Socially Responsive, Environmentally Friendly Logic

Samson Abramsky

http://web.comlab.ox.ac.uk/oucl/work/samson.abramsky/

Oxford University Computing Laboratory

February 22, 2006

From the Manifesto

> "*Traditionally, logic has dealt with the zero-agent notion of truth and the one-agent notion of reasoning. In the last decades, research focus in logic shifted from these topics to the vast field of "interactive logic", encompassing logics of communication and interaction. The main applications of this move to n-agent notions are logical approaches to games and social software.* "

However, while there are certainly applications of multi-modal logics to reasoning *about* n-person games, the more intimate connections between Games and Logic in various forms of Game Semantics have all been based on 2-person games.

We are therefore led to ask

What kind of logic has a natural semantics in multi-agent games?

IF quantifiers and such: what about the *syntax*?

Another topic which has been studied extensively in recent years has been the logical aspects of games of imperfect information, starting with Henkin-style branching quantifiers, and continuing with the IF-logic of Hintikka and Sandu.

The Henkin quantifier:

$$\begin{pmatrix} \forall x & \exists y \\ \forall u & \exists v \end{pmatrix} A(x, y, u, v).$$

IF quantifier:

$$\forall x.\, \exists y.\, \forall u.\, \exists v/x.\, A(x, y, u, v).$$

How can we give a properly compositional *syntax* for such constructs? For example, how can we build up a formula with branching quantifiers piece by piece?

Overview

We shall:

- Give a natural logical syntax with a semantics in multi-agent games.
- Give a formal semantics in concurrent, multi-agent games which is compositional, mathematically precise, and surprisingly simple and elegant (iwmss).
- Extend the syntax and semantics to give a fully compositional account of IF and other quantifiers.
- Expose some surprising connections with several lines of work in Theoretical Computer Science.

Overview

We shall:

- Give a natural logical syntax with a semantics in multi-agent games.
- Give a formal semantics in concurrent, multi-agent games which is compositional, mathematically precise, and surprisingly simple and elegant (iwmss).
- Extend the syntax and semantics to give a fully compositional account of IF and other quantifiers.
- Expose some surprising connections with several lines of work in Theoretical Computer Science.

Things we *haven't* done (yet):

- Develop a proof theory for this logic
- Explore a wider range of (esp. multiplicative) connectives
- Develop applications

From 2-person to n-person games

The naturalness of 2-person games

The basic metaphor of Game Semantics of Logic is that the players stand for **Proponent** and **Opponent**, or **Verifier** and **Falsifier**, or (with Computer Science motivation) for **System** and **Environment**. This 2-agent view sets up a natural *duality*, which arises by interchanging the rôles of the two players. This duality is crucial in defining the key notion of *composition of strategies* in the Game Semantics developed in Computer Science. It stands as the Game-Semantical correlate of the logical notion of *polarity*, and the categorical notions of *domain* and *codomain*, and *co-* and *contra-variance*.

So this link between Logic and 2-person games runs deep, and should make us wary of facile generalization. It can be seen as having the same weight as the binary nature of composition in Categories, or of the Cut Rule in Logic. Are there good multi-ary generalizations of these notions?

Nevertheless . . . we shall put forward a simple system which seems to us to offer a natural generalization.

Background: 2-person games

As a starting point, we shall briefly review a Hintikka-style 'Game-Theoretical Semantics' of ordinary first-order logic, in negation normal form. Given a model $\mathcal{M}$, a game is assigned to each sentence as follows.

- **Literals** $A(a_1, \ldots, a_n)$, $\neg A(a_1, \ldots, a_n)$. The game is trivial in this case. Verifier wins if the literal is true in the model, and Falsifier wins otherwise.
- **Conjunction** $\varphi_1 \wedge \varphi_2$. The game $G(\varphi_1 \wedge \varphi_2)$ has a first move by Falsifier which chooses one of the sub-formulas φ_i, $i = 1, 2$. It then proceeds with $G(\varphi_i)$.
- **Disjunction** $G(\varphi_1 \vee \varphi_2)$ has a first move by Verifier which chooses one of the sub-formulas φ_i, $i = 1, 2$. It then proceeds with $G(\varphi_i)$.
- **Universal Quantification** $G(\forall x.\, \varphi)$ has a first move by Falsifier, which chooses an element a of $\mathcal{M}$. The game proceeds as $G(\varphi[a/x])$.
- **Existential Quantification** Dually, Verifier chooses the element.

The point of this interpretation is that $\mathcal{M} \models \varphi$ in the usual Tarskian sense if and only if Verifier has a winning strategy for $G(\varphi)$.

An Aside

Note that there is a very natural game-semantical interpretation of negation: $G(\neg\varphi)$ is the same game as $G(\varphi)$, but with the rôles of Verifier and Falsifier interchanged. But what kind of logic is this a negation for?

