Categories and Preorders in Value Iteration

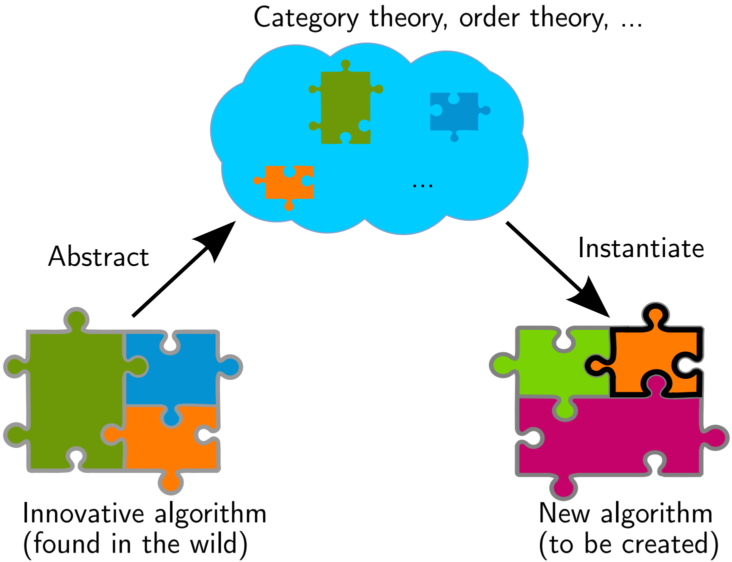
Fixed Points and Surrogate Models

Louis Rustenholz,
supervised by Ichiro Hasuo and Jérémy Dubut

30 August 2021

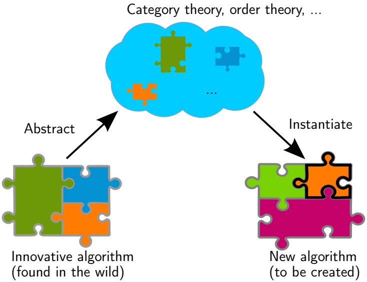
Introduction

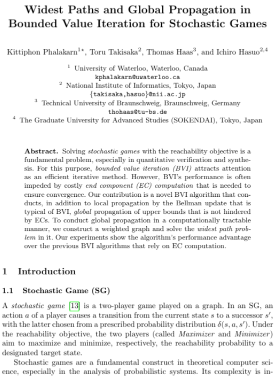
Goal: extract algorithmic ideas



In this work

Application domain	→	Model checking
Innovative algorithm	→	Solves reachability in Stochastic Games
Valuable ideas	→	Fixed points + Surrogate models





Widest Paths and Global Propagation in Bounded Value Iteration for Stochastic Games

Kittiphon Phalakarn[1*], Toru Takisaka[2], Thomas Haas[3], and Ichiro Hasuo[2,4]

[1] University of Waterloo, Waterloo, Canada
kphalakarn@uwaterloo.ca
[2] National Institute of Informatics, Tokyo, Japan
{takisaka,hasuo}@nii.ac.jp
[3] Technical University of Braunschweig, Braunschweig, Germany
thohaas@tu-bs.de
[4] The Graduate University for Advanced Studies (SOKENDAI), Tokyo, Japan

Abstract. Solving *stochastic games* with the reachability objective is a fundamental problem, especially in quantitative verification and synthesis. For this purpose, *bounded value iteration (BVI)* attracts attention as an efficient iterative method. However, BVI's performance is often impeded by costly *end component (EC) computation* that is needed to ensure convergence. Our contribution is a novel BVI algorithm that conducts, in addition to local propagation by the Bellman update that is typical of BVI, *global* propagation of upper bounds that is not hindered by ECs. To conduct global propagation in a computationally tractable manner, we construct a weighted graph and solve the *widest path problem* in it. Our experiments show the algorithm's performance advantage over the previous BVI algorithms that rely on EC computation.

1 Introduction

1.1 Stochastic Game (SG)

A *stochastic game* [13] is a two-player game played on a graph. In an SG, an action a of a player causes a transition from the current state s to a successor s', with the latter chosen from a prescribed probability distribution $\delta(s, a, s')$. Under the reachability objective, the two players (called *Maximizer* and *Minimizer*) aim to maximize and minimize, respectively, the reachability probability to a designated target state.

Stochastic games are a fundamental construct in theoretical computer science, especially in the analysis of probabilistic systems. Its complexity is in-

Contributions: axiomatise (Bounded) Value Iteration

VI

$$\begin{array}{ccc} \textbf{Preord} & \xrightarrow{T_l} & \textbf{Preord} \\ \downarrow U & & \downarrow U \\ \textbf{Set} & \xrightarrow{T} & \textbf{Set} \end{array}$$

