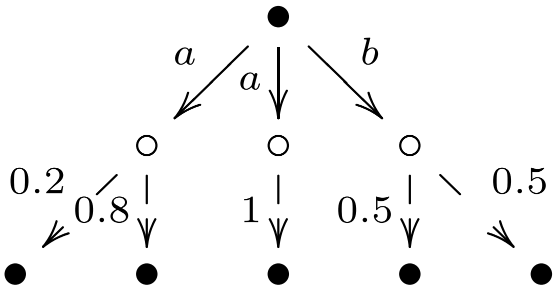
Coalgebraic Logics: Modalities Beyond Kripke Semantics

Part IV: Combining

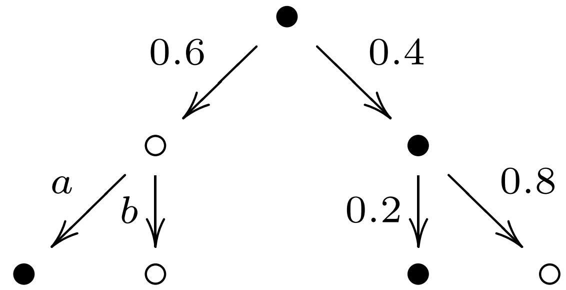
Dirk Pattinson, Imperial College London

Compositional Reasoning

Example. Combining Probabilities and Non-Determinism



Simple Segala Systems



Alternating Systems

Coalgebraic Interpretation

$$C \to \mathcal{P}^A(\mathsf{D}(C))$$

$$C \to \mathcal{P}^A(C) + \mathsf{D}(C)$$

Semantics of Combination. Functor Composition – ingredients represent features.

Logics for Combined Systems

Simple Segala Systems: $C \to \mathcal{P}^A(\mathsf{D}(C))$

$\mathcal{F}_n \ni \phi ::= \top \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \Box_a\psi$ (nondeterministic formulas; $\psi \in \mathcal{F}_u$, $a \in A$)

$\mathcal{F}_u \ni \psi ::= \top \mid \psi_1 \wedge \psi_2 \mid \neg\psi \mid L_p\phi$ (probabilistic formulas; $\phi \in \mathcal{F}_n$, $p \in [0,1] \cap \mathbb{Q}$).

Alternating Systems: $C \to \mathcal{P}^A(C) + \mathsf{D}(C)$

$\mathcal{F}_o \ni \rho ::= \top \mid \rho_1 \wedge \rho_2 \mid \neg\rho \mid \phi + \psi$ (alternating formulas; $\phi \in \mathcal{F}_u$, $\psi \in \mathcal{F}_n$)

$\mathcal{F}_u \ni \phi ::= \top \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid L_p\rho$ (probabilistic formulas; $\rho \in \mathcal{F}_o$, $p \in [0,1] \cap \mathbb{Q}$)

$\mathcal{F}_n \ni \psi ::= \top \mid \psi_1 \wedge \psi_2 \mid \neg\psi_2 \mid \Box_a\rho$ (nondeterministic formulas; $\rho \in \mathcal{F}_o$, $a \in A$)

Languages

- multisorted, one sort per feature

Language Composition

- mimics functor composition

Features and Gluings by Example

Features. Modalities and Axioms of the Building Blocks

- N, nondeterminism: unary modalities $\Box_a$ for $a \in A$
- U, uncertainty: unary modalities L_p for $p \in [0,1]$
- C, choice: binary modality $+$

Rule(schema) for C

$$\frac{(\bigwedge_{i=1}^{m} \alpha_i \to \bigvee_{j=1}^{n} \beta_j) : 1 \quad (\bigwedge_{i=1}^{m} \gamma_i \to \bigvee_{j=1}^{n} \delta_j) : 2}{\bigwedge_{i=1}^{m}(\alpha_i + \gamma_i) \to \bigvee_{j=1}^{n}(\beta_j + \delta_j)} \ (m, n \geq 0)$$