In fact, the above Game semantics should really be seen as applying to the *additive fragment of Linear Logic*, rather than to Classical Logic. Note in particular that it fails to yield a proper analysis of *implication*, surely the key logical connective.

Indeed, if we render $\varphi \longrightarrow \psi$ as $\neg\varphi \ \vee \ \psi$, then note that $G(\neg\varphi \ \vee \ \psi)$ does not allow for any flow of information between the antecedent and the consequent of the implication. At the very first step, one of $\neg\varphi$ or ψ is chosen, and the other is discarded. In order to have the possibility of such information flow, it is necessary for $G(\neg\varphi)$ and $G(\psi)$ to run *concurrently*. This takes us into the realm of the *multiplicative connectives* in the sense of Linear Logic. The game-theoretical interpretation of negation also belongs to the multiplicative level.

From 2-person to n-person games

We shall now describe a simple syntax which will carry a natural interpretation in n-agent games. We say that the resulting logic is 'socially responsive' in that it allows for the actions of multiple agents. We fix, once and for all, a set of agents $\mathcal{A}$, ranged over by $\alpha, \beta, \gamma, \ldots$

We introduce an $\mathcal{A}$-indexed family of binary connectives $\oplus_\alpha$, and an $\mathcal{A}$-indexed family of quantifiers Q_α. Thus we have a syntax:

$$\varphi \quad ::= \quad L \;\mid\; \varphi \oplus_\alpha \psi \;\mid\; Q_\alpha x.\, \varphi.$$

(Here L ranges over literals).

The intended interpretation of $\varphi \oplus_\alpha \psi$ is a game in which agent α initially chooses either φ or ψ, and then play proceeds in the game determined by the chosen sub-formula. Similarly, for $Q_\alpha x.\, \varphi$, α initially chooses an instance a for x, and play proceeds as for $\varphi[a/x]$.

n generalizes 2 — and negation?

2-person games as a special case

If we take $\mathcal{A} = \{V, F\}$, then we can make the following identifications:

$$\bigoplus_V = \vee, \qquad \bigoplus_F = \wedge, \qquad Q_V = \exists, \qquad Q_F = \forall.$$

Whither negation?

In 2-person Game Semantics, negation is interpreted as rôle interchange. This generalizes in the multi-agent setting to *rôle permutation*. Each permutation $\pi \in S(\mathcal{A})$ ($S(X)$ being the symmetric group on the set X) induces a logical operation $\hat{\pi}(\varphi)$ of permutation of the rôles of the agents in the game corresponding to A. In the 2-agent cases, there are two permutations in $S(\{V, F\})$, the identity (a 'no-op'), and the transposition $V \leftrightarrow F$, which corresponds exactly to the usual game-theoretical negation.

Other connectives

In the light of our remarks about the essentially *additive* character (in the sense of Linear Logic) of the connectives $\oplus_\alpha$ and their game-semantical interpretation, it is also natural to consider some multiplicative-style connectives. We shall introduce two very basic connectives of this kind, which will prove particularly useful for the compositional analysis of branching quantifiers:

1. **Parallel Composition**, $\varphi \| \psi$. The intended semantics is that $G(\varphi \| \psi)$ is the game in which play in $G(\varphi)$ and $G(\psi)$ proceeds in parallel.
2. **Sequential Composition**, $\varphi \cdot \psi$. Here we firstly play in $G(\varphi)$ to a conclusion, and then play in $G(\psi)$.

It will also be useful to introduce a constant $\mathbf{1}$ for a “neutral” or “vacuously true” proposition. The intended semantics is an empty game, in which nothing happens. Thus we should expect $\mathbf{1}$ to be a unit for both sequential and parallel composition.

The Syntax

Thus the syntax for our multi-agent logic $\mathcal{L}_{\mathcal{A}}$ stands as follows.

$$\varphi \quad ::= \quad \mathbf{1} \quad | \quad A \quad | \quad \varphi \oplus_\alpha \psi \quad | \quad Q_\alpha x.\, \varphi \quad | \quad \varphi \cdot \psi \quad | \quad \varphi \| \psi \quad | \quad \hat{\pi}(\varphi).$$

Here A ranges over atomic formulas, and $\pi \in S(\mathcal{A})$.

Quantifiers as Particles

The syntax of $\mathcal{L}_\mathcal{A}$ is more powerful than may first appear. Consider an idea which may seem strange at first, although it has some precursors in Dynamic Logic and the semantics of non-deterministic programming languages. Instead of treating quantifiers purely as prefixing operators $Q_\alpha x.\varphi$ in the usual way, we can consider them as stand-alone particles $Q_\alpha x \equiv Q_\alpha x.\mathbf{1}$. We should expect to have

$$(Q_\alpha x) \cdot \varphi \;\equiv\; (Q_\alpha x.\mathbf{1}) \cdot \varphi \;\equiv\; Q_\alpha x.(\mathbf{1} \cdot \varphi) \;\equiv\; Q_\alpha x.\,\varphi. \quad (1)$$

Branching Quantifiers

Thus this particle view of quantifiers does not lose any generality with respect to the usual syntax for quantification. But we can also express much more.

