{cases}$$

2.4 Games

We consider games between several players that can consist of finitely many or infinitely many rounds. The game starts is a certain state and, in each round, every player chooses an action to perform. These actions determine the state the game enters next. To determine the outcome of a game, we assume that it produces an output value each turn and that, at the end of the game, it returns some result. Together, the produced sequence of output values and the final result will determine the outcome. Formally, a game is therefore given by

- a set N of *players,*
- for each player $p \in N$, a set A_p of *actions* for player p,
- a set S of *states* of the game,
- a set R of *results,*
- a set O of *output values,* and
- a function

$$\gamma : S \times \prod_{p \in N} A_p \to \mathbb{C}(R + S \times O)\,.$$

Thus, a game is a process where the input has the special form $\prod_{p \in N} A_p$. In particular, a game γ is a Γ-coalgebra

$$\gamma : S \to \Gamma(S)\,,$$

where Γ is the *game* functor

$$\Gamma(S) := \mathbb{C}(R + S \times O)^{\prod_{p \in N} A_p}\,.$$

Example 2.5. To formalise the Prisoner's Dilemma in our framework we use two players $N := \{1, 2\}$, each with two actions $A_p := \{c, d\}$ ('confess' and 'deny'). The game needs only one state $S := \{*\}$, no outputs $O := \emptyset$, and results $R := \mathbb{R} \times \mathbb{R}$. The deterministic game function $\gamma : S \to R^{A_1 \times A_2}$ is defined by

$$\gamma(*)(a_1, a_2) := \begin{cases} (1, 1) & \text{if } (a_1, a_2) = (c, c)\,, \\ (2, -1) & \text{if } (a_1, a_2) = (d, c)\,, \\ (-1, 2) & \text{if } (a_1, a_2) = (c, d)\,, \\ (0, 0) & \text{if } (a_1, a_2) = (d, d)\,. \end{cases}$$

Example 2.6. Let us also formalise the Repeated Prisoner's Dilemma. Again, there are two players $N := \{1, 2\}$ with two actions $A_p := \{c, d\}$ each. We still have only one state $S := \{*\}$, but now use outputs $O := \mathbb{R} \times \mathbb{R}$ and no results $R := \emptyset$. The deterministic game function $\gamma : S \to (S \times O)^{A_1 \times A_2}$ is defined by

$$\gamma(*)(a_1, a_2) := \begin{cases} (*, (1, 1)) & \text{if } (a_1, a_2) = (c, c)\,, \\ (*, (2, -1)) & \text{if } (a_1, a_2) = (d, c)\,, \\ (*, (-1, 2)) & \text{if } (a_1, a_2) = (c, d)\,, \\ (*, (0, 0)) & \text{if } (a_1, a_2) = (d, d)\,. \end{cases}$$

2.5 Players and strategies

Let $\gamma : S \to \Gamma(S)$ be a game. A *strategy* for a player $p \in N$ is a function telling him which action to choose in a given turn of the game. The player has access to his current observations and his knowledge of the play so far. Thus, formally a strategy is a function

$$\sigma : E_p \times B_p \to \mathbb{C}(E_p \times A_p)\,,$$

where E_p is the *epistemic state* of player p and B_p is the set of possible *observations*. Again, we write σ as a coalgebra

$$\sigma : E_p \to \mathbb{C}(E_p \times A_p)^{B_p}\,,$$

that is, a process with inputs B_p, outputs A_p, and results $R = \emptyset$.

The observations of a player depend on the current input, the output of the previous turn, and the actions of all players during the previous turn. To specify what exactly player p can observe, we use a function

$$\beta_p : O \times \prod_{p \in N} A_p \to B_p\,,$$

which we assume is part of the description of the game.

Example 2.7. Suppose we are playing the Repeated Prisoner's Dilemma. A probabilistic strategy for player 1 would be to copy the previous action of the other player with probability 2/3, and to choose the other action with probability 1/3. We use only one state $E_1 := \{*\}$ and the observations $B_1 := \{c, d\}$ are the previous action of player 2.

$$\sigma_1 : \{*\} \to \mathbb{D}_{\text{fin}}(\{*\} \times \{c, d\})^{\{c,d\}} : * \mapsto d$$

where

$$d(x)(*, y) := \begin{cases} 2/3 & \text{if } x = y\,, \\ 1/3 & \text{if } x \neq y\,. \end{cases}$$

If, in a game γ, we fix strategies $(\sigma_p)_{p \in N_0}$ for a subset $N_0 \subseteq N$ of the players, we obtain a new game with players $N \setminus N_0$. We denote this game by $\gamma[\sigma_p]_{p \in N_0}$. The formal definition is as follows. For players $p \in N \setminus N_0$ where no strategy is provided, we introduce a non-deterministic dummy strategy that, independently of the input, always tells the player to play some action from A_p without restricting his choice. This strategy uses only one state. Its formal definition is

$$\sigma_p : 1 \to \mathcal{P}_{\text{fin}}(1 \times A_p)^{B_p} : i \mapsto A_p\,.$$

With the help of these dummy strategies, we can define the desired game as

$$\gamma[\sigma_p]_{p \in N_0} := \Big[\Big[f \triangleright \prod_{p \in N} \sigma_p\Big]^{\circlearrowleft} \triangleright \gamma\Big]^{\circlearrowleft},$$

where the function

$$f : O \times \prod_{p \in N} A_p \to \prod_{p \in N} B_p : (c, \bar{a}) \mapsto (\beta_p(c, \bar{a}))_{p \in N}\,,$$

computes the observations of each player.

2.6 Game trees

Given a game γ and strategies σ_p for each player, we would like to compute the result of the game if each player follows her strategy. Besides the techniques from the previous section, we need one more definition: that of a *game tree*. Informally, a game tree is a tree containing all possible sequences of events allowed in the game. The formal definition is based on the notion of a *final* coalgebra.

Definition 2.8. Let $\mathbb{F}$ be a functor. An $\mathbb{F}$-coalgebra $\omega : \Omega \to \mathbb{F}(\Omega)$ is *final* if, for every $\mathbb{F}$-coalgebra $h : X \to \mathbb{F}(X)$, there exists a unique morphism $\varphi : X \to \Omega$ such that the diagram

$$\begin{array}{ccc} X & \xrightarrow{\varphi} & \Omega \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle \omega} \\ \mathbb{F}(X) & \xrightarrow[\mathbb{F}(\varphi)]{} & \mathbb{F}(\Omega) \end{array}$$

commutes.

For the process functors

$$\Pi_0(X) = \mathbb{C}(R + X \times O)^I$$

there exist final Π_0-coalgebras $\omega : \Omega \to \Pi_0(\Omega)$, provided that the choice functor $\mathbb{C}$ is sufficiently well-behaved. In particular, this is the case for the three choice functors $\mathbb{C}_{\mathrm{det}}$, $\mathbb{C}_{\mathrm{ndet}}$, and $\mathbb{C}_{\mathrm{prob}}$.

Let us describe the final Π_0-coalgebras for the choice functors $\mathbb{C}$ introduced above. The elements of these final coalgebras are *trees,* which are directed acyclic graphs such that there exists one vertex, the *root* of the tree, with the property that every other vertex can be reached by a unique path from the root. A tree is (A, B, C)-*labelled* if it has of more than one vertex and

- the root is unlabelled,
- every other inner vertex is labelled by an element of B,
- every leaf is labelled by an element of $A \cup B$,
- every edge is labelled by an element of C.

If there is an edge with label c from a vertex x to a vertex y, we call y the c-*successor* of x.

(a) We start with the functor

$$\Pi_0(X) := (A + X \times B)^C$$

for $\mathbb{C} = \mathbb{C}_{\mathrm{det}}$. In this case the final Π_0-coalgebra $\omega : \Omega \to \Pi_0(\Omega)$ takes the following form. The set Ω consists of all (A, B, C)-labelled trees that are *deterministic,* that is, such that every leaf has a label in A and every inner vertex has exactly one c-successor, for each $c \in C$. The function ω is defined as follows. Given a tree T and a value $c \in C$, we distinguish two cases depending on the label of the c-successor x of the root. If x is labelled by an element $a \in A$, we set $\omega(T)(c) := a$. If x is labelled by an element $b \in B$, we set $\omega(T)(c) := \langle T', b\rangle$ where T' is the subtree of T rooted at x.

To see that this is indeed the final Π_0-coalgebra, consider an arbitrary Π_0-coalgebra $\pi : S \to \Pi_0(S)$. The required unique function $\varphi : S \to \Omega$ is given by

$$\varphi(s) := T_s\,, \quad \text{for } s \in S\,,$$

where the tree T_s is defined as follows.

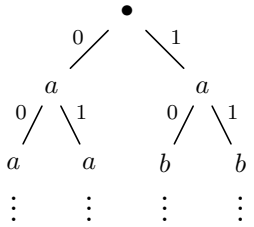
We first construct a graph $\langle V, E\rangle$ with set of vertices $V := A + S \times B$ and the following edges. For every $\langle s, b\rangle \in S \times B$, there is a c-labelled edge from $\langle s, b\rangle$ to $\pi(s)(c)$. The elements of A have no outgoing edges. The vertex labelling is the natural one: a vertex $a \in A$ gets the label a and a vertex $\langle s, b\rangle$ gets the label $b \in B$.