Leaf structures

$$\frac{\Omega : \mathbf{CLat},\ A : \mathbf{Set},\ \tau : DA \to \Omega}{(\Delta_A, \Omega, \tau) : \mathcal{F}_1}\textit{(Constant ppt)} \qquad \frac{\Omega : \mathbf{CLat},\ A : \mathbf{Set},\ \tau : \mathrm{Hom}(A, \Omega) \to \Omega}{(\mathrm{Hom}(A, -), \Omega, \tau) : \mathcal{F}_1}\textit{(Reader ppt)}$$

$$\frac{\Omega : \mathbf{CLat}}{(\mathcal{P}_{\mathrm{em}}, \Omega, \sup) : \mathcal{F}_1}\textit{(Max game)} \qquad \frac{\Omega : \mathbf{CLat}}{(\mathcal{P}_{\mathrm{em}}, \Omega, \inf) : \mathcal{F}_1}\textit{(Min game)} \qquad \frac{}{(\mathcal{D}_{\mathrm{tl}}, [0,1], \mathbb{E}) : \mathcal{F}_1}\textit{(Random process)}$$

$$\frac{(W, +) : \mathbf{Monoid},\ (W, \leq_+) : \mathbf{CLat},\ \tau : (\mathrm{Hom}_{\mathrm{fin}}(W, W), \preceq_{tl}) \to (W, \leq_+)}{(\mathrm{Hom}_{\mathrm{fin}}(-, W), W, \tau) : \mathcal{F}_1}\textit{(Generalised random process)}$$

$$\frac{\mathcal{T}_a = (T_a, \Omega_a, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega_b, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a \times \mathcal{T}_b := (T_a \times T_b, \Omega_a \times \Omega_b, \tau_a \times \tau_b) : \mathcal{F}_1}\textit{(Product)} \qquad \frac{\mathcal{T}_a = (T_a, \Omega, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a + \mathcal{T}_b := (T_a + T_b, \Omega, \tau_a + \tau_b) : \mathcal{F}_1}\textit{(Coproduct)}$$

$$\frac{\mathcal{T}_a = (T_a, \Omega, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a \circ \mathcal{T}_b := (T_a \circ T_b, \Omega, \tau_a \circ T_a \tau_b) : \mathcal{F}_1}\textit{(Composition)} \qquad \frac{\mathcal{T} = (T, \Omega, \tau) : \mathcal{F}_1,\ \tilde{\tau} : T\Omega \to \Omega}{(T, \Omega, \tilde{\tau}) : \mathcal{F}_1}\textit{(Modality replacement)}$$

BVI + (some) Surrogate Models

$$\begin{array}{ccc} & & \textbf{Preord} \\ & \nearrow^{T_s} & \downarrow U \\ \textbf{Set} & \xrightarrow{T} & \textbf{Set} \end{array}$$

Shape structures

$MDP \lhd SG$

Connections

$$V(g) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V(g_\alpha)$$

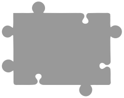
Plan

1. Research approach
2. Representation of model checking problems
3. Axiomatisation of VI technique
4. Axiomatisation of surrogate model technique
5. Future work

Approach and Context

Goal: extract algorithmic ideas – Research approach

“Semantics applied to algorithmic concepts”



Innovative algorithm
(found in the wild)

Goal: extract algorithmic ideas – Research approach

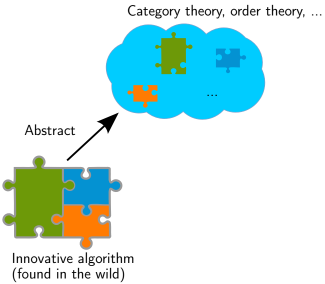
"Semantics applied to algorithmic concepts"

Innovative algorithm
(found in the wild)

- Understand what makes the algorithm work

Goal: extract algorithmic ideas – Research approach

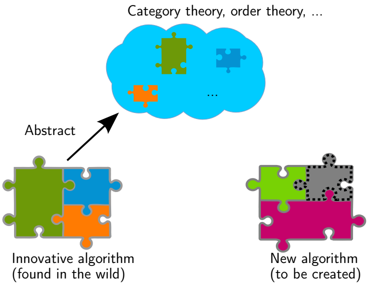
"Semantics applied to algorithmic concepts"



- Generalise the problem solved by the algorithm
- Understand what makes the algorithm work
- Axiomatise those properties, create new abstract structures
- Prove theorems about those abstract structures

Goal: extract algorithmic ideas – Research approach

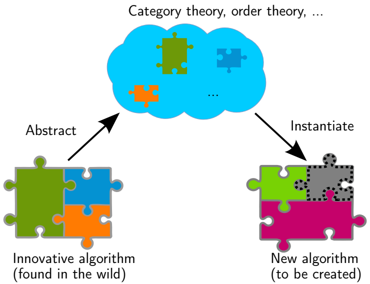
"Semantics applied to algorithmic concepts"



- Generalise the problem solved by the algorithm
- Understand what makes the algorithm work
- Axiomatise those properties, create new abstract structures
- Prove theorems about those abstract structures

Goal: extract algorithmic ideas – Research approach

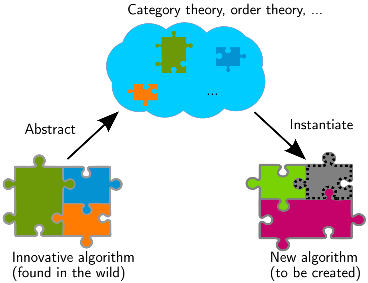
"Semantics applied to algorithmic concepts"