Example 1 *(2-agent case).*

$$[(\forall x \cdot \exists y) \parallel (\forall u \cdot \exists v)] \cdot A(x, y, u, v).$$

*This is the **Henkin quantifier**, expressed compositionally in the syntax of $\mathcal{L}_{\mathcal{A}}$.*

More generally:

Proposition 2 *Every partially-ordered quantifier prefix in which the partial order is a series-parallel poset can be expressed in the syntax of $\mathcal{L}_{\mathcal{A}}$.*

Semantics of $\mathcal{L}_{\mathcal{A}}$

Semantics of $\mathcal{L}_{\mathcal{A}}$

We shall now develop a semantics for this logic. This semantics will be built in two levels:

1. **Static Semantics of Formulas** To each formula $\varphi \in \mathcal{L}_{\mathcal{A}}$, we shall assign a form of game rich enough to allow for concurrent actions, and for rather general forms of temporal or causal dependency between moves. This assignment will be fully compositional.
2. **Dynamic Semantics** We then formulate a notion of *strategy* for these games. For each agent $\alpha \in \mathcal{A}$ there will be a notion of α-strategy. We shall show to build strategies for the games arising from formulas, compositionally from the strategies for the sub-formulas. We shall also define the key notion of how to evaluate *strategy profiles*, *i.e.* a choice of strategy for each agent, interacting with each other to reach a collective outcome. We shall find an elegant mathematical expression for this, *prima facie* very complicated, operational notion.

Concrete Data Structures as Concurrent Games

The structures we shall find convenient, and indeed very natural, to use as our formal representations of games were introduced by Gilles Kahn and Gordon Plotkin in 1975 (although their paper only appeared in a journal in 1993). They arose for Kahn and Plotkin in providing a representation theory for their notion of *concrete domains*. The term used by Kahn and Plotkin for these structures was *information matrices*; subsequently, they have usually been called *concrete data structures*, and we shall follow this latter terminology (although in some ways, the original name is more evocative in our context of use).

This is essentially the notion of *Concurrent Games* which we introduced in 1998, and used with Paul-Andre Mellies to prove *Full Completeness for MALL* (Proc. LiCS 1999). See also *Sequentiality vs Concurrency in Games and Logic*, MSCS 13, 2003.

Definition of CDS

A CDS is a structure

$$M = (C, V, D, \vdash)$$

- C is a set of *cells*, or ‘loci of decisions’—places where the agent can make their moves.
- V is a set of ‘values’, which label the choices which can be made by the agents from their menus of possible moves.
- $D \subseteq C \times V$ is a set of *decisions*, representing the possible choices which can be made for how to ‘fill’ a cell. (The more usual terminology for decisions is ‘events’; here we have followed Kahn and Plotkin’s original terminology, which is very apt for our purposes.
- The relation $\vdash \subseteq \mathcal{P}_{\mathsf{f}}(D) \times C$ is an *enabling relation* which determines the possible temporal flows of events in a CDS. ($\mathcal{P}_{\mathsf{f}}(X)$ is the set of *finite* subsets of X.)

States

A *state* of a CDS M is a set $s \subseteq D$ such that:

- s is a partial function, *i.e.* each cell is filled at most once.
- If $(c, v) \in s$, then there is a sequence of decisions

$$(c_1, v_1), \ldots, (c_k, v_k) = (c, v)$$

in s such that, for all j, $1 \leq j \leq k$, for some $\Gamma_j \subseteq \{(c_i, v_i) \mid 1 \leq i < j\}$:

$$\Gamma_j \vdash c_j.$$

This is a "causal well-foundedness" condition. (Kahn and Plotkin phrase it as: "c *has a proof in* x".) Note that, in order for there to be any non-empty configurations, there must be *initial cells* c_0 such that $\varnothing \vdash c_0$.

We write $\mathcal{D}(M)$ for the set of configurations, partially ordered by set inclusion. This is a concrete domain in the sense of Kahn and Plotkin.

$\mathcal{A}$-games

To obtain a structure to represent a multi-agent game, we shall consider a CDS M augmented with a *labelling map*

$$\lambda_M : C_M \longrightarrow \mathcal{A}$$

which indicates which agent is responsible for filling each cell. We call (M, λ_M) an $\mathcal{A}$-*game*.
We are now ready to specify the compositional assignment of an $\mathcal{A}$-game to each formula of $\mathcal{L}_\mathcal{A}$. We assume a set $\mathcal{I}$ which will be used as the domain of quantification. We shall use $\uplus$ for the disjoint union of sets.

Static Semantics

Static Semantics

- **Constant** 1. This is assigned the empty CDS $(\emptyset, \emptyset, \emptyset, \emptyset)$.
- **Atomic formulas** A. These are assigned the empty CDS $(\emptyset, \emptyset, \emptyset, \emptyset)$.