The tree T_s is now obtained from the unravelling of this graph starting at a vertex $\langle s, b\rangle$, for an arbitrary $b \in B$, by forgetting the label b of the root. Formally, the *unravelling* of a graph $\langle V, E\rangle$ starting at a vertex s is defined as the tree consisting of all finite paths through the graph that start at s. There is an edge between two such paths if the second one is obtained from the first one by appending a single edge. This edge also determines the label of edge label. The vertex labelling of the tree is obtained by labelling each path with the label of its end-vertex.

Example 2.9. Let $\pi \in \Pi(\{s_a, s_b\}; \{0,1\}, \{a,b\}, \emptyset)$ be the deterministic process defined by

$$\pi(s_x)(y) := \begin{cases} (s_a, x) & \text{if } y = 0\,, \\ (s_b, x) & \text{if } y = 1\,. \end{cases}$$

The (top of the) tree $\varphi(s_a)$ has the following form:



To see that the function φ defined in this way has the required property we need to check that

$$\Pi_0(\varphi) \circ \pi = \omega \circ \varphi\,.$$

For $s \in S$ and $c \in C$, suppose that

$$\pi(s)(c) = \langle s', b\rangle \in S \times B\,.$$

Let $T := \varphi(s)$ and $T' := \varphi(s')$. Note that T' is equal to the subtree of T rooted at the c-successor of the root and that this c-successor is labelled by b. Hence,

$$(\Pi_0(\varphi) \circ \pi)(s)(c) = \Pi_0(\varphi)(\langle s', b\rangle) = \langle T', b\rangle = \omega(T)(c) = (\omega \circ \varphi)(s)(c)\,.$$

In the case where $\pi(s)(c) = a \in A$ we argue similarly.

(b) Consider the functor

$$\Pi_0(X) := \mathcal{P}_{\text{fin}}(A + X \times B)^C$$

for non-deterministic games. In this case the final Π_0-coalgebra $\omega : \Omega \to \Pi_0(\Omega)$ takes the following form. The set Ω consists of all (A, B, C)-labelled trees where each vertex has only finitely many c-successors, for every $c \in C$. The function ω is defined as follows. Given a tree T and a value $c \in C$, let S be the set of all c-successors of the root of T. Then $\omega(T)(c)$ returns the set

$$\{\, a \in A \mid \text{some } x \in S \text{ has label } a \,\} \cup \{\, \langle T_x, b\rangle \mid x \in S \text{ has label } b \in B \,\}\,,$$

where T_x is the subtree of T rooted at x.

Given an arbitrary Π_0-coalgebra $\pi : S \to \Pi_0(S)$, the required unique function $\varphi : S \to \Omega$ is defined similarly as in (a). We set

$$\varphi(s) := T_s\,, \quad \text{for } s \in S\,,$$

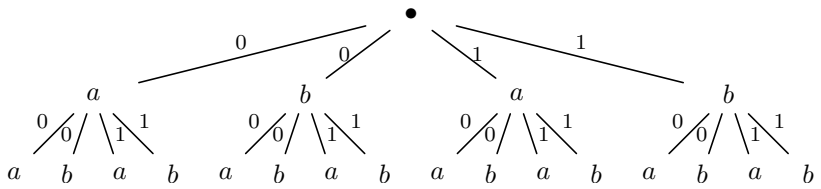
where T_s is the unravelling of the following graph $\langle V, E\rangle$. Again the set of vertices is $V := A + S \times B$ and the vertex labelling is the natural one. For each $\langle s, b\rangle \in S \times B$, there there is a c-labelled edge from $\langle s, b\rangle$ to x, for every $x \in \pi(s)(c)$.

As above, a straightforward calculation shows that the function φ defined in this way has the required properties.

Example 2.10. Let $\pi \in \Pi(\{*\}; \{0,1\}, \{a,b\}, \emptyset)$ be the non-deterministic process defined by

$$\pi(*)(x) := \{(*, a), (*, b)\}\,.$$

The tree $\varphi(*)$ has the following form:



(c) Finally, consider the functor

$$\Pi_0(X) := \mathbb{D}_{\text{fin}}(A + X \times B)^C$$

for probabilistic games. In this case the final Π_0-coalgebra $\omega : \Omega \to \Pi_0(\Omega)$ takes the following form. The set Ω consists of all $(A, B, C \times [0,1])$-labelled trees where, for every $c \in C$ and every vertex v,

- v has only finitely many outgoing edges labelled $\langle c, p\rangle$, for some $p \in [0,1]$,
- the sum of all values p such that there is an outgoing edge with label $\langle c, p\rangle$ equals 1, and
- v does not have two outgoing edges with labels $\langle c, p\rangle$ and $\langle c, p'\rangle$ where $p, p' \in [0,1]$ and such that the subtrees rooted at the corresponding successors are isomorphic.

The function ω is defined as follows. Given a tree T, a value $c \in C$, $a \in A$, and $\langle T', b\rangle \in \Omega \times B$, we set

$$\omega(T)(c)(a) := p$$

if the root of T has an outgoing edge with label $\langle c, p\rangle$ that leads to a leaf with label a, and we set

$$\omega(T)(c)(\langle T', b\rangle) := p$$

if the root of T has an outgoing edge with label $\langle c, p\rangle$ that leads to an inner vertex x with label b such that the subtree of T rooted at x is equal to T'. In all other cases, we set

$$\omega(T)(c)(x) := 0\,.$$

Given an arbitrary Π_0-coalgebra $\pi : S \to \Pi_0(S)$, the required unique function $\varphi : S \to \Omega$ is defined similarly as in (a). We set

$$\varphi(s) := T_s\,, \quad \text{for } s \in S\,,$$

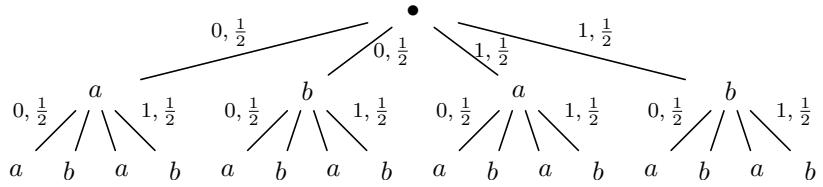
where T_s is the unravelling of the following graph $\langle V, E\rangle$. Again the set of vertices is $V := A + S \times B$ and the vertex labelling is the natural one. For each $\langle s, b\rangle \in S \times B$ and every $x \in A + S \times B$, there is a $\langle c, \pi(s)(c)(x)\rangle$-labelled edge from $\langle s, b\rangle$ to x.

As above, a straightforward calculation shows that the function φ defined in this way has the required properties.

Example 2.11. Let $\pi \in \Pi(\{*\}; \{0,1\}, \{a,b\}, \emptyset)$ be the probabilistic process defined by

$$\pi(*)(x)(s,c) := 1/2\,.$$

The tree $\varphi(*)$ has the following form:



(Due to space considerations we have omitted some edge labels.)

We have seen that the final coalgebras consist of trees describing all possible sequences in the game. Given a game $\gamma : S \to \Gamma(S)$ and the unique morphism $\varphi : S \to \Omega$ into the final Γ-coalgebra, we call the tree $\varphi(s)$ the *game tree* of γ when starting in state $s \in S$.

2.7 The outcome of a game

After these preparations we can determine the outcome of a game. Given a game γ and strategies σ_p for each player, we can compute a game $\gamma[\sigma_p]_p$ without players and determine its game tree T. Hence, it remains to define how to read off the outcome from a game tree.

Let $\gamma : S \to \Gamma(S)$ be a game without players and let $\omega : \Omega \to \Gamma(\Omega)$ be the final Γ-coalgebra. To define the outcome γ we specify a set U of *outcomes* and two functions $\varrho : \Omega \to U$ and $\tau : \Gamma(U) \to U$ such that

$$\begin{array}{ccc} \Omega & \xrightarrow{\omega} & \Gamma(\Omega) \\ {\scriptstyle\varrho}\downarrow & & \downarrow{\scriptstyle\Gamma(\varrho)} \\ U & \xleftarrow[\tau]{} & \Gamma(U) \end{array}$$

Intuitively, ϱ maps a game tree to its outcome, while τ computes the outcome of a game from the outcomes of its subgames. Hence, τ performs a local computation, while ϱ is needed to compute the limit of an infinite sequence of turns. Ideally, the function τ uniquely determines ϱ. This is the case, for instance, for discounted pay-off games, where the value of a game mostly depends on an initial segment of the game tree.

Example 2.12. Consider a deterministic two player game with $R = \mathbb{R} \times \mathbb{R}$ and $O = \mathbb{R} \times \mathbb{R}$. Fixing deterministic strategies for both players, we obtain a deterministic zero-player game, whose game tree is either an infinite sequence over O or a finite sequence where the last element is from R and the remaining ones are from O.