- Generalise the problem solved by the algorithm
- Understand what makes the algorithm work
- Axiomatise those properties, create new abstract structures
- Prove theorems about those abstract structures
- Instantiate (have in mind the contexts in which we want to)

Goal: extract algorithmic ideas – Research approach

"Semantics applied to algorithmic concepts"

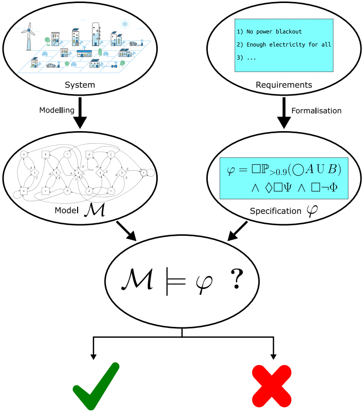


- Generalise the problem solved by the algorithm
- Understand what makes the algorithm work
- Axiomatise those properties, create new abstract structures
- Prove theorems about those abstract structures
- Instantiate (have in mind the contexts in which we want to)

Broader Context: Model Checking

Processes

Logic

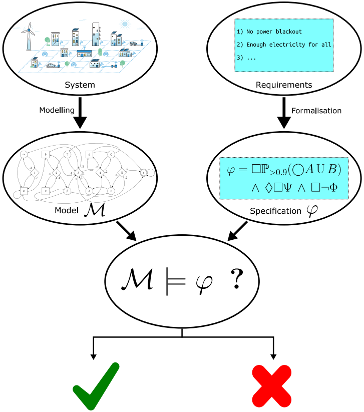


Here, $\Omega = \mathrm{Bool}$.
$\Omega = [0, 1]$, $\Omega = \mathbb{N}_\infty$, etc.

Model Checking: Adopt the right formalism

Processes

- New process types ?



Logic

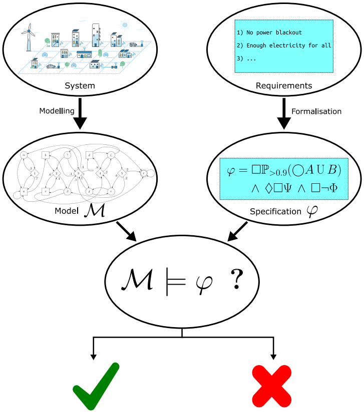
Here, $\Omega = \mathrm{Bool}$.

$\Omega = [0,1]$, $\Omega = \mathbb{N}_\infty$, etc.

Model Checking: Adopt the right formalism

Processes

- New process types ?
- Generalise with categories, coalgebra [Jacobs & Rutten, 90s]



Logic

Here, $\Omega = \mathrm{Bool}$.

$\Omega = [0, 1]$, $\Omega = \mathbb{N}_\infty$, etc.

Model Checking: Adopt the right formalism

Processes

- New process types ?
- Generalise with categories, coalgebra [Jacobs & Rutten, 90s]

Process type

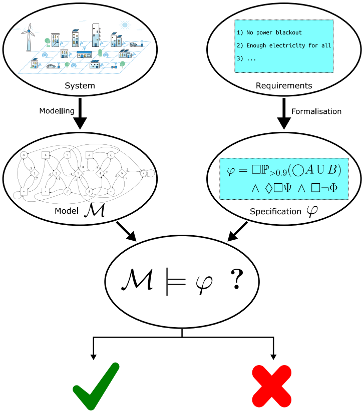
$\updownarrow$

$T : \mathbf{Set} \to \mathbf{Set}$

Process

$\updownarrow$

$\mathcal{M} : X \to TX$



Logic

Here, $\Omega = \mathrm{Bool}$.
$\Omega = [0,1]$, $\Omega = \mathbb{N}_\infty$, etc.

Model Checking: Adopt the right formalism

Processes

- New process types ?
- Generalise with categories, coalgebra [Jacobs & Rutten, 90s]

Process type

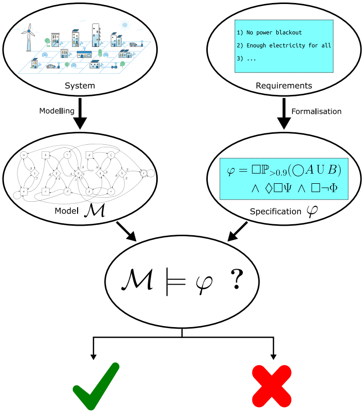
$\updownarrow$

$T : \mathbf{Set} \to \mathbf{Set}$

Process

$\updownarrow$

$\mathcal{M} : X \to TX$



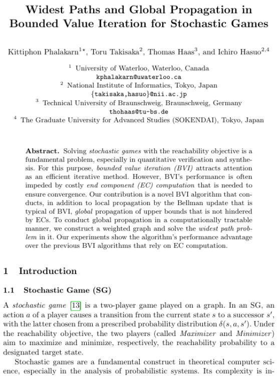
Here, $\Omega = \mathrm{Bool}$.

$\Omega = [0,1]$, $\Omega = \mathbb{N}_\infty$, etc.