Static Semantics of Choice Connectives

Let

$$M = [\![\varphi]\!], \qquad N = [\![\psi]\!]$$

The CDS $M \oplus_\alpha N$ is defined as follows:

$$(C_M \uplus C_N \uplus \{c_0\}, V_M \cup V_N \cup \{1, 2\}, D_M \uplus D_N \uplus \{(c_0, 1), (c_0, 2)\}, \vdash_{M \oplus_\alpha N})$$

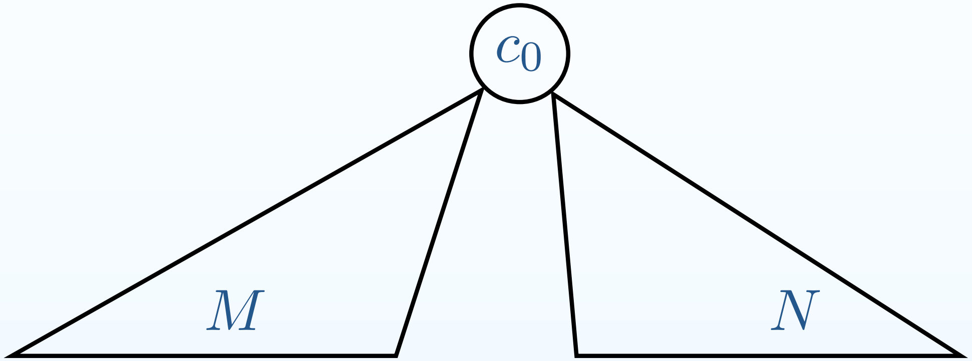
where

$$\begin{array}{lcl} \vdash_{M \oplus_\alpha N} c_0 & & \\ (c_0, 1), \Gamma \vdash_{M \oplus_\alpha N} c & \Longleftrightarrow & \Gamma \vdash_M c \\ (c_0, 2), \Gamma \vdash_{M \oplus_\alpha N} c & \Longleftrightarrow & \Gamma \vdash_N c. \end{array}$$

The labelling function $\lambda_{M \oplus_\alpha N}$:

$$\begin{array}{lcl} c_0 & \mapsto & \alpha \\ c & \mapsto & \lambda_M(c) \quad (c \in C_M) \\ c & \mapsto & \lambda_N(c) \quad (c \in C_N). \end{array}$$

In a picture



Initially, only the new cell c_0 is enabled. It can be filled with either of the values 1 or 2. If it is filled by 1, we can proceed as in $M = [\![\varphi]\!]$, while if it is filled with 2, we proceed as in $N = [\![\psi]\!]$. This makes the usual informal specification precise, in a rather general setting.

Quantifiers

Let $M = [\![\varphi]\!]$.

$$Q_\alpha(M) \;=\; (C_M \uplus \{c_0\}, V_M \cup \mathcal{I}, D_M \uplus (\{c_0\} \times \mathcal{I}), \vdash_{Q_\alpha(M)}).$$

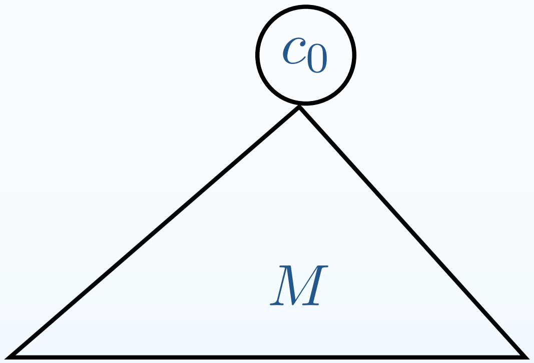
$$\vdash_{Q_\alpha(M)} c_0$$
$$(c_0, a), \Gamma \vdash_{Q_\alpha(M)} c \iff \Gamma \vdash_M c \quad (a \in \mathcal{I}).$$

This is a variant of the standard *lifting* construction on CDS.

The labelling function, $\lambda_{Q_\alpha(M)}$:

$$c_0 \mapsto \alpha, \qquad c \mapsto \lambda_M(c) \quad (c \in C_M).$$

In a Picture



Initially, only the new cell c_0 is enabled. It can be filled by α with any choice of individual a from the domain of quantification $\mathcal{I}$. Subsequently, we play as in M.

Parallel Composition

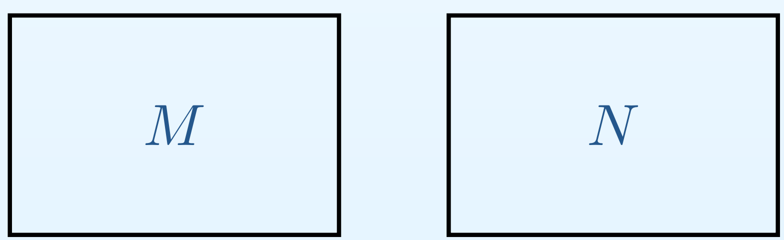
Let $M = [\![\varphi]\!]$, $N = [\![\psi]\!]$: we define

$$M \| N \quad = \quad (C_M \uplus C_N, V_M \uplus V_N, D_M \uplus D_N, \vdash_M \uplus \vdash_N).$$

The labelling function is defined by:

$$\lambda_{M \| N}(c) = \begin{cases} \lambda_M(c) & (c \in C_M) \\ \lambda_N(c) & (c \in C_N). \end{cases}$$

Pictorially:



Decisions in M and N can be made concurrently, with no causal or temporal constraints between them. This is the standard *product* construction on CDS.