Choosing a discount factor $\lambda \in (0,1)$, we can define the outcome by the functions

$$\tau : R + U \times O \to U \quad \text{and} \quad \varrho : \Omega \to U\,,$$

where $U := \mathbb{R} \times \mathbb{R}$ and

$$\begin{aligned} \tau(x,y) &:= (x,y)\,, && \text{for } (x,y) \in R\,, \\ \tau((x,y),(u,v)) &:= (\lambda x + u, \lambda y + v)\,, && \text{for } ((x,y),(u,v)) \in U \times O\,. \end{aligned}$$

The function ϱ is uniquely determined by τ. An explicit definition is

$$\varrho(x_n, y_n)_{n<\alpha} := \left(\textstyle\sum_{n<\alpha} \lambda^n x_n, \sum_{n<\alpha} \lambda^n y_n\right).$$

Having defined the outcome of a game, we can introduced equilibria. Consider an game γ with set of players N and set of outcomes $U := \mathbb{R}^N$. Let $\tau : \Gamma(U) \to U$ and $\varrho : \Omega \to U$ be the functions to compute the outcome of γ. For a tuple $\bar{\sigma} = (\sigma_p)_{p \in N}$ of strategies, we denote by $\varphi[\bar{\sigma}] : S \to \Omega$ the function from the reduced game $\gamma[\bar{\sigma}]$ to the final coalgebra.

Given strategies σ_p, for each $p \in N$, and an initial game state $s_0 \in S$, we say that σ_p is a *best response* to the other strategies if

$$\varrho(\varphi[\bar{\sigma}](s_0)) \geq \varrho(\varphi[\bar{\sigma}'](s_0))\,,$$

for all tuples $\bar{\sigma}'$ that differ from $\bar{\sigma}$ only in the p-th component.

The tuple $\bar{\sigma}$ is a *Nash equilibrium* of γ if, for every player $p \in N$, σ_p is a best response to the other strategies.

2.8 Summary

Summing up the preceding sections, we have seen that we can specify a game by the following data:

- a set N of *players,*
- for each $p \in N$, a set A_p of *actions* for player p,
- for each $p \in N$, a set B_p of *observations* for player p,
- a set S of *states* of the game,
- a set R of *results,*
- a set O of *output values,*
- a set U of *outcomes,*
- a function

  $$\gamma : S \to \mathbb{C}(R + S \times O)^{\prod_{p \in N} A_p}$$

  computing a single step of the game,
- for each $p \in N$, a function

  $$\beta_p : O \times \prod_{p \in N} A_p \to B_p$$

  computing the observations of player p, and
- two functions $\varrho : \Omega \to U$ and $\tau : \Gamma(U) \to U$ that satisfy

  $$\varrho = \tau \circ \Gamma(\varrho) \circ \omega$$

  and that compute the outcome of a play.

3 Examples

In this section we present the formulations of two basic games in our framework. The first game is one with imperfect information as imperfect monitoring. The second game is one with incomplete information.

The usual approach in economics is to reduce incomplete to imperfect information. Incomplete information denotes situations where the type of agents is not known while imperfect information denotes situations where the state of the game is not known.

In our framework this differentiation is not important both kinds of lack of information are captured by unobservable state spaces. However, some more experiences in representing economic games in our framework will clarify these differences.

3.1 Imperfect Public Monitoring

The imperfect information game with imperfect monitoring and noisy signal considers games where the agents' actions may not be directly observable. The state of the game is driven by a probabilistic state transition and may be either "good" or "bad". This information is publicly observed by the agents, i.e. all players observe the same signal. The payoff is a function of this public outcome.

Again we have two players with two actions each: $N = \{1, 2\}$ and $A_p = \{c, d\}$. There are no results $R = \emptyset$ since the game never ends. The output values are $O = \mathbb{R} \times \mathbb{R} \times Y$ with $Y := \{G, B\}$ that encode the pay-offs in the stage games and the public signal. The game has a single state $S = \{*\}$.

The game for the probabilistic functor $\mathbb{C}_\gamma = \mathbb{C}_{\text{prob}}$, is the function

$$\begin{aligned} \gamma : S &\to \mathbb{C}_{\text{prob}}(S \times O)^{A_1 \times A_2} \\ (*, a_1, a_2) &\mapsto (*, (r_1, r_2, y)) \end{aligned}$$

where

$$r_p = \begin{cases} 1 + \frac{2-2k}{k-m} & \text{if } (a_p, y) = (c, G) \\ 1 - \frac{2k}{k-m} & \text{if } (a_p, y) = (c, B) \\ \frac{2-2n}{m-n} & \text{if } (a_p, y) = (d, G) \\ \frac{-2n}{m-n} & \text{if } (a_p, y) = (d, B) \end{cases}$$

$$y = \begin{cases} G \text{ with probability } k & \text{if } (a_1, a_2) = (c, c) \\ G \text{ with probability } m & \text{if } (a_1, a_2) = (c, d) \vee (a_1, a_2) = (d, c) \\ G \text{ with probability } n & \text{if } (a_1, a_2) = (d, d) \\ B \text{ with probability } 1 - k & \text{if } (a_1, a_2) = (c, c) \\ B \text{ with probability } 1 - m & \text{if } (a_1, a_2) = (c, d) \vee (a_1, a_2) = (d, c) \\ B \text{ with probability } 1 - n & \text{if } (a_1, a_2) = (d, d) \end{cases}$$

The probabilities of the state transition are characterize by the parameters $k > m > n$. The parameters are chosen such that the expected value of the payoffs

is given by the prisoner's dilemma matrix:

	c	d
c	$(1,1)$	$(-1,2)$
d	$(2,-1)$	$(0,0)$

The game has imperfect information so that the epistemic state of the players is not the state of the game. Each player knows the history of his actions and the history of the public signals

$$E_p = (A_p \times Y)^*.$$

The observation function is

$$\begin{aligned} \beta_p : O \times A_1 \times A_2 &\to B_p \\ ((r_1, r_2, y), a_1, a_2) &\mapsto (r_p, y, a_p). \end{aligned}$$

We consider deterministic strategies with choice functor $\mathbb{C}_\sigma = \mathbb{C}_{\text{det}}$. An example of always (unconditionally) playing d for a player p is given by

$$\begin{aligned} \sigma_1 : E_1 &\to (E_1 \times A_1)^{B_1} \\ (h, (r_1, y, a_1)) &\mapsto (h(a_1, y), d) \end{aligned}$$

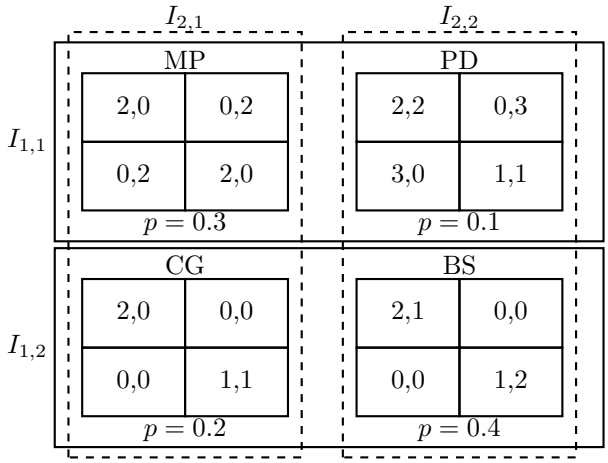
where $h(a_p, y)$ denotes that the history h of the epistemic state is extended by the action a_p and the public signal y.

3.2 Incomplete Information

In a Bayesian game of incomplete information the types of agents are not common knowledge. In the simplest case we take types of agents to be represented as different payoff functions of the game and each agent knows his own type but not the one of his opponent.

In the following example we define that the game is played only once. However, our framework is rich enough in order to easily extend the game to be played finitely, infinitely or potentially infinitely often. The types of agents can be drawn repeatedly or as in the following example only once. The agents can use Bayesian updating or in fact any kind of learning rule.

We define a Bayesian game of four 2×2 games: Matching Pennies (MP), Prisoner's Dilemma (PD), Coordination Game (CG) and Battle of the Sexes (BS) with equivalence classes $I_{i,j}$ for players i and types j. Player 1, if of type 1, knows that the payoff is either MP or PD and if of type 2, that the payoff is either CG or BS. Player 2, if of type 1, knows that the payoff is either MP or CG and if of type 2, that the payoff is either PD or BS. The probabilities are given by $p_{MP} = 0.3, p_{PD} = 0.1, p_{CG} = 0.2$ and $p_{BS} = 0.4$.



The state space of the game is $S = \{*, MP, PD, CG, BS\}$. The output space of the game is $O = S$ and the result space is $R = \mathbb{R} \times \mathbb{R}$. The action spaces are $A_1 = \{U, D\}$ and $A_2 = \{L, R\}$ and the epistemic state spaces have a single state $E_1 = E_2 = \{*\}$. We formalize this game in two rounds.