Logic

- What should we do when algorithms manipulate predicate explicitly ?
- Semantics ?
  - Fibrations
  - Predicate transformers
  - Domain theory

Base algorithm [CAV20]



Widest Paths and Global Propagation in Bounded Value Iteration for Stochastic Games

Kittiphon Phalakarn[1*], Toru Takisaka[2], Thomas Haas[3], and Ichiro Hasuo[2,4]

[1] University of Waterloo, Waterloo, Canada
kphalakarn@uwaterloo.ca
[2] National Institute of Informatics, Tokyo, Japan
{takisaka,hasuo}@nii.ac.jp
[3] Technical University of Braunschweig, Braunschweig, Germany
thohaas@tu-bs.de
[4] The Graduate University for Advanced Studies (SOKENDAI), Tokyo, Japan

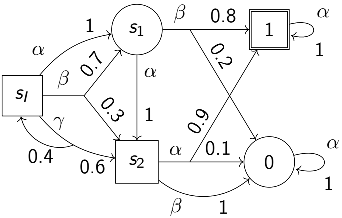
Abstract. Solving *stochastic games* with the reachability objective is a fundamental problem, especially in quantitative verification and synthesis. For this purpose, *bounded value iteration (BVI)* attracts attention as an efficient iterative method. However, BVI's performance is often impeded by costly *end component (EC) computation* that is needed to ensure convergence. Our contribution is a novel BVI algorithm that conducts, in addition to local propagation by the Bellman update that is typical of BVI, *global* propagation of upper bounds that is not hindered by ECs. To conduct global propagation in a computationally tractable manner, we construct a weighted graph and solve the *widest path problem* in it. Our experiments show the algorithm's performance advantage over the previous BVI algorithms that rely on EC computation.

1 Introduction

1.1 Stochastic Game (SG)

A *stochastic game* [13] is a two-player game played on a graph. In an SG, an action a of a player causes a transition from the current state s to a successor s', with the latter chosen from a prescribed probability distribution $\delta(s, a, s')$. Under the reachability objective, the two players (called *Maximizer* and *Minimizer*) aim to maximize and minimize, respectively, the reachability probability to a designated target state.

Stochastic games are a fundamental construct in theoretical computer science, especially in the analysis of probabilistic systems. Its complexity is in-



2.5-player game

$$\mathbb{P}(\Diamond 1) = ?$$

A novel algorithm for reachability in Stochastic Games

Base algorithm – Three main ingredients

- Value Iteration (VI) [Bellman, 1957]

- 2.5-player game (SG) $\rightarrow$ 1.5-player game (MDP)
- 1.5-player game (MDP) $\rightarrow$ Weighted Graph (WG)

Base algorithm – Three main ingredients

- Value Iteration (VI) [Bellman, 1957]

$$\left(\begin{array}{c} \text{Compute a value function } V(g) \text{ by iterating an operator } \mathcal{B} \\ V(g) = \sup_{\alpha:\mathbf{Ord}} \mathcal{B}^{\alpha}(\bot) \end{array} \right)$$

$\uparrow$ Fixed point characterisation

- 2.5-player game (SG) $\rightarrow$ 1.5-player game (MDP)

$\updownarrow$ Surrogate models

- 1.5-player game (MDP) $\rightarrow$ Weighted Graph (WG)

Base algorithm – Three main ingredients

- Value Iteration (VI) [Bellman, 1957]

$$\left(\begin{array}{c} \text{Compute a value function } V(g) \text{ by iterating an operator } \mathcal{B} \\ V(g) = \sup_{\alpha:\mathbf{Ord}} \mathcal{B}^{\alpha}(\bot) \end{array} \right)$$

↑ Fixed point characterisation

- 2.5-player game (SG) → 1.5-player game (MDP)

↕ Surrogate models

- 1.5-player game (MDP) → Weighted Graph (WG)

Incremental and Approximative

Converge to the solution

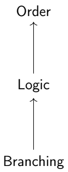
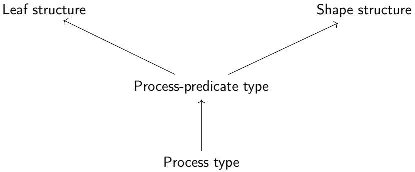
Approximate the solution
to a hard problem
by solutions to simple problems

Theoretical ingredients

- Category theory, coalgebra, enriched categories
- Order theory
- Semantics, weakest precondition semantics, domain theory, etc.

Modelisation

Contribution: definition of structures

Nature of the structures introduced	Our main definitions
	

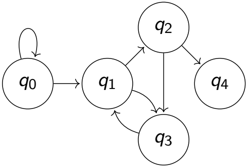
- **Process types** to model processes
  (e.g. graphs with accepting states)
- **Process predicate types** to model problems
  (e.g. reachability of accepting state in graphs)
- **Leaf structure** to axiomatise Value Iteration
- **Shape structure** to axiomatise Surrogate Models

Model processes – From [Jacobs & Rutten, 90s]

Definition (Coalgebra)

A *coalgebra* is a $g : X \to TX$, where

- X : **Set** is the *state space*,
- T : **Set** $\to$ **Set** is the *process type*.