Sequential Composition

We say that a configuration $s \in \mathcal{D}(M)$ is *maximal* if

$$\forall t \in \mathcal{D}(M)[s \subseteq t \;\Rightarrow\; s = t].$$

We write $\mathsf{Max}(M)$ for the set of maximal elements of $\mathcal{D}(M)$.
Let $M = [\![\varphi]\!]$, $N = [\![\psi]\!]$: we define

$$M \cdot N \quad = \quad (C_M \uplus C_N, V_M \uplus V_N, D_M \uplus D_N, \vdash_{M \cdot N}),$$

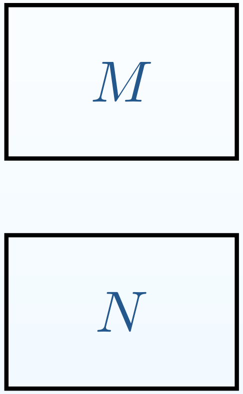
where

$$\Gamma \vdash_{M \cdot N} c \quad \Longleftrightarrow \quad \Gamma_M \vdash c \;\;\vee\;\; (\Gamma = x \cup \Delta \;\wedge\; x \in \mathsf{Max}(M) \;\wedge\; \Delta \vdash_N c).$$

The labelling function is defined by:

$$\lambda_{M \| N}(c) = \begin{cases} \lambda_M(c) & (c \in C_M) \\ \lambda_N(c) & (c \in C_N). \end{cases}$$

Pictorially



The idea is that firstly we reach a maximal configuration in M—a "complete play"—and then we can continue in N. Note that this construction makes sense for arbitrary CDS M, N. Even if M has infinite maximal configurations, the finitary nature of the enabling relation means that no events from N can occur in $M \cdot N$ following an infinite play in M.

Note that the difference between $M \| N$ and $M \cdot N$ is purely one of temporality of causality: when events can occur (or, in the alternative terminology, decisions can be made) relative to each other.

Role Switching

The CDS $[\![\hat{\pi}(\varphi)]\!]$ is *the same* as $M = [\![\varphi]\!]$. However:

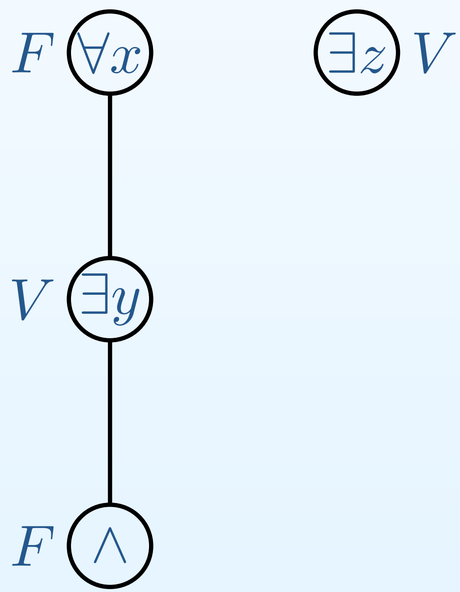
$$\lambda_{\hat{\pi}(M)} = \pi \circ \lambda_M.$$

Example

We take $\mathcal{A} = \{V, F\}$, and use standard notation for choice connectives and quantifiers. Consider the formula

$$\forall x.\exists y.[A(x,y) \ \wedge \ B(y)] \parallel \exists z.C(z).$$

The corresponding CDS has four cells:



In a maximal configuration of this CDS, these cells are all filled. If the $\forall x$ cell is filled with $a \in \mathcal{I}$, $\exists y$ with $b \in \mathcal{I}$, $\wedge$ with 1, and $\exists z$ with $c \in \mathcal{I}$, this corresponds to: $A(a,b) \parallel C(c)$.

Dynamic Semantics

Dynamic Semantics: Concurrent Strategies

We now turn to the task of defining a suitable notion of *strategy* for our games. We shall view a strategy for an agent α on the game M as a function $\sigma : \mathcal{D}(M) \rightarrow \mathcal{D}(M)$. The idea is that $\sigma(x)$ shows the moves which agent α would make, in the situation represented by the configuration x, when following the strategy represented by σ. Some formal features of σ follow immediately from this:

- (S1) Since past moves cannot be undone, we must have $x \subseteq \sigma(x)$, *i.e.* σ is *increasing*.
- (S2) If $(c, v) \in \sigma(x) \setminus x$, it must be the case that $\lambda_M(c) = \alpha$, since α is only able to make decisions in its own cells.