1. In the first round the game is in state $*$ and nature realizes the types of the players, the actions of the players are irrelevant.

$$\gamma : S \to \mathbb{C}_{\text{prob}}(R + S \times O)^{A_1 \times A_2}$$

$$(*, a_1, a_2) \mapsto \begin{cases} (MP, MP) & \text{with } p_{MP} = 0.3 \\ (PD, PD) & \text{with } p_{PD} = 0.1 \\ (CG, CG) & \text{with } p_{CG} = 0.2 \\ (BS, BS) & \text{with } p_{BS} = 0.4 \end{cases}$$

   In the second round the game yields a result.

$$(MP, a_1, a_2) \mapsto \begin{cases} (2, 0) & \text{if } a_1 = U, a_2 = L \\ (0, 2) & \text{if } a_1 = U, a_2 = R \\ (0, 2) & \text{if } a_1 = D, a_2 = L \\ (2, 0) & \text{if } a_1 = D, a_2 = R \end{cases}$$

   An analogous definition has to be given for the other type realizations PD, CG, BS.

2. The observation function of the first player is

$$\beta_1 : O \times A_1 \times A_2 \to B_1$$

$$(o, a_1, a_2) \mapsto \begin{cases} * & \text{if } o = * \\ \{MP, PD\} & \text{if } o = MP \vee o = PD \\ \{CG, BS\} & \text{if } o = CG \vee o = BS \end{cases}$$

   The observation function of the player 2 is defined analogously.

3. The strategies for both players are given by

$$\sigma_p : \{*\} \to \mathbb{C}(\{*\} \times A_p)^{B_p}$$

For example, the strategy of player 1 who plays U if he is of type 1 and D if he is of type 2 is given by

$$(*, b_1) \mapsto \begin{cases} U \text{ if } b_1 = \{MP, PD\} \\ D \text{ if } b_1 = \{CG, BS\} \end{cases}$$

In the first round when nature chooses the types, the strategy of the players is irrelevant since it does not matter in the game function.

4 Unification

We have shown how to instantiate games in our general framework. Its high level of abstraction provides a path to a unification with other subfields of economics. Beside game theory we want to mention the relations to computational, network and agent based economics, macroeconomics and econometrics, and to point to the reflexive structures in these subfields. The coalgebraic tools are related to functorial fixed points on the category of sets. These fixed points are well suited to model reflexive and self-referential structures that are so far hardly modeled in social sciences even so they constitute the core feature of societies. By reflexivity we mean the interaction of an object with its encoding or of a context with its content. This happens in general in societies where modelers are part of the modeled system and mutually model each other.

4.1 Computational Economics

Our framework is built on mathematical techniques that were developed in theoretical computer science in fields such as modeling and verification of processes, semantics of programming languages, logic, and automata theory. These techniques are well suited to model interactive behavior and our framework accordingly provides the beginning of an interpreter for an economic language. The models therein are representable as executable code. This helps on the difficult, not formalized, error prone, unreliable and accordingly tedious ad hoc approach in computational economics when it comes to simulate, analyze, solve and synthesize economic theories in software. A next step is to program a game theoretical engine with our type definitions, natural transformations and basic operations for the composition of games. Games are to be programed in this framework as instantiations and transformations of the defined types of the engine. In the development of our framework we need to iterate between examples and improvements of the interpreter of the language and to extract the reoccurring operations and specifications into the vocabulary that is needed in game theory and economics.

Our approach to computational economics aims to extend the software for simulation and numerical solution of models by proof assistants. An important next goal is to add the logical structures that naturally come with coalgebras and that are used in computer science, system and language design in order to proof

and specify properties of systems. At the specification level we need to provide logical predicates for the specific needs in the domain of economics at a high level like predicates of equilibria or other properties of strategy profiles. Logical specification languages and predicates defined in this way may contribute to the development and increased usage of economic ontologies in economics as it is done in the economy and the research on the semantic web. This aims to reduce the ambiguities regarding the meaning of the expressions that are used in natural language economics and to foster interdisciplinary collaborations. At the verification level we need to provide tools to check for example safety conditions like invariances over state transitions. These are often some kind of logic modalities of necessity and possibility of some temporal, epistemic or deontic kind.

Logical tools can assist during abduction and debugging cycles of theory development. Abduction is the process to infer additional properties or restrictions on the maintained set of logical sentences such that intended theorems can be proved [9]. The theorems may involve safety conditions, desired properties or some empirical facts to be explained. This process of induction during deduction is hardly formalizable but computational tools may assist in generating counter examples that can then be used to induce additional restrictions by the human capability of pattern matching. Here it is of paramount importance to define formal translations (natural transformations) between the representations (functors) since patterns and their detection do depend on the representation. The generation of counter examples by the logical tools of model generation is important for the debugging cycle of theory development that is so far done by hand.

4.2 Computing Economic Agents

The current approach in economics to proof properties of theories is rather specific to the theories and properties of interest. Our abstract framework can provide a uniform approach to this logical goal since coalgebras come with general proof systems [70, 71]. The predicates and logical tools that need to be integrated into our framework serve the central goal to proof properties of games like the question what equilibria can result from certain strategies.

A dual goal is to ask what strategies can result in certain equilibria or what specific outcomes can be the result of some behavior. An interesting question is whether the duality of both tasks, analysis of games by asking for outcomes given behavior and the design of games by asking for behavior given outcomes can be formally dualized.

The goal of establishing a formal duality of the tasks of design and analysis of games is to endogeneize the policy maker into theory. Agents should be able to analyze the game they play as well as to change the rules of the game in order to model the policy maker being integrated in the model. This again points to the need to model reflexive structures that most prominently appeared in the Lucas critique. The basic reflexive structure of lambda calculus suggests an obvious approach. If players are modeled as computers (or universal Turing machines as coalgebras [57] or monoidal categories [91]) then the computation of payoffs can be taken to be a player as well [81]. Hence, lambda calculus or computable functions provide approaches to the reflexivity of institutional economics as the interaction of the context and the content.

4.3 Reflexivity

Reflexivity is a self-referential pattern or a kind of fixed point where objects and their encoding interact. This is an important, if not a defining feature of biological and social systems [117, 66, 76, 52, 86, 97, 96].

The usual fixed points in economics are Brouwer's kind of fixed points. They are not suited to accommodate the reflexive nature of societies, we need fixed points on structures, hence functorial ones which generalize largest fixed points on posets.

The intuition how coalgebras generalize largest fixed points is to understand partial orders as categories, where the objects of the category are the elements of the partial order and there is at most one arrow between any two objects indicating the order relation between them. Having at most one arrow captures partiality. Functors, that preserve order, can be taken as monotonic functions. Final coalgebras arise as functorial fixed points and generalize largest fixed points on posets to any category and functor [8]. Final coalgebras μF can be taken as functorial fixed points since their structure maps are isomorphisms by the Lambek lemma, hence $\mu F \simeq \mathbb{F}(\mu F)$.

In game theory largest fixed points have been already used in a small number of influential applications. An existing strand in game theory are supermodular games that have been build on largest fixed points on posets in order to model strategic complementarities and multiple equilibria [84, 82, 83]. In the macroeconomic literature largest fixed points of posets have been used for the approximation of value function iteration on Bellman equations in dynamic programming [28, 29, 27, 69]. An important paper in game theory is [6] that became an important technique in the literature on infinitely repeated games, see the text book [79]. This technique uses largest fixed points of set valued functions, heads and tails as continuation values, equivalence classes of behavioral equality like bisimulation and automatons. The goal is to characterize a set of payoffs that can be the result of subgame perfect equilibria in infinitely repeated games.

These largest fixed points on posets have been steps in the direction we follow but they went not far enough into the functorial fixed points of the coalgebraic approach. It is interesting to understand how these economically motivated constructions can be merged into the abstract coalgebraic techniques.

An important generalization of the games that we have seen in our framework are epistemic games that are based on largest fixed points as well [21, 20]. In epistemic games one goal is to construct a domain with infinite hierarchies of beliefs of beliefs that arise for example in Keynes' notion of a beauty contest that drives expectation formation on prices. The circularity arises if fundamentals drive prices and beliefs that in turn drive fundamentals. The implied question is whether any kind of opportunity cost argumentation needs to ultimately answer how the feedback of mutual opportunities can be resolved.

One goal of epistemic game theory is to substantiate the notion of common knowledge and rationality. Common knowledge is a coinductive modality [23]. The domains that are constructed in epistemic game theory are Harsanyi type spaces [51] and have been constructed as coalgebras on the category of measurable spaces [85]. The unobservable state space models are accordingly a natural approach to private hence unobservable types of agents.

The coalgebraic construction of functorial fixed points on the category of sets can be extended to the category of measurable spaces as in the case of Harsanyi

types spaces. In case of functorial fixed points on games in networks played over networks we may change the underlying category of sets to the one of relations or generalizations thereof like the category of span(graphs) that was used for parallel processes [64]. The relational categories need to capture ultimately the process logical nature of double accounting as a core technique of economies which is a theory of the measurement of parallel processes with only local exchange of information such that a global state of a balanced sheet is maintained [63]. The underlying problem to be solved is a stock-flow consistent macroeconomic approach [53, 72] in order to generalize Walras law, double accounting and budget constraints, into some sort of decentralized constraint propagation algorithm that represents the search processes in the economy across markets, hierarchies and time. This requires some relational category where we can model money and value theory that arises as a property of the network where the money is used. It is a rather complicated problem since within the time structure and belief formation that determine the value of money we have to model the banking and financial sector, fiscal and monetary policy as well as theories of capital formation and ultimately decentralized and hierarchical structures.