$$
\begin{aligned}
g : \{q_0, q_1, q_2, q_3, q_4\} &\to \mathcal{P}\{q_0, q_1, q_2, q_3, q_4\} \\
q_0 &\mapsto \{q_0, q_1\} \\
q_1 &\mapsto \{q_2, q_3\} \\
q_2 &\mapsto \{q_3, q_4\} \\
q_3 &\mapsto \{q_1\} \\
q_4 &\mapsto \varnothing
\end{aligned}
$$

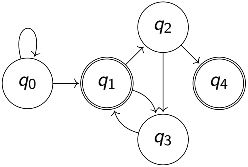
$\mathcal{P}$ as the process type of graphs

Model processes – From [Jacobs & Rutten, 90s]

Definition (Coalgebra)

A *coalgebra* is a $g : X \to TX$, where

- X : **Set** is the *state space*,
- T : **Set** $\to$ **Set** is the *process type*.



$$
\begin{aligned}
g : \{q_0, q_1, q_2, q_3, q_4\} &\to 2 \times \mathcal{P}\{q_0, q_1, q_2, q_3, q_4\} \\
q_0 &\mapsto (\bot, \{q_0, q_1\}) \\
q_1 &\mapsto (\top, \{q_2, q_3\}) \\
q_2 &\mapsto (\bot, \{q_3, q_4\}) \\
q_3 &\mapsto (\bot, \{q_1\}) \\
q_4 &\mapsto (\top, \varnothing)
\end{aligned}
$$

$2 \times \mathcal{P}$ as the process type of graphs with accepting states

Model processes – From [Jacobs & Rutten, 90s]

Definition (Coalgebra)

A *coalgebra* is a $g : X \to TX$, where

- X : **Set** is the *state space*,
- T : **Set** $\to$ **Set** is the *process type*.

Very flexible!

- $\mathcal{D}$ for Markov chains
- $2 \times (1 + (-))^{\Sigma}$ for Deterministic Automata
- $2 \times \mathrm{Hom}(\Sigma, 1 + (-))$ for Non-deterministic Automata
- $\{\bot, \top\} \times \{\Box, \bigcirc\} \times \mathcal{PD}(-)$ for Stochastic Games
- etc.

Model problems – From [Hasuo, 15]

Definition (Process predicate type)

A ppt is a tuple $\mathcal{T} = (T, \Omega, \tau)$, where

- T : **Set** $\rightarrow$ **Set** is a *process type*,
- Ω : **Set** is a *truth object*,
- $\tau : T\Omega \rightarrow \Omega$ is a *modality*.

Definition (Weakest precondition transformer)

Given $\mathcal{T} = (T, \Omega, \tau)$, $g : X \rightarrow TX$,
we introduce the predicate transformer

$$\begin{aligned} g^*_\tau : \Omega^X &\rightarrow \Omega^X \\ p &\mapsto \tau \circ Tp \circ g \end{aligned}$$

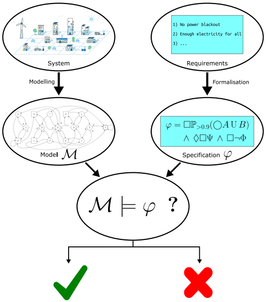
i.e. $g^*_\tau(p) : X \xrightarrow{g} TX \xrightarrow{Tp} T\Omega \xrightarrow{\tau} \Omega$.

Model problems – Examples of ppt

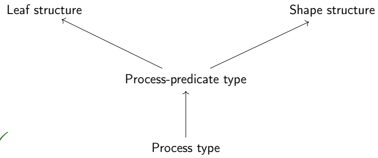
Problem	Process predicate type
$E\lozenge 1$ in NTS	$\left(2 \times \mathcal{P}(-), \mathrm{Bool}, \top + \sup\right)$
$A\lozenge 1$ in NTS	$\left(2 \times \mathcal{P}(-), \mathrm{Bool}, \top + \inf\right)$
$\mathbb{P}(\lozenge 1)$ in MC	$\left(2 \times \mathcal{D}(-), [0,1], 1 + \mathbb{E}\right)$
$\mathbb{P}(\lozenge 1)$ in MDP	$\left(2 \times \mathcal{PD}(-), [0,1], 1 + \sup \circ \mathbb{E}\right)$
$\mathbb{P}(\lozenge 1)$ in SG	$\left(2 \times 2 \times \mathcal{PD}(-), [0,1], (1 + \sup \circ \mathbb{E}) + (1 + \inf \circ \mathbb{E})\right)$

Also suited to graph problems.

Model model-checking – Conclusion



- Order ?
- Logic ✓
- Branching ✓



Type of model	$\longleftrightarrow$	Process type $T : \mathbf{Set} \to \mathbf{Set}$
Specific model	$\longleftrightarrow$	Coalgebra $g : X \to TX$
Type of model-checking problem	$\longleftrightarrow$	Ppt (T, Ω, τ)

Fixed points

An example: Bellman Operator as Weakest Precondition

$$\mathbb{P}(\Diamond 1) \text{ in SG} \quad \longleftrightarrow \quad (2 \times 2 \times \mathcal{PD}(-), [0,1], \beta)$$

Example

With the *Bellman modality*

$$\begin{aligned} \beta : 2 \times 2 \times \mathcal{PD}[0,1] &\longrightarrow [0,1] \\ (-, \top, -) &\longmapsto 1 \\ (\square, \bot, t) &\longmapsto \sup_{d \in t} \mathbb{E}(d) \\ (\bigcirc, \bot, t) &\longmapsto \inf_{d \in t} \mathbb{E}(d) \end{aligned}$$