Gluings. Specification of Feature Composition

Simple Segala Systems

$$G_1(a) = \mathsf{N}(\mathsf{U}(a))$$

Alternating Systems

$$G_2(a) = \mathsf{C}(\mathsf{U}(a), \mathsf{N}(a))$$

Features and Gluings, Formally

Defn. An n-ary *feature* $\mathsf{F} = (\Lambda, \mathsf{R})$ comprises

- a set Λ of modal operators $L : i_1, \dots, i_k$ where $1 \leq i_1, \dots, i_k \leq n$ are argument sorts
- a set R of *one-step rules* of the form $(\phi_1; \dots; \phi_n)/\psi$,where
  - the ϕ_i are purely propositional
  - ψ is a clause over $\heartsuit(p_1, \dots, p_k)$ where $\heartsuit : i_1, \dots, i_k \in \Lambda$ and $p_j \in V_{i_j}$

Defn. *Feature expressions* are terms built with features as function symbols

$$t ::= a \mid \mathsf{F}(t_1, \dots, t_n) \qquad a \in \mathsf{S}, \mathsf{F} \in \Phi \ n\text{-ary.}$$

A *gluing* of Φ over S is a family $\mathsf{G} = (t_a)_{a \in \mathsf{S}}$ of feature expressions.

Semantics

Defn. A *structure* for an n-ary feature $\mathsf{F} = (\Lambda, \mathsf{R})$ consists of

- a functor $[\![\mathsf{F}]\!] : \mathsf{Set}^n \to \mathsf{Set}$
- an assignment of predicate liftings

$$[\![\heartsuit]\!]_X : \mathcal{P}(X_{i_1}) \times \cdots \times \mathcal{P}(X_{i_k}) \to \mathcal{P}([\![\mathsf{F}]\!] X)$$

indexed over $X = (X_1, \ldots, X_n) \in \mathsf{Set}^n$ to operators $\heartsuit : i_1, \ldots, i_k \in \Lambda$

Easy Consequence. Every gluing $\mathsf{G} = (t_a)_{a \in \mathsf{S}}$ gives rise to a functor

$$[\![\mathsf{G}]\!] : \mathsf{Set}^{\mathsf{S}} \to \mathsf{Set}^{\mathsf{S}}$$

by compositionality.

Idea. Given a gluing G, its models are S-sorted $[\![\mathsf{G}]\!]$-coalgebras

$$(C_s)_{s \in \mathsf{S}} \xrightarrow{(\gamma_s)_{s \in \mathsf{S}}} [\![\mathsf{G}]\!](C_s)_{s \in \mathsf{S}}$$

(previous definitions apply on a pre-component basis)

Examples

Nondeterminism over a set A of action as unary feature N with $\Box_a : 1$

$$[\![\mathsf{N}]\!]X = \mathcal{P}^A(X) \text{ with } [\![\Box_a]\!]_X(B) = \{f : A \to \mathcal{P}(X) \mid f(a) \subseteq B\}$$

(in the same way, all previous logics arise as features)

Choice as binary feature C with $+ : 1, 2$

$$[\![\mathsf{C}]\!](X, Y) = X + Y \text{ with } [\![+]\!]_{X,Y}(A, B) = A + B$$

Fusion as a binary feature P with $\pi_i : i$ for $i = 1, 2$ and

$$[\![\times]\!](X, Y) = X \times Y \text{ and } [\![\pi_i]\!]_{X_1,X_2}(A) = \{(x_1, x_2) \in X_1 \times X_2 \mid x_i \in A\}$$

Simple Segala Systems

$$[\![a \mapsto \mathsf{N}(\mathsf{U}(a))]\!]X = \mathcal{P}^A(\mathsf{D}(X))$$

Alternating Systems

$$[\![a \mapsto \mathsf{C}(\mathsf{U}(a), \mathsf{N}(a))]\!]X = \mathcal{P}^A(X) + \mathsf{D}(X)$$

The Logic of a Gluing

Types of a gluing G: the set $\mathsf{Types}(\mathsf{G})$ of proper subterms of a gluing