Further Conditions

We shall impose two further conditions. While not quite as compelling as the two above, they also have clear motivations.

- (S3) Idempotence: $\sigma(\sigma(x)) = \sigma(x)$. Since the only information in $\sigma(x)$ over and above what is in x is what σ put there, this is a reasonable normalizing assumption.
- (S4) Monotonicity: $x \subseteq y \;\Rightarrow\; \sigma(x) \subseteq \sigma(y)$. This condition reflects the idea that configurations only contain *positive* information.

Taking the conditions (S1), (S3), (S4) together says that σ is a *closure operator* on $\mathcal{D}(M)^{\top}$. A closure operator additionally satisfying (S2) is said to be an α-closure operator. We write $\mathsf{Cl}_{\alpha}(M)$ for the set of α-closure operators on $\mathcal{D}(M)^{\top}$.

Local *vs.* Global Definitions of Strategy Sets

The full specification of the game based on a CDS M will comprise a set of strategies $S_\alpha(M) \subseteq \mathsf{Cl}_\alpha(M)$ for each agent α. By limiting the set of strategies suitably, we can in effect impose constraints on the information available to agents.

There are two main compositional approaches to defining the strategy sets S_α.

1. We can define the strategy sets themselves directly, by induction on the construction on φ. This is the "global" approach. It is akin to realizability, and in general leads to strategy sets of high logical complexity.
2. We can use an indirect, more "local" approach, in which we add some structure to the underlying games, and use this to state conditions on strategies, usually phrased as conditions on individual plays or runs of strategies. $S_\alpha([\![\varphi]\!])$ is then defined to be the set of strategies in $\mathsf{Cl}_\alpha([\![\varphi]\!])$ satisfying these conditions. This has in fact been the main approach used in the Game Semantics of programming languages. However, this approach has not been developed for the kind of concurrent, multi-agent games being considered here.

Local conditions on strategies

The idea is to capture, as part of the structure of the underlying game, which information about the current configuration should be available to agent α when it makes a decision at cell c. This can be formalized by a function

$$\gamma_M : C_M \longrightarrow [\mathcal{D}(M) \longrightarrow \mathcal{D}(M)]$$

which for each cell c assigns a function $\gamma_M(c)$ on configurations. The idea is that, if $\lambda_M(c) = \alpha$, $\gamma_M(c)(x)$ restricts x to the part which should be visible to agent α, and on the basis of which he has to decide how to fill c. It follows that $\gamma_M(c)$ should be *decreasing*: $\gamma_M(c)(x) \subseteq x$. We add the assumptions of monotonicity and idempotence, with the same motivation as for strategies. It follows that $\gamma_M(c)$ is a *co-closure operator*.

The Local Condition

Notation

Remembering that a configuration $x \in \mathcal{D}(M)$ is a partial function, we write $x{\searrow}c$ to mean that x is defined at the cell c, or that "x fills c", as it is usually expressed. Also, we write C_M^α for the set of α-labelled cells in C_M.

Now, given such a function γ_M, we can define the strategy set $S_\alpha(M)$ as the set of all $\sigma \in \mathsf{Cl}_\alpha(M)$ satisfying the following condition:

$$\forall x \in \mathcal{D}(M). \forall c \in C_M^\alpha . [\sigma(x){\searrow}c \;\Rightarrow\; \sigma(\gamma_M(c)(x)){\searrow}c].$$

Defining the visibility function

- **Atomic formulas, and constant 1.** This case is trivial, since the set of cells is empty.
- **Choice connectives $\varphi \oplus_\alpha \psi$.** Let $M = [\![\varphi]\!]$, $N = [\![\psi]\!]$. We define $\gamma_{M \oplus_\alpha N}$ by:

$$\gamma_{M \oplus_\alpha N}(c)(x) = \begin{cases} x, & c = c_0 \\ \gamma_M(c)(x \setminus (c,1)), & (c \in C_M) \\ \gamma_M(c)(x \setminus (c,2)), & (c \in C_N). \end{cases}$$

- **Quantifiers $Q_\alpha.\varphi$.** Let $M = [\![\varphi]\!]$.

$$\begin{aligned} \gamma_{Q_\alpha(M)}(c_0)(x) &= x, \\ \lambda_{Q_\alpha(M)}(c)(\{(c_0,a)\} \uplus x) &= \{(c_0,a)\} \cup \gamma_M(c)(x) \;\; (c \in C_M). \end{aligned}$$

Thus the choice initially made by α to decide the value of the quantifier is visible to all the agents.