For any Stochastic Game $g : X \to 2 \times 2 \times \mathcal{PD}X$,
The weakest precondition transformer g^*_β is simply the Bellman Operator!

$$\mathcal{B} = g^*_\beta : [0,1]^X \longrightarrow [0,1]^X$$

$$p \longmapsto \left(x \longmapsto \begin{cases} 1 & \text{if } x \text{ is a goal state} \\ \sup_a \sum_y \delta(x,a,y)p(y) & \text{if } x \text{ belongs to Maximizer} \\ \inf_a \sum_y \delta(x,a,y)p(y) & \text{if } x \text{ belongs to Minimizer} \end{cases} \right)$$

An example: Fixed point characterisation, Value Iteration

- $\mathbb{P}(\Diamond 1)$ in SG $\quad \longleftrightarrow \quad (2 \times 2 \times \mathcal{PD}(-), [0,1], \beta)$
- $g^*_\beta : [0,1]^X \to [0,1]^X$

Take the usual $([0,1], \leq)$: **CLat**, pointwise order on $[0,1]^X$. Then,

$$\mathbb{P}(\Diamond 1) = \mathrm{lfp}\, g^*_\beta$$

Using Knaster-Tarski / Cousot-Cousot, since g^*_β is monotone,

$$\mathbb{P}(\Diamond 1) = \sup_{\alpha:\mathbf{Ord}} (g^*_\beta)^\alpha(\bot).$$

This can be generalised!

Leaves, fixed points, VI algorithms (1) – Generalise

Ppt $\mathcal{T} = (T, \Omega, \tau)$, coalgebra $g : X \to TX$.

- VI algorithms compute the *value function* $V(g)$ of g in $\mathcal{T}$.
- We want to define V generally by saying

$$V(g) := \operatorname{lfp} g^*_\tau = \sup_{\alpha:\mathbf{Ord}} (g^*_\tau)^\alpha(\bot)$$

Leaves, fixed points, VI algorithms (1) – Generalise

Ppt $\mathcal{T} = (T, \Omega, \tau)$, coalgebra $g : X \to TX$.

- VI algorithms compute the *value function* $V(g)$ of g in $\mathcal{T}$.
- We want to define V generally by saying

$$V(g) := \operatorname{lfp} g^*_\tau = \sup_{\alpha:\mathbf{Ord}} (g^*_\tau)^\alpha(\bot)$$

- For that, make Ω a complete lattice, and g^*_τ monotone.

Leaves, fixed points, VI algorithms (1) – Generalise

Ppt $\mathcal{T} = (T, \Omega, \tau)$, coalgebra $g : X \to TX$.

- VI algorithms compute the *value function* $V(g)$ of g in $\mathcal{T}$.
- We want to define V generally by saying

$$V(g) := \operatorname{lfp} g^*_\tau = \sup_{\alpha:\textbf{Ord}} (g^*_\tau)^\alpha(\bot)$$

- For that, make Ω a complete lattice, and g^*_τ monotone.
- In the next slide, we provide a theorem to check monotonicity easily, looking only at T and τ (not g).

"Monotone liftings provide fixed points"

Leaves, fixed points, VI algorithms (2) – Axiomatise

$$\text{Ppt } \mathcal{T} = (T, \Omega, \tau), \quad \Omega_l = (\Omega, \leq) : \mathbf{CLat}.$$

Theorem (Leaf structure)

To construct a fixed point theory (i.e. to ensure that each g_τ^* is monotone), it is sufficient to construct a lifting

$$\begin{array}{ccc} \mathbf{Preord} & \xrightarrow{T_l} & \mathbf{Preord} \\ \downarrow U & & \downarrow U \\ \mathbf{Set} & \xrightarrow{T} & \mathbf{Set} \end{array}$$

*where T_l : **Preord** $\to$ **Preord** is enriched over **Preord** and $\tau : T_l(\Omega, \leq) \to (\Omega, \leq)$ is monotone.*

"Monotone liftings provide fixed points"

Definition (Leaf structure)

In such conditions, (T_l, Ω_l, τ) is called a ppt with leaf structure.

Leaves, fixed points, VI algorithms (3) – Recipe

Theorem (Family of examples)

The family $\mathcal{F}_1$ is a family of ppt with leaf structure.