Example. The gluings

$$G_1 = (a \mapsto \mathsf{N}(\mathsf{U}(a))) \quad \text{and} \quad G_2 = (a \mapsto \mathsf{N}(b), b \mapsto \mathsf{U}(a))$$

(morally) have the same types $a, \mathsf{U}(a)$ and a, b, but different semantics:

$$[\![G_1]\!](X) = \mathcal{P}^A(\mathsf{D}(X)) \quad \text{and} \quad [\![G_2]\!](X, Y) = (\mathcal{P}^A(Y), \mathsf{D}(X))$$

Typed Formulas:

$$\frac{\phi_1 : s_1, \ldots, \phi_n : s_n}{\heartsuit(\phi_{i_1}, \ldots, \phi_{i_n}) : \mathsf{F}(s_1, \ldots, s_n)} \qquad \text{if } \mathsf{F}(s_1, \ldots, s_n) \in \mathsf{Types}(\mathsf{G})$$

$$\frac{\phi_1 : s_1, \ldots, \phi_n : s_n}{\heartsuit(\phi_{i_1}, \ldots, \phi_{i_n}) : a} \qquad \text{if } \mathsf{G}(a) = \mathsf{F}(a_1, \ldots, s_n)$$

Note. G_1 and G_2 have (morally) the same set of typed formulas.

Semantic Reconciliation

Suppose that G is a gluing and $(C, \gamma) \in \mathsf{Coalg}(\llbracket \mathsf{G} \rrbracket)$.

- for $s \in \mathsf{Types}(\mathsf{G})$, an *$s$-state* of C is an element of $\llbracket s \rrbracket(C)$

Example

- for $G_1 = (a \to \mathsf{N}(\mathsf{U}(a))$ and $(C, \gamma \to \mathcal{P}^A(\mathsf{D}(C))$, we have
  - a-states are the elements of C
  - $\mathsf{U}(a)$-states are the elements of $\llbracket \mathsf{U}(a) \rrbracket(C) = \mathsf{D}(C)$
- for $G_2 = (a \mapsto \mathsf{N}(b), b \mapsto \mathsf{U}(a))$ and $((C_a, C_b), \gamma_a : C_a \to \mathcal{P}^A(C_b), \gamma_b : C_b \to \mathsf{D}(C_a)$ we have
  - a-states are elements of $\llbracket a \rrbracket(C_a, C_b) = C_a$
  - b-states are elements of $\llbracket b \rrbracket(C_a, C_b) = C_b$

Satisfiability Problem. For a gluing G and $s \in \mathsf{Types}(\mathsf{G})$, is $\phi : s$ satisfiable in an s-state of some $(C, \gamma) \in \mathsf{Coalg}(\llbracket \mathsf{G} \rrbracket)$? valid in all s-states of all (C, γ)?

Flat Gluings

Recall. For the gluings

$$G_1 = (a \mapsto \mathsf{N}(\mathsf{U}(a))) \quad \text{and} \quad G_2 = (a \mapsto \mathsf{N}(b), b \mapsto \mathsf{U}(a))$$

we have the same satisfiability problem, but different semantics!

Flattening. Every Gluing $\mathsf{G} = (t_a)_{a \in \mathsf{S}}$ has a flattening $\mathsf{G}^\flat = (t'_{s'})_{s' \in \mathsf{S}'}$ where

$$\mathsf{S}' = \mathsf{Types}(G) \quad \text{and} \quad t'_{s'} = \begin{cases} t_s & \text{for } s \in \mathsf{S} \\ t'_{s'} = s & \text{otherwise} \end{cases}$$

Main Theorem. The satisfiability problems over G and $\mathsf{G}^\flat$ are equivalent.

Proof Sketch. “Padding with identities”, e.g. a satisfying model

$$\left(C \xrightarrow{\gamma} \mathcal{P}^A(\mathsf{D}(X))\right) \text{ yields } \begin{pmatrix} C \\ \mathsf{D}(C) \end{pmatrix} \xrightarrow{\gamma, id} \begin{pmatrix} \mathcal{P}^A(\mathsf{D}(C)) \\ \mathsf{D}(C) \end{pmatrix}$$

Benefit. Pick the “easiest” gluing to decide satsifiability.