The formulation of agents as coalgebras or unobservable state space systems prepares the ground for the integration with the learning literature of agents that are isomorphic to the econometrician [100]. This, as well as the dualization of analysis and design of games, brings us to reflexivity as the core issue of social systems, the interaction of the model and the theory, or the object and its encoding, respectively, that is underlying the Lucas critique. We can instantiate one form of reflexivity in our networks by games where the strategic actions are the structures of the network. Hence we may formulate games in the network being played about the network. These games connect us to institutional economics or the interaction of the context and the content. We have argued above that the reflexive lambda calculus,hence Scott domains or in general tools of computable functions are natural tools for institutional economics.

Just as the coalgebraic or functorial fixed points can be switched from the category of sets to the category of measurable spaces in order to accommodate the construction of beliefs of beliefs in terms of distributions over distributions we may take functorial fixed points to construct the domains for rules to change rules, games over games or as in our case of games over networks. The early work of Vassilakis [115, 116, 114] builds on the computer scientific work on data types and Scott domains and applies it to some economic reflexivities. However, coalgebras have not been developed at that time and accordingly Vassilakis has used for his functorial fixed points the category of domains, which has ordered sets as objects. Hence, our coalgebraic approach is a simpler starting point without the complications of domains, which we may need to explore later.

One of the most important features of category theory is that the categorical universal constructions are self-participating and do not dependent on the structures they operate on. For example, the characterization of products as categorical limits or dually of disjoint unions as colimits is independent of the structures they operate on, be them sets for cartesian products or topological spaces. This data independency and universality is the underlying reason why category theory is well prepared to discuss the notion of compositionality and modularity. Moreover, categories are in a sense fractal or hierarchical themselves since the functors form the objects of a category with natural transformations as arrows. This kind of hierarchical constructions can be extended infinitely

and are very convenient for reflexive structures. Both, self-participation and fractality, contribute to an evolutive modeling approach. Once constructions on categories repeat themselves they can be relocated into the underlying category in order to simplify the constructions on top of it. The goal for economics is to arrive at a category of the economy as its state with functors as state transition in the category of categories.

We have seen in the definition the essential building block of categories as the composition of arrows. This makes category theory into a necessary tool for a relational or structural social science [41] that can hope to understand the nature of complex or compositional systems [101]. The ability to construct properties by self-participating universals has far reaching philosophical consequences one is that we can hope for policies that respect self-determination of humans in form of a help to help themselves [40]. The non-self-participating nature of set theory suggests that it is insufficient to understand reflexivity and therefore synthesis and ultimately societies.

Reflexivity is a deep common root of mathematics, computer science, biology, economics, linguistics, social science and philosophy. In computer science it appears most prominently as the need to model languages that need to be able to interpret themselves [15, 14, 13]. This appeared for the first time in computability theory where programs processing the codes of other programs (or their own) were introduced. It resulted from the simple but far reaching idea of von Neumann to place code and data in the same memory and hence to make code into data of (other) code just as a theory appears in the modeled systems in economics. One approach to this code as data problem is based on the untyped lambda calculus, a bare-boned functional programming language where functions operate on functions. This led to functorial fixed points of type $D \simeq D^D$. A solution was finally provided by domain theory [105, 4].

The solution for these so called reflexive domains is the genuine reflexive problem of self-referential structures. They can be detected in natural languages as expressions of the form X of X [10, 12, 11], set of sets, function of functions, belief of beliefs, rules of rules, game of games or economics of economics. These are interactions of the object and its encoding as in the cognitive act of Anna calling herself "Anna", the Anna of "Anna". It is the identity generating act of naming and accordingly the mathematics of quoting that is captured by functorial fixed points.

The Lucas critique can be formulated as the need for the economics of economics to be economics or for the need to model the economic agents isomorphic to the econometrician, both being inside the modeled systems. The economics of economics studies the production function of economics. This paper being about reflexivity practices reflexivity since it wants to change the production function of economics. This process is a higher order function since economic consulting takes production functions as inputs, transforms them and outputs new ones, just as in lambda calculus. Economics professors produce economists who become consultants who transform production functions. But economic professors also produce other economic professors. The economy has to be described by a higher order production function just as Robinson Crusoe's coconut which produces a coconut – out of a coconut. Similarly capital is an operator, input or output and in need of a type theory just as value theory and money theory. One function of money is to provide a unit for accounting, hence money is a type constructor and exchange rates some natural transformations. Once

we have the reflexive tools of recursive functorial fixed points at our disposal they appear to be a natural starting point in order to model the prevailing circularities in social cognitive systems that think about themselves. However, coalgebras on the category of sets and recursive domain equations $X \simeq \mathbb{F}(X)$ that are solved by final coalgebras are a natural interface to import and then generalize the usual mathematics of economics.

4.4 Networks

We can place agents in our framework into a network structure that can be further generalized to any algebra and thus a formal syntax. The network in our example represents the locality of the observed information in the game of imperfect monitoring. There are many other different kinds of hierarchical structures that may be encoded in that way: workers that aggregate into firms, then into sectors, then into economies, and finally into the world economy. At each level different information may be modeled to be available. Beside the hierarchical or informational structures, of course, we may encode networks as spatial structures. Another important hierarchy are time scales that naturally arise in models of overlapping generations or yield curves managed by central banks. In general, parallel and hierarchical processes with different time scales can not be controlled by single concrete control units and need to be managed by some emergent or composed control structure [38]. Hence, probably the most important hierarchical structure that we study in social sciences is the hierarchy of control that brings us into the realm of political economics, political science and theories of democracy. Again, democracy is a reflexive structure in that peoples are meant to reign themselves.

4.5 Agent Based Economics

Our framework can be taken as a complementary approach for agent based economics [111]. We provide an agent based programing language and the semantics for it. Object oriented languages are the main kind of languages in the agent based approach that simulate data that trace the interactions of the programmed agents. The data is then used to statistically infer the properties of the composed systems. This is a convenient way to get familiar interactively with the maintained theory and to generate hypotheses but it can not substitute a formal account of the compositionally of social systems just as simulations can not substitute the formal derivation of the behavior of the programing languages in theoretical computer science.

Probably the most sever problem is that ontological and epistemic states, of the game and the agents, respectively, as opposed to our approach can not be properly distinguished in agent based approaches [112]. It is not clear whether a program needs a global or local state as a feature of the model or due to the implementation details. Without such a distinction it is also not clear whether emergent properties of the system arise as a feature of the system or as an emergent understanding of the modeler. So, do simulated emergent properties arise from the interaction of the subparts of the system or the ignorance of the modeler?

The linguistic or knowledge representation caveat in the simulation based approach is that object oriented languages are simpler to use and are, at least

pragmatically, more expressive than differential equations. But this expressivity is payed by a reduced decidability in terms of statistical inference only. It is better to directly approach the problem by tools of language design and to find an optimal tradeoff between expressivity and decidability, tailored to the modeling problem at hand. The optimal tradeoff is the core issue in the knowledge representation research that needs to be applied to economic language design [48, 19].

Object orientation in programing languages was meant to cope with large software systems by compositionality and modularization of components. It was developed without a formal semantics but it is now formalized by coalgebraic semantics in theoretical computer science [95, 59]. Object orientation is a way to cope with complexity by modularizing subparts that interact only at their interfaces with some contracted behavior arising from unobservable state spaces. Hence, coalgebras are a natural approach to object oriented modularization.

A formal semantics or account of the composition of behavior from the behavior of subparts as opposed to simulation alone is especially important in order to understand how errors propagate in systems during the interaction of the parts and also how to properly design and compose systems from parts. Both, the error propagation and design issues arise equally in programing languages and social sciences and can not be sufficiently studied by the current approach of statistical inference of system properties from simulated data.

Hence, our approach to compositionality by natural transformations can be taken as an attempt to complement the simulation approach in agent based economics. We do not want to rely on the law of large numbers alone in order to derive properties of the behavior of the composed system but aim at analytic, or more accurately, synthetic composition.

4.6 Macroeconomics

The macroeconomic literature relates to our approach obviously through the goal to model aggregation. At the technical level both are related by states that are distributions that result at the next level in distributions over distributions. These features arise during the composition of stochastic micro agents and macro games. In the literature on heterogenous agents in macroeconomics [49] we find similar structures in terms of idiosyncratic and aggregated shocks and distributions as states. In our framework we see distributions as states and distributions over distributions as natural transformations.

On the technical level of the languages of microeconomics and macroeconomics there is still a gap to fill given that macroeconomics and economic dynamics is traditionally formulated in differential and difference equations. Coalgebras can unite these structures with our approach as well. Calculus has a coalgebraic form [90], metric coinduction [68, 67] can operate on functional vector spaces [75] and the continuum can be constructed as a final coalgebra [89]. However, a methodological and philosophical question is whether there are any real numbers in the economy at all and what kind of approximation are we working at if real numbers approximate discrete structures of reality and discrete computers approximate the real numbers.