$$\frac{\Omega : \mathbf{CLat},\ A : \mathbf{Set},\ \tau : DA \to \Omega}{(\Delta_A, \Omega, \tau) : \mathcal{F}_1}\textit{(Constant ppt)} \qquad \frac{\Omega : \mathbf{CLat},\ A : \mathbf{Set},\ \tau : \mathrm{Hom}(A, \Omega) \to \Omega}{(\mathrm{Hom}(A, -), \Omega, \tau) : \mathcal{F}_1}\textit{(Reader ppt)}$$

$$\frac{\Omega : \mathbf{CLat}}{(\mathcal{P}_{\mathrm{em}}, \Omega, \sup) : \mathcal{F}_1}\textit{(Max game)} \qquad \frac{\Omega : \mathbf{CLat}}{(\mathcal{P}_{\mathrm{em}}, \Omega, \inf) : \mathcal{F}_1}\textit{(Min game)} \qquad \frac{}{(\mathcal{D}_{\mathrm{tl}}, [0,1], \mathbb{E}) : \mathcal{F}_1}\textit{(Random process)}$$

$$\frac{(W, +) : \mathbf{Monoid},\ (W, \leq_+) : \mathbf{CLat},\ \tau : (\mathrm{Hom}_{\mathrm{fin}}(W, W), \preceq_{tl}) \to (W, \leq_+)}{(\mathrm{Hom}_{\mathrm{fin}}(-, W), W, \tau) : \mathcal{F}_1}\textit{(Generalised random process)}$$

$$\frac{\mathcal{T}_a = (T_a, \Omega_a, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega_b, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a \times \mathcal{T}_b := (T_a \times T_b, \Omega_a \times \Omega_b, \tau_a \times \tau_b) : \mathcal{F}_1}\textit{(Product)} \qquad \frac{\mathcal{T}_a = (T_a, \Omega, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a + \mathcal{T}_b := (T_a + T_b, \Omega, \tau_a + \tau_b) : \mathcal{F}_1}\textit{(Coproduct)}$$

$$\frac{\mathcal{T}_a = (T_a, \Omega, \tau_a) : \mathcal{F}_1,\ \mathcal{T}_b = (T_b, \Omega, \tau_b) : \mathcal{F}_1}{\mathcal{T}_a \circ \mathcal{T}_b := (T_a \circ T_b, \Omega, \tau_a \circ T_a \tau_b) : \mathcal{F}_1}\textit{(Composition)} \qquad \frac{\mathcal{T} = (T, \Omega, \tau) : \mathcal{F}_1,\ \tilde{\tau} : T\Omega \to \Omega}{(T, \Omega, \tilde{\tau}) : \mathcal{F}_1}\textit{(Modality replacement)}$$

"Any problem with non-determinism (choice or randomness),
where players optimise expectation,
can be solved using VI"

Leaf structure – Conclusion

- **Represent** model checking problems as ppt.
- **Axiomatise** the conditions enabling VI as "ppt with leaf structure".
- In this nice categorical context, **prove** that VI works.
- **Explain** the "categorical essence" of VI.
- **Instantiate** to a large family of examples.

VI has meaning for problem $\mathcal{T} \iff \mathcal{T}$ admits a "monotone lifting"

Interlude – Categorical Structure

Lemma (Functoriality and Monotonicty of Value)

Let $\mathcal{T} = (T, \Omega, \tau)$ be a ppt with leaf structure $\mathcal{T}_l$.

- $$\begin{aligned} V : \mathbf{Coalg}(T) &\longrightarrow \mathbf{Set}/\Omega \\ g &\longmapsto \sup_{\alpha} g_{\tau}^{\alpha}(\bot) \end{aligned}$$

  $$\begin{array}{ccc} X & \xrightarrow{\varphi} & Y \\ \downarrow{\scriptstyle c} & & \downarrow{\scriptstyle d} \\ TX & \xrightarrow{T\varphi} & TY \end{array} \quad \longmapsto \quad \begin{array}{ccc} X & \xrightarrow{\varphi} & Y \\ & {\scriptstyle V(c)} \searrow \quad \swarrow {\scriptstyle V(d)} & \\ & \Omega & \end{array}$$

  is a well-defined **Set**-functor.
- Moreover, for pointwise orders based on Ω_l, and any $c, d : X \to TX$,

  $$c_{\tau}^{*} \leq d_{\tau}^{*} \implies V(c) \leq V(d).$$

Surrogate models

Shapes, connections and BVI algorithms (1) – Result

Axiomatise the conditions enabling BVI with surrogate models.

Shapes, connections and BVI algorithms (1) – Result

Axiomatise the conditions enabling BVI with surrogate models.

- Two ingredients. Start with **"shape structures"** on ppt, and create **"connections"** $\mathcal{T}_{\downarrow} \triangleleft \mathcal{T}_{\uparrow}$.

Shapes, connections and BVI algorithms (1) – Result

Axiomatise the conditions enabling BVI with surrogate models.

- Two ingredients. Start with **"shape structures"** on ppt, and create **"connections"** $\mathcal{T}_\downarrow \triangleleft \mathcal{T}_\uparrow$.
- Suppose that you want to compute $V_\downarrow(g_\downarrow)$.

  If we have leaf structure + "tight" connection

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\big((g_\downarrow)^*_{\mathcal{T}_\downarrow}\big) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

Shapes, connections and BVI algorithms (1) – Result

Axiomatise the conditions enabling BVI with surrogate models.

- Two ingredients. Start with **"shape structures"** on ppt, and create **"connections"** $\mathcal{T}_{\downarrow} \triangleleft \mathcal{T}_{\uparrow}$.
- Suppose that you want to compute $V_{\downarrow}(g_{\downarrow})$.
  If we have leaf structure + "tight" connection

$$V_{\downarrow}(g_{\downarrow}) := \mathrm{lfp}\big((g_{\downarrow})^{*}_{\mathcal{T}_{\downarrow}}\big) = \sup_{\alpha:\mathbf{Ord}} p_{\alpha} = \inf_{\alpha:\mathbf{Ord}} V_{\uparrow}(g_{\alpha}).$$

- Lower bounds p_{α} provided by VI.
- Upper bounds $V_{\uparrow}(g_{\alpha})$ are solutions of surrogate models.
  The surrogate problem g_{α} is built from the lower bound p_{α}.