Defining the visibility function ctd

- **Parallel Composition** $\varphi \| \psi$. Let $M = [\![\varphi]\!]$, $N = [\![\psi]\!]$. We define $\gamma_{M\|N}$ by:

$$\gamma_{M\|N}(c)(x) = \begin{cases} \pi_M(x), & c \in C_M \\ \pi_N(x), & c \in C_N \end{cases}$$

Here π_M, π_N are the *projection functions*; e.g.

$$\pi_M(x) = \{(c, v) \in x \mid c \in C_M\}.$$

Thus the view at a cell in the sub-game M or N is what it would have been if we were playing only in that sub-game. This implements a complete block on information flow between the two sub-games. It can be seen as corresponding directly to the Linear Logic connective $\otimes$.

Defining the visibility function ctd

- **Sequential Composition** $\varphi \cdot \psi$. Let $M = [\![\varphi]\!]$, $N = [\![\psi]\!]$. We define $\gamma_{M \cdot N}$ by:

$$\gamma_{M \cdot N}(c)(x) = \begin{cases} \gamma_M(c)(x), & c \in C_M \\ \pi_M(x) \cup \gamma_N(c)(\pi_N(x)), & c \in C_N. \end{cases}$$

  Thus while we are playing in M, visibility is at it was in that sub-game. When we have finished a complete play y in M and start to play in N, we can see the whole completed play y, together with what is visible in the sub-game N.

- **Role Permutation** $\hat{\pi}(\varphi)$. We set $\gamma_{[\![\hat{\pi}(\varphi)]\!]} = \gamma_{[\![\varphi]\!]}$. The same information is available from each cell; but, for example, if agent α had more information available than agent β in M, that advantage will be transferred to β in $\hat{\pi}(M)$ if π interchanges α and β.

Evaluation of Strategy Profiles

Consider a CDS M with a strategy set S_α for each agent $\alpha \in \mathcal{A}$. A *strategy profile* is an $\mathcal{A}$-tuple

$$(\sigma_\alpha)_{\alpha \in \mathcal{A}} \in \prod_{\alpha \in \mathcal{A}} S_\alpha$$

which picks out a choice of strategy for each $\alpha \in \mathcal{A}$. The key operation in "bringing the semantics to life" is to define the outcome of playing these strategies off against each other. Given the concurrent nature of our games, and the complex forms of temporal dependency and information flow which may arise in them, it might seem that a formal definition of this operation will necessarily be highly complex and rather messy. In fact, our mathematical framework allows for a very elegant and clean definition. We shall use the notation $\langle \sigma_\alpha \rangle_{\alpha \in \mathcal{A}}$ for this operation. It maps strategy profiles to *configurations of* M.

Definition 3 *We define* $\langle \sigma_\alpha \rangle_{\alpha \in \mathcal{A}}$ *to be the* least common fixpoint *of the family of closure operators* $(\sigma_\alpha)_{\alpha \in \mathcal{A}}$*; i.e. the least element* x *of* $\mathcal{D}(M)$ *such that* $\sigma_\alpha(x) = x$ *for all* $\alpha \in \mathcal{A}$.

For Doubting Thomas

Proposition 4 *Any family of closure operators C on a complete lattice L has a common least fixpoint. In case the lattice has finite height, or $C = \{c_1, \ldots, c_n\}$ is finite and the closure operators in C are continuous (i.e. preserve directed joins) this common least fixpoint can be obtained constructively by the expression*

$$\bigvee_{k \in \omega} c^k(\bot), \qquad \text{where } c = c_1 \circ \cdots \circ c_k.$$

Any permutation of the order of the composition in defining c, or indeed any "schedule" which ensures that each closure operator is applied infinitely often, will lead to the same result.

Remark We pause to mention another connection with Theoretical Computer Science. Our use of closure operators as strategies, and the definition of the evaluation of strategy profiles as least common fixpoints, builds on ideas which arose originally in the semantics of *dataflow* and *concurrent constraint programming*.

Environmentally friendly logic

Towards environmentally friendly logic

It is, or should be, an aphorism of semantics that:

The key to compositionality is parameterization.

Choosing the parameters aright allows the meaning of expressions to be made sensitive to their contexts, and hence defined compositionally. While this principle could—in theory—be carried to the point of trivialization, in practice the identification of the right form of parameterization does usually represent some genuine insight into the structure at hand.
We shall now describe an approach to making the syntax of quantifier particles, including IF-quantifiers, fully compositional. This can then serve as a basis for a fully compositional account of valuations on outcomes.

Bidirectional Scoping

Note firstly a certain kind of quasi-duality between quantifiers and atomic formulas. Quantifiers $Q_\alpha x$ project the scope of x inwards over sequential compositions (but not across parallel compositions). Atomic formulas $A(x)$ depend on variables coming from an outer scope.

Now consider IF-quantifiers $\forall x/y$ which bind x, but also declare that it does *not* depend on an outer quantification over y. This is a peculiar binding construct, quite apart from its semantic interpretation. The bidirectional reach of the scope—inwards for x, outwards for y—is unusual, and difficult to make sense of in isolation from a given context of use. So in fact, it seems hard to give a decent compositional *syntax* for IF-quantifiers, before we even start to think about semantics.