The categorical machinery points to a more elaborate mathematics of formal aggregation. Our natural transformations simply state the composition or aggregation operations but we are also interested in the conditions which

enable aggregation and show how additional properties arise from the combination of subparts without being properly in the subparts. These structures are mathematically discussed in sheaf theory [78] that formalize the transition from local to global structures in mathematics as in algebraic geometry. In the computer scientific literature [46, 45, 47, 80] they have been used to formalize composition by categorical colimits (that construct initial algebras and generalize disjoint unions) and to formalize behavior by limits (that construct the final coalgebras and generalize products). The sheaf conditions describe how objects arises out of observations from different points of view and when these observations can be consistently composed. In econometrics we similarly ask how differently identified structural models fit together on the behavioral level. After all models are meant to incompletely describe the one and only economy and we would like to know whether they are consistent in one way or another. If sheaves are used to model the world then cosheaves can be used to weight the models [31]. In econometrics this may be used for a generalized model weighting as in Bayesian statistics. In theory this may be used to arrive at the identity of a social system that arises from the models that are maintained by the individual agents. This can be related to the maximally self-fulfilling notion of rational expectation models and to the question for conditions that render expectations to be rational derived from the same model of the observer and the observed.

4.7 Econometrics

The program of the rational expectations revolution is to model the economically reasoning agents in economic models as econometricians [100] that decide upon observations. However, it is still at stake to pass beyond the so called model communism which denotes that the observer and the observed maintain the same model. Our framework is well prepared to this kind of reflexivity of the interaction of the object of interest (the economy) and its encoding (in the models of the agents interacting in the economy).

Our agents and games are represented by processes that are coalgebras. Coalgebras formalize the unobservable state space models [99] that underly the Kalman filter. The duality of Kalman filters and control systems is well known in macroeconomics. Coalgebras and the inherently duality based category theory are therefore natural generalizations of these economic tools.

A causal empirical theory suggests (directed) conditional independence assumptions but data does only provide (undirected) correlations. A major problem in econometrics is accordingly the fact that in general in parametric specifications of joint densities there are more causal parameters than can be delivered by correlations. Hence, there is a gap that in econometrics is filled under the heading of identification. We are left with classes of theories that are, in econometrics so called observationally equivalent [108]. This is called behavioral equivalence and bisimulation in coalgebra and establishes an algebraic path to the core problem of econometrics that relate by identification the correlations of data to the causal theories of interest. The only way to do this is to change and experiment the systems, hence to formalize the notion of an experiment given non-experimental data only. Non experimentality refers to the inability of totally control the environment of the experiment in order to unambiguously identify causalities. Hence, a core issue of science is the dual notion of construction and observation and therefore of algebras and coalgebras, see also [113].

Statistical models and their implied joint densities are represented as graphs in the machine learning literature of Bayesian networks [93, 92]. Bayesian networks use the conditional independence assumptions to reduce the complexity and increase compositionality during the modeling process by visualizing equations of joint densities as graphs. A graph theoretic representation allows as well to exploit the composition of large Bayes networks from simpler ones in order to automatically tailor efficient algorithms to irregular networks. We can think of this approach as a decomposition of sparse matrices in linear econometric models along their block structure in order to avoid the inversion of the whole matrices. The graph theoretical representation replaces the usual matrix inversion in econometric algorithms by belief propagations in networks that maintain the global property of the validity of the Bayes formula by a local exchange of informations as a decentralized communication algorithms between appropriate subgraphs [54]. This ultimately represents the statistical inversion and learning of the densities of the unobservable random variables from observed data.

This graphical approach meets with our framework in influence diagrams [60]. Influence diagrams extend the nodes of Bayesian networks that represent random variables by decision nodes which are set by agents given some utility nodes. Here the composition and decomposition is along the mutual influence of the involved agents on their payoff functions rather than conditional independencies as in Bayes networks. Therefore influence diagrams generalize the decomposition of random variables in Bayes networks via conditional independence assumptions to non-influence assumptions. Our framework relates to influence diagrams since our observing agents do interact in networks as well.

However, Bayes networks are only probabilistic models at the level of propositional logic and Markov logic is developed in order to extend them to predicates within a suitable probabilistic approach [35]. Our goal to unify our approach with econometrics and modal logic is similar. In terms of a categorical approach to Bayes networks [43] and logic, the path to an algebraic predicate-based statistics is via adjoint functors as logical quantifiers [30, 110]. Predicates are needed for a more expressive language beyond propositional logic in order to improve models of relational structures [35]. However, it is important to recapture the role and tools of statistics in that it is usually meant to capture the unmodeled part of the world that is the general goal of approximation and that can be approached algebraically by the tools of abstract interpretation [42, 104, 102] developed in program language semantics. Statistics is also meant to provide a model of uncertainty. However, uncertainty in social systems may be cast as a non-well founded [50] and hence coalgebraic structure. How to combine nonmodeling, uncertainty, statistics and coalgebra is therefore on open quest for the future.

A possible path might be to consider coalgebraic structure as fundamentally capturing openness as infinity and by that uncertainty. There might be an economically interpretable relation to the open world semantics of knowledge representation research [48]. An interesting questions is how the asymmetry of a "no" (proved "no" or not provable) in a theorem prover and the open world semantics relate to the asymmetry of losses and gains in the cumulative prospect theory of Tversky and Kahneman. The open and closed world semantics are different with respect to the assumptions on the truth value of non mentioned sentences. This is related also to non monotonic reasoning where an additional logical sentence may render a formally provable theorem unprovable. This sug-

gests a formal approach to model that in an open world new information may render a formally valueless option valuable, a situation that is impossible in the world of monotonic reasoning where it is always the case that additional sentences may not be used. The links of this line of reasoning might contribute to the literature of incomplete contracts and markets in economics. However these intricacies turn out to be resolvable, they strongly point to the need for logical systems in social sciences that are beyond the exclusive "true" and "false" but ones where a third "I don't know" value is possible and important. A statistical approach to the third value is in [32]. Coalgebras, that are not build on the axiom of foundation, seem to allow these lines of thought, that underlies Luhmann's sociology, since the Russell sets "do, do not, do, do not ..." contain themselves. The infinite sequence of "true" and "false" may be taken to represent the third truth value "I don't know" of both "true and false".

4.8 Endogenous Language

A theory of a sufficiently complex system is an approximation to a certain degree for some subparts of the system. The motto of numerical economics is that the question is not whether but where to approximate [61]. Numerical economics usually approximates a model that is assumed to be the truth and we need to rethink the model and proof theory of economics. However, the numerical motto is an approximation and we need to take it serious that a theory is an approximation and the questions is what is an approximation of theories and what a theory of approximations.

We need to ask for the marginal utility of improving a model by Taylor expansions at the level of functors and domains [31] and by a transition from the local approximating structures to the global object, the society. But again, there is the reflexive caveat: social theories are only approximations of the modeled system, and yet they are meant to be implemented as policies or institutions in order to become the rules that drive the systems.

The quest for approximation is answered in computer science under the heading of abstract interpretation [42, 104, 102] that provides a notion of approximation of the semantics of a programing language. Types are put in hierarchies like booleans that approximate natural numbers, integers, rationals and reals and the goal is to ensure behavioral invariance under different levels of approximations. In economics we want to add some features to a theory and ask which behavior is preserved and which one is new and most of all how is it related to the measured behavior of the modeled system. In terms of compositionality of theories it is of paramount interest to check whether and in what sense different theories show consistent behaviors. Of course, in order to take advantage of the toolbox of formal semantics and abstract interpretation we need to interpret the meaning of the computer scientific concepts in the realm of economics, with operational and denotational semantics being two important ones.

The framework we are proposing is based on category theory and we have to ask how to endogeneize even this process as the generalization of the goal to model the modeler. We need to take the endogeneization of scientific, every day and political economic languages as our ultimate problem of the coevolution of objects and their encoding [33, 37]. A general point is that we usually can only see, understand and intentionally construct what we can write down as a theory, if only for the purpose of communication and cooperation. An important

question is to account for the notion of truth implied by a given language, mathematics or dogma in social science. This goal to endogeneize the language itself may be approached mathematically by topos theory that can account for local mathematical languages [17, 33] and differentiated notions of truth that we may take to be something beyond "true" or "false" or ultimately as some degree of approximation only and most of all dependent on the point of view and some individual notions of (relative) truth or mutual inconsistency [16].

5 Conclusion

We have seen in this paper how to represent a variety of games in one unified and abstract framework. One goal of this framework is to ensure compositionality of games from simpler ones for an approach to synthesis in social science. Our framework is sufficiently abstract in order to be uniform over types of games and other economic theories.

Another goal of our framework is to unify economics and its main tools under a sufficiently abstract language. The coalgebraic language can subsume all of game theoretical models and it comes with internal logics, proof systems and a direct implementability in code. It most of all provides the simplest account of the core feature of social sciences namely reflexivity. The fixed points that are used in economics have to be accompanied by the functorial fixed points that allow us to model reflexivity and hyper structures such as beliefs of beliefs or the history of histories for their reinterpretations and ultimately we need to arrive at the economics of economics where the economic process of knowledge generation itself is formalized and decentralized. In the present paper we have seen that the simplest domain for hyper structures provides us with the behaviors of many models specified in game theory. The generalization into more general functorial fixed points seems to be a proper path for many of the insufficiently modeled economic issues.