Completeness and Decidability

Compositional Reasoning using a rule $(\phi_1; \ldots; \phi_n)/\psi$ is a rule of F

$$\frac{\vdash_{s_1} \phi_1\sigma, \ldots, \vdash_{s_n} \phi_n\sigma}{\vdash_{\mathsf{F}(s_1,\ldots,s_n)} \psi\sigma} \qquad \text{if } (\mathsf{F}(s_1, \ldots, s_n) \in \mathsf{Types}(\mathsf{G}))$$

$$\frac{\vdash_{s_1} \phi_1\sigma \ldots \vdash_{s_n} \phi_n\sigma}{\vdash_{a} \psi\sigma} \qquad \text{if } a \to \mathsf{F}(s_1, \ldots, s_n) \in \mathsf{G}$$

(Note the difference in the typing discipline)

Soundness. If all features are one-step sound, then $\mathsf{Coalg}([\![\mathsf{G}]\!]) \models_s \phi$ if $\vdash_s \phi$

Completeness If all features are one-step complete, then $\vdash_s \phi$ if $\mathsf{Coalg}([\![\mathsf{G}]\!]) \models_s \phi$.

Complexity. If all features are one-step sound, complete and NPMV, then satisfiability of $\phi : s$ is in PSPACE.

Proof Sketch. Straightforward generalisation of one-sorted results for flat gluings.

Example: Probabilistic Coalitions

Idea. Coalitions of agents can force *probabilities* of events.

Formulas. Two-sorted structure (coalitions/probabilities)

$$\mathcal{F}_c \ni \psi ::= \top \mid \rho_1 \wedge \rho_2 \mid \neg\rho \mid [C]\psi \qquad \text{(coalition formulas; } \psi \in \mathcal{F}_u\text{)}$$

$$\mathcal{F}_u \ni \psi ::= \top \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid L_p\phi \qquad \text{(probabilistic formulas; } \phi \in \mathcal{F}_c\text{)}$$

Example.

(coalition level) $[C]L_p\phi$ – C can ensure that $P(\phi) \geq p$

(probabilistic level) $M_p([C]\phi \vee [D]\phi)$ – $P($ C or D can force $\phi) \leq p$

Equivalent Semantics where $\mathsf{G}(X)$ are game frames / distributions over X

$$C \to \mathsf{G}(D)$$
$$D \to \mathsf{D}(C)$$

two-sorted models

$$C \to \mathsf{G} \circ \mathsf{D}(C)$$

Coalition Models

$$D \to \mathsf{D} \circ \mathsf{G}(D)$$

Probabilistic Models

Haskell Implementation

Gluings are a Haskell data type

```
Seg = HML <.> PML <.> S              Alt = HML <.> S <+> PML <
KK  = K <.> S <*> K <.> S            PC  = CL <.> PML <.> S
```

are constructed using `<.>` (composition), `<+>` (choice) and `<*>` (fusion)

Specific data type for **Flat gluings:** e.g. `flatten Seg` yields the flat gluing

```
[FlatUnary HML 1, FlatUnary PML 0]
```

corresponing to

$$(s_0 \to \mathsf{N}(s_1), s_1 \to \mathsf{U}(s_0))$$

Satisfiability checking by automatic generation of a solver for the language

```
data L0 = (propositional connectives) | HML0 Char L1
data L1 = (propositional connectives) |  PML1 Rational L0
```

Conclusions

Coalgebraic Semantics allows for

- uniform proofs of soundness, completeness, complexity
- compositionality – heterogeneous systems

Implementation.

- proof of concept – no optimisation
- slow in comparision with logic specific solvers
- but covers many new logics / combinations

In the pipeline. One-step logics are not of the most general variety . . .

- add fixpoints
- allow nested modalities
- optimise GML/PML/propositional reasoning
- interaction between features