Once again, there is work coming from Theoretical Computer Science which is suggestive: namely the π-calculus, with its *scope restriction* and *extrusion*. The action calculi subsequently developed by Milner are even more suggestive, although only certain features are relevant here.

Arities and Co-arities

We reformulate our logical syntax: each syntactic constituent will have an *arity* and a *co-arity*. Concretely, arities and co-arities are finite sets of variables. We write a syntactic expression as

$$X \xrightarrow{\varphi} Y$$

where X is the arity, and Y is the co-arity. The idea is that the arity specifies the variables that φ expects to have bound by its outer environment, while the co-arity represents variables that it binds in its inner environment.

The quantifier particle $Q_\alpha x$ can be described in these terms as

$$\varnothing \xrightarrow{Q_\alpha x} \{x\} \qquad (2)$$

or more generally as

$$X \xrightarrow{Q_\alpha x} X \uplus \{x\}.$$

An atom $A(x_1, \ldots, x_n)$ will have the form, indeed dual to (2):

$$\{x_1, \ldots, x_n\} \xrightarrow{A(x_1,\ldots,x_n)} \varnothing.$$

The Category $\mathbf{C}(\mathcal{L}_{\mathcal{A}})$

We specify "typed" versions of sequential and parallel composition:

$$\frac{X \xrightarrow{\varphi} Y \quad Y \xrightarrow{\psi} Z}{X \xrightarrow{\varphi \cdot \psi} Z} \qquad \frac{X_1 \xrightarrow{\varphi} Y_1 \quad X_2 \xrightarrow{\psi} Y_2}{X_1 \uplus X_2 \xrightarrow{\varphi \| \psi} Y_1 \uplus Y_2}$$

The constant $\mathbf{1}$ has the form

$$X \xrightarrow{\mathbf{1}} \varnothing$$

for any X.

We use a notion of *structural congruence*, as in the π-calculus and action calculi.

$$\varphi \cdot (\psi \cdot \theta) \equiv (\varphi \cdot \psi) \cdot \theta, \qquad \mathbf{1} \cdot \varphi \equiv \varphi \equiv \varphi \cdot \mathbf{1}$$

wherever these expressions make sense.

Thus we are in fact describing a *category* $\mathbf{C}(\mathcal{L}_{\mathcal{A}})$. The objects are the arities—"co-arities" are simply arities appearing as the codomains of arrows in the category. The arrows are the syntactic expressions modulo structural congruence; and the composition in the category is sequential composition.

Remark

One might have expected that $\mathbf{C}(\mathcal{L}_{\mathcal{A}})$ would form a *monoidal category*, with the functorial action of the tensor product given by parallel composition. However, this is not the case. The bifunctoriality of tensor:

$$(\varphi_1 \| \psi_1) \cdot (\varphi_2 \| \psi_2) \equiv (\varphi_1 \cdot \varphi_2) \| (\psi_1 \cdot \psi_2)$$

is violated by our partial information interpretation of sequential and parallel composition. For example,

$$(\forall x \| \forall y) \cdot (\exists u \| \exists v) \cdot A(x, y, u, v) \quad \not\equiv \quad ((\forall x \cdot \exists u) \| (\forall y \cdot \exists v)) \cdot A(x, y, u, v),$$

since in the first expression, the choice by Verifier of u can depend on Falsifier's choice of y, and the choice of v can depend on that of x, while these dependencies are excluded in the second expression.

Completing the Picture

To complete the picture: for the choice connectives, we have

$$\frac{X \xrightarrow{\varphi} \varnothing \quad X \xrightarrow{\psi} \varnothing}{X \xrightarrow{\varphi \oplus_\alpha \psi} \varnothing}$$

and for role interchange

$$\frac{X \xrightarrow{\varphi} Y}{X \xrightarrow{\hat{\pi}(\varphi)} Y}.$$

For the IF-quantifier we have

$$X \uplus \{y\} \xrightarrow{\forall x/y} X \uplus \{x\} \uplus \{y\},$$

which makes explicit the fact that y occurs free in $\forall x/y$.

The arrows in $\mathbf{C}(\mathcal{L}_\mathcal{A})$ will be the well-formed formulas (both open and "co-open") of our logic. In particular, the sentences or closed formulas will be the arrows of the form $\varnothing \xrightarrow{\varphi} \varnothing$.

Loose Ends

- There is an elegant extension of our semantics to the EF syntax, including IF quantifiers.
- Compositional definition of *valuation functions* on outcomes.
- Global definition of the strategy sets, comparison with local definition. (Issues of safety and liveness).

Further directions:

- Proof theory. Semantics of proofs, Curry-Howard etc.
- Extensions to fixed-point and second-order logic.
- Applications!

Paper to appear in Aho, Tuomo and Ahti-Veikko Pietarinen (eds.), *Truth and Games: Essays in Honour of Gabriel Sandu*, Helsinki: Acta Philosophica Fennica. Also available on my web page.