The linguistic point of view suggests a division of labor for economics. Theoretical economics supplies the linguistic means to express economic theories. Applied economics on the other side uses the language in order to formulate theories and extract them from data. The goal is to answer economic questions in order to implement economic policies for and by groups of individuals and to set up their institutions. The pragmatic development of the formal economic language is to start with ontologies that are used in applied work and extract the repeating patterns into the vocabulary and the syntax of the engine and the interpreter of the formal language. Hence, the development of economic theory takes place as the reflexive interaction of the syntax and the semantics of its language. And this is what we need to model: the institutional economics of an interacting network (syntax) and behavior (semantics) as a game that is played in a network over the structure of the network. Reflexive agents are autonomous if they form their environment or if the categorical imperative of behavior may be thought as the rule that materializes as an institution. Thus reflexivity is the core of the European enlightening [52] and still far from being at its end - if not mathematized.

References

[1] S. Abramsky and N. Tzevelekos. Introduction to categories and categorical logic. In Bob Coecke, editor, *New Structures for Physics*. 2011.

[2] S. Abramsky and J. Zvesper. From lawvere to brandenburger-keisler: interactive forms of diagonalization and self-reference. *Arxiv preprint arXiv:1006.0992*, 2010.

[3] Samson Abramsky and Bob Coecke. Categorical quantum mechanics. In Kurt Engesser, Dov M. Gabbay, and Daniel Lehmann, editors, *Quantum Logic and Quantum Structures*. Elsevier, 2008.

[4] Samson Abramsky and Achim Jung. Domain theory. *Handbook of Logic in Computer Science*, 3:1—168, 1994.

[5] Samson Abramsky and Viktor Winschel. Coalgebraic analysis of subgame-perfect equilibria in infinite games without discounting. *arXiv:1210.4537*, October 2012.

[6] D. Abreu, D. Pearce, and E. Stacchetti. Toward a theory of discounted repeated games with imperfect monitoring. *Econometrica*, 58(5):10411063, 1990.

[7] F. Ackerman. Still dead after all these years: interpreting the failure of general equilibrium theory. *Journal of Economic Methodology*, 9(2):119139, 2002.

[8] J. Adamek, S. Milius, and L. S Moss. Initial algebras and terminal coalgebras: a survey. 2010.

[9] Konstantine Arkoudas. Specification, abduction, and proof. In *Automated Technology for Verification and Analysis*, pages 294–309. 2004.

[10] N. A Baas and C. Emmeche. On emergence and explanation. *Intellectica*, 25(2):6783, 1997.

[11] Nils A. Baas. Hyperstructures, topology and datasets. *Axiomathes*, 19(3):281–295, June 2009.

[12] Nils A. Baas, Andre C. Ehresmann, and Jean-Paul Vanbremeersch. Hyperstructures and memory evolutive systems. *International Journal of General Systems*, 33:553–568, October 2004.

[13] H. Barendregt. Reflection and its use: from science to meditation. *Spiritual Information, Templeton Foundation Press, Philadelphia and London*, pages 415–423, 2005.

[14] Hendrik Pieter Barendregt. *The lambda calculus. Its syntax and semantics.* North-Holland Elsevier, 2 edition, 1985.

[15] Henk Barendregt. The impact of the lambda calculus in logic and computer science. *The Bulletin of Symbolic Logic*, 3(2):215, 181, 1997.

[16] C. Baskent. Some non-classical approaches to the branderburger-keisler paradox. *Arxiv preprint arXiv:1107.4929*, 2011.

[17] J. L. Bell. From absolute to local mathematics. *Synthese*, 69(3):409426, 1986.

[18] Sebastian Berger. *The Foundations of Non-Equilibrium Economics: The Principle of Circular Cumulative Causation*. Routledge Chapman & Hall, August 2009.

[19] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann, 1 edition, May 2004.

[20] A. Brandenburger, A. Friedenberg, and H. J. Keisler. Fixed points in epistemic game theory. In Samson Abramsky and Michael Mislove, editors, *Mathematical Foundations of Information Flow*. American Mathematical Society, July 2012.

[21] Adam Brandenburger. Epistemic game theory: an overview. 2008.

[22] Adam Brandenburger and H. Jerome Keisler. An impossibility theorem on beliefs in games. *Studia Logica*, 84(2):211–240, November 2006.

[23] Venanzio Capretta. *Common Knowledge as a Coinductive Modality*. 2007.

[24] Manuel Clavel, Francisco Durn, Steven Eker, Patrick Lincoln, Narciso Mart-Oliet, Jos Meseguer, and Carolyn Talcott. *All About Maude - A High-Performance Logical Framework: How to Specify, Program, and Verify Systems in Rewriting Logic*. Springer, 1 edition, September 2007.

[25] Bob Coecke, editor. *New Structures for Physics*. 2011.

[26] David Colander. *Post Walrasian Macroeconomics: Beyond the Dynamic Stochastic General Equilibrium Model*. Cambridge University Press, 1 edition, July 2006.

[27] W. J. Coleman. Equilibrium in a production economy with an income tax. *Econometrica*, page 10911104, 1991.

[28] Wilbur John Coleman II. An algorithm to solve dynamic models. International Finance Discussion Paper 351, Board of Governors of the Federal Reserve System (U.S.), 1989.

[29] Wilbur John Coleman II. Solving the stochastic growth model by policy-function iteration. *Journal of Business & Economic Statistics*, 8(1):27–29, January 1990.

[30] Jared Culbertson and Kirk Sturtz. A categorical foundation for bayesian probability. *arXiv:1205.1488*, 2012.

[31] Justin Curry. Sheaves, cosheaves and applications. *arXiv:1303.3255*, March 2013.

[32] A. P. Dempster. The dempster-shafer calculus for statisticians. *International Journal of Approximate Reasoning*, 2007.

[33] A. Doering and C. J. Isham. A topos foundation for theories of physics: I. formal languages for physics. *arXiv:quant-ph/0703060*, March 2007. J.Math.Phys.49:053515,2008.

[34] Kees Doets and Jan van Eijck. *The Haskell Road to Logic, Maths and Programming*. King's College Publications, London, 2004.

[35] Pedro Domingos and Daniel Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan and Claypool Publishers, June 2009.

[36] Kurt Dopfer. *The Evolutionary Foundations of Economics*. Cambridge University Press, May 2005.

[37] A. Dring and C. J. Isham. What is a thing?: Topos theory chapter 13 What is a thing?: Topos theory in the foundations of physics. In Bob Coecke, editor, *New Structures for Physics*. 2011.

[38] A C Ehresmann and J.P. Vanbremeersch. *Memory Evolutive Systems; Hierarchy, Emergence, Cognition*. Elsevier Science, 1 edition, July 2007.

[39] Jan Van Eijck and Rineke Verbrugge, editors. *Games, Actions, and Social Software: Multidisciplinary Aspects*. Springer, 2012 edition, November 2012.

[40] D. Ellerman. A theory of adjoint functors-with some thoughts about their philosophical significance. In *What is category theory?*, volume 3, page 127, 2006.

[41] Mustafa Emirbayer. Manifesto for a relational sociology. *The American Journal of Sociology*, 103(2):281–317, September 1997.

[42] Franois Fages and Sylvain Soliman. Abstract interpretation and types for systems biology. *Theor. Comput. Sci.*, 403(1):5270, August 2008.

[43] Brendan Fong. Causal theories: A categorical perspective on bayesian networks. Technical report, University of Oxford, 2012.

[44] Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, 1991.

[45] Joseph Goguen. Mathematical representation of hierarchically organized systems. In Ernst Attinger, editor, *Global Systems Dynamics*, pages 112–128. 1970.

[46] Joseph Goguen. Complexity of hierarchically organized systems and the structure of musical experiences. *International Journal Of General System*, 3(4):233251, 1977.

[47] Joseph A Goguen. Sheaf semantics for concurrent interacting objects. *MATHEMATICAL STRUCTURES IN COMPUTER SCIENCE*, 2:159—191, 1992.

[48] Frank Van Harmelen, Vladimir Lifschitz, and Bruce Porter. *Handbook of Knowledge Representation*. Elsevier New York, 2007.

[49] Jonathan Heathcote, Kjetil Storesletten, and Giovanni L. Violante. Quantitative macroeconomics with heterogeneous households. *Annual Review of Economics*, 1(1):319–354, 2009.

[50] Aviad Heifetz. Non-well-founded-type spaces. *Games and Economic Behavior*, 16(2):202–217, October 1996.

[51] Aviad Heifetz and Dov Samet. Topology-free typology of beliefs. *Journal of Economic Theory*, 82(2):324–341, October 1998.

[52] Johannes Heinrichs. *Logik des Sozialen. Woraus Gesellschaft entsteht.* Steno Verlag, January 2005.

[53] Martin Hellwig. The challenge of monetary theory. *European Economic Review*, 37(23):215–242, April 1993.

[54] C Huang and A Darwiche. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, 15(3):225–263, 1996.

[55] Graham Hutton. *Programming in Haskell.* Cambridge University Press, illustrated edition edition, January 2007.

[56] Matthew O. Jackson. *Social and Economic Networks.* Princeton University Press, August 2008.

[57] Bart Jacobs. *Coalgebraic Walks, in Quantum and Turing Computation.* 2010.

[58] Bart Jacobs. *Introduction to Coalgebra. Towards Mathematics of States and Observations.* 2012.

[59] Bart Jacobs and Erik Poll. Coalgebras and monads in the semantics of java. *Theoretical Computer Science*, 291:2003, 2002.

[60] Finn Jensen. *Bayesian Networks and Decision Graphs (Information Science and Statistics).* Springer, July 2001.

[61] Kenneth L. Judd. *Numerical Methods in Economics.* MIT Press, November 1998.

[62] Immanuel Kant. *Metaphysical Foundations of Natural Science.* 1786. Page Version ID: 544675467.

[63] P. Katis, N. Sabadini, and R.F.C. Walters. On partita doppia. *Arxiv preprint arXiv:0803.2429*, 2008.

[64] Piergiulio Katis, N. Sabadini, and R. F. C. Walters. Span(graph): A categorical algebra of transition systems. In Michael Johnson, editor, *Algebraic Methodology and Software Technology*, volume 1349, pages 307–321. Springer-Verlag, Berlin/Heidelberg, 1997.

[65] B. Klin. Bialgebras for structural operational semantics: An introduction. *Theoretical Computer Science*, 412(38):50435069, 2011.

[66] Christian Knudsen. Equilibrium, perfect rationality and the problem of self-reference in economics. In Uskali Maki, editor, *Rationality, Institutions and Economic Methodology.* Routledge Chapman & Hall, 1993.

[67] Dexter Kozen. Coinductive proof principles for stochastic processes. In *Proc. 21st Symp. Logic in Computer Science (LICS06*, page 359366. IEEE, 2006.

[68] Dexter Kozen and Nicholas Ruozzi. Applications of metric coinduction. *Logical Methods in Computer Science*, 5(3), September 2009.

[69] Moritz Kuhn. Recursive equilibria in an aiyagari-style economy with permanent income shocks. Technical report, 2012.

[70] A. Kurz. Coalgebras and their logics. *ACM SIGACT News*, 37(2):5777, 2006.

[71] A. Kurz. Coalgebras, stone duality, modal logic. 2006.

[72] Marc Lavoie and Wynne Godley. *Monetary Economics: An Integrated Approach to Credit, Money, Income, Production and Wealth.* Palgrave Macmillan, second Edition,Revised edition,2nd edition edition, October 2011.

[73] F. William Lawvere. Diagonal arguments and cartesian closed categories. In *Category Theory, Homology Theory and their Applications II*, volume 92, pages 134–145. Springer Berlin Heidelberg, 1969.

[74] Pierre Lescanne and Matthieu Perrinel. Backward coinduction, nash equilibrium and the rationality of escalation. *Acta Informatica*, 49(3):117–137, March 2012.

[75] Lars Ljungqvist and Thomas J. Sargent. *Recursive Macroeconomic Theory.* Mit Press, 2nd ed. edition, September 2004.

[76] Niklas Luhmann. *Die Gesellschaft der Gesellschaft.* Suhrkamp Verlag, 7 edition, April 1998.

[77] Saunders Mac Lane. *Categories for the Working Mathematician.* Springer, 2nd edition, September 1998.

[78] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic. A First Introduction to Topos Theory.* Springer, Berlin, 2. nachdr. d. 1. a. von 1992. edition, 1992.

[79] George J. Mailath and Larry Samuelson. *Repeated Games and Reputations: Long-Run Relationships.* Oxford University Press, 2006.

[80] Grant Malcolm. Sheaves, objects, and distributed systems. *Electronic Notes in Theoretical Computer Science*, 225:3–19, January 2009.

[81] G. Masumoto and T. Ikegami. The lambda-game system: An approach to a meta-game. *Advances in Artificial Life*, page 695699, 2001.

[82] P. Milgrom and C. Shannon. Monotone comparative statics. *Econometrica*, page 157180, 1994.

[83] Paul Milgrom and John Roberts. Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica*, 58(6):pp. 1255–1277, 1990.

[84] Paul Milgrom and John Roberts. Comparing equilibria. *American Economic Review*, 84(3):441–59, 1994.

[85] Lawrence S. Moss and Ignacio D. Viglizzo. Harsanyi type spaces and final coalgebras constructed from satisfied theories. *Electronic Notes in Theoretical Computer Science*, 106:279–295, 2004.

[86] M. Mossio, G. Longo, and J. Stewart. A computable expression of closure to efficient causation. *Journal of theoretical biology*, 257(3):489498, 2009.

[87] Martin Osborne. *Course in Game Theory*. MIT Press, first edition edition, 1994.

[88] Bryan O'Sullivan, Donald Bruce Stewart, and John Goerzen. *Real World Haskell*. O'Reilly, 1 edition, December 2008.

[89] D Pavlovic. The continuum as a final coalgebra. *Theoretical Computer Science*, 280:105–122, May 2002.

[90] D. Pavlovic and M. H. Escardo. Calculus in coinductive form. In *Logic in Computer Science, 1998. Proceedings. Thirteenth Annual IEEE Symposium on*, page 408417, 2002.

[91] Dusko Pavlovic. Monoidal computer i: Basic computability by string diagrams. *arXiv:1208.5205*, August 2012.

[92] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publ Inc, 1988.

[93] Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2 edition, September 2009.

[94] Gordon Plotkin. A structural approach to operational semantics. *The Journal of Logic and Algebraic Programming*, 6061(0):17–139, July 2004.

[95] Horst Reichel. An approach to object semantics based on terminal coalgebras. *Mathematical Structures in Computer Science*, 5(02):129–152, 1995.

[96] Robert Rosen. *Life itself : a comprehensive inquiry into the nature, origin, and fabrication of life*. Columbia University Press, New York, 1991.

[97] Robert Rosen, Judith Rosen, John J. Kineman, and Mihai Nadin. *Anticipatory Systems: Philosophical, Mathematical, and Methodological Foundations*. Springer New York, 2nd ed. 2012 edition, January 2012.

[98] J. J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

[99] J. J. M. M Rutten. Coalgebraic foundations of linear systems. In *Proceedings of the 2nd international conference on Algebra and coalgebra in computer science*, pages 425–446, Berlin, Heidelberg, 2007. Springer-Verlag.

[100] Thomas J. Sargent. *Bounded Rationality in Macroeconomics*. Clarendon Press, November 1993.

[101] R. Keith Sawyer. *Social Emergence: Societies As Complex Systems*. Cambridge University Press, October 2005.

[102] D. Schmidt. Abstract interpretation from a topological perspective. *Static Analysis*, page 293308, 2009.

[103] D. A Schmidt. *Denotational Semantics: A Methodology for Language Development*. Allyn and Bacon, Inc., 1986.

[104] D. A. Schmidt. Abstract interpretation from a denotational-semantics perspective. *Electronic Notes in Theoretical Computer Science*, 249:1937, 2009.

[105] Dana Scott. Continuous lattices. In F. W. Lawvere, editor, *Toposes, Algebraic Geometry and Logic*, volume 274 of *Lecture Notes in Mathematics*, pages 97–136. Springer Berlin Heidelberg, 1972.

[106] Esther-Mirjam Sent. Sargent's symmetry saga: ontrological versus technical constraints. In Uskali Mki, editor, *The Economic World View: Studies in the Ontology of Economics*. Cambridge University Press, 2001.

[107] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, December 2008.

[108] Christopher A. Sims. Macroeconomics and reality. *Econometrica*, 48(1):1–48, January 1980.

[109] A. Sokolova. Probabilistic systems coalgebraically: A survey. *Theoretical Computer Science*, 2011.

[110] Kirk Sturtz. Quantifiers as adjoint in probability. *arXiv:1208.2938*, August 2012.

[111] Leigh Tesfatsion, Kenneth L. Judd, Michael D. Intriligator, and Kenneth J. Arrow. *Handbook of Computational Economics, Volume 2: Agent-Based Computational Economics*. North Holland, May 2006.

[112] Baltasar Trancon y Widemann and Michael Hauhs. Distributive-law semantics for cellular automata and agent-based models. 2011.

[113] Jan H. van Schuppen. System theory for system identification. *Journal of Econometrics*, 118(1-2):313–339, 2004.

[114] S. Vassilakis. Functional fixed points. Technical report, Stanford - Institute for Thoretical Economics, 1991.

[115] Spyros Vassilakis. Economic data types. Technical report, University of Pittsburgh, 1989.

[116] Spyros Vassilakis. Some economic applications of scott domains. *Mathematical Social Sciences*, 24(2-3):173–208, 1992.

[117] J. Steven Winrich. Self-reference and the incomplete structure of neoclassical economics. *Journal of Economic Issues*, 18(4):987–1005, 1984.

[118] N. S Yanofsky. A universal approach to self-referential paradoxes, incompleteness and fixed points. *Bulletin of Symbolic Logic*, 9(3):362386, 2003.