Shapes, connections and BVI algorithms (2) – Intuition

Axiomatise the conditions enabling BVI with surrogate models.

- We need to compare the branching structures of processes with different types. (Example: Minimizer restriction in SG, going to SG or MDP)

Shapes, connections and BVI algorithms (2) – Intuition

Axiomatise the conditions enabling BVI with surrogate models.

- We need to compare the branching structures of processes with different types. (Example: Minimizer restriction in SG, going to SG or MDP)
- **First**, compare branching structure on a **single type**. **"Shape structure"** Extension T_s : **Set** $\to$ **Preord** compatible with τ.
  $\tau : T_s\Omega = (T_0\Omega_0, \sqsubseteq) \to (\Omega_0, \leq)$

Shapes, connections and BVI algorithms (2) – Intuition

Axiomatise the conditions enabling BVI with surrogate models.

- We need to compare the branching structures of processes with different types. (Example: Minimizer restriction in SG, going to SG or MDP)
- **First**, compare branching structure on a **single type**. **"Shape structure"** Extension T_s : **Set** $\to$ **Preord** compatible with τ.
  $\tau : T_s\Omega = (T_0\Omega_0, \sqsubseteq) \to (\Omega_0, \leq)$
- **Second**, relate **multiple types** with **connections** $(T_\downarrow, \Omega_s, \tau_\downarrow) \triangleleft (T_\uparrow, \Omega_s, \tau_\uparrow)$.

$$T_{\downarrow,0} \xRightarrow{\alpha_\downarrow} T_{*,0} \xLeftarrow{\alpha_\uparrow} T_{\uparrow,0}.$$

$$\begin{array}{ccccc} T_\downarrow\Omega & \xrightarrow{\alpha_{\downarrow,\Omega_0}} & T_*\Omega & \xleftarrow{\alpha_{\uparrow,\Omega_0}} & T_\uparrow\Omega \\ & \leq & \downarrow \tau_* & \leq & \\ \tau_\downarrow \searrow & & \Omega & & \swarrow \tau_\uparrow \end{array}$$

Shapes, connections and BVI algorithms (3) – Comment

Theorem (Double approximation)

Let $\mathcal{T}_\downarrow$, $\mathcal{T}_\uparrow$ be ppt with both shape and leaf structure such that $\mathcal{T}_\downarrow \lhd \mathcal{T}_\uparrow$. Moreover, suppose that $\mathcal{T}_\uparrow$ is a ***tight overapproximation*** *of $\mathcal{T}_\downarrow$.*

For any $g_\downarrow : X \to T_\downarrow X$, $V_\downarrow(g_\downarrow) : \Omega^X$ can be computed in the following way. For each ordinal α, compute

- $p_\alpha = (g_\downarrow)^\alpha_{\tau_\downarrow}(\bot) : \Omega^X$,
- $g_\alpha : X \to T_\uparrow X$ *such that* $g_\downarrow \lhd g_\alpha$ *and* $(g_\downarrow)^*_{\tau_\downarrow}(p_\alpha) = (g_\alpha)^*_{\tau_\uparrow}(p_\alpha)$,
- $V_\uparrow(g_\alpha)$.

Then,

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\big((g_\downarrow)^*_{\tau_\downarrow}\big) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

Shapes, connections and BVI algorithms (3) – Comment

Theorem (Double approximation)

*Let $\mathcal{T}_\downarrow$, $\mathcal{T}_\uparrow$ be ppt with both shape and leaf structure such that $\mathcal{T}_\downarrow \triangleleft \mathcal{T}_\uparrow$. Moreover, suppose that $\mathcal{T}_\uparrow$ is a **tight overapproximation** of $\mathcal{T}_\downarrow$.*

For any $g_\downarrow : X \to T_\downarrow X$, $V_\downarrow(g_\downarrow) : \Omega^X$ can be computed in the following way. For each ordinal α, compute

- $p_\alpha = (g_\downarrow)^\alpha_{\mathcal{T}_\downarrow}(\bot) : \Omega^X$,
- $g_\alpha : X \to T_\uparrow X$ *such that* $g_\downarrow \triangleleft g_\alpha$ *and* $(g_\downarrow)^*_{\mathcal{T}_\downarrow}(p_\alpha) = (g_\alpha)^*_{\mathcal{T}_\uparrow}(p_\alpha)$,
- $V_\uparrow(g_\alpha)$.

Then,

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\big((g_\downarrow)^*_{\mathcal{T}_\downarrow}\big) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

Shapes, connections and BVI algorithms (3) – Comment

Theorem (Double approximation)

Let $\mathcal{T}_\downarrow$, $\mathcal{T}_\uparrow$ be ppt with both shape and leaf structure such that $\mathcal{T}_\downarrow \lhd \mathcal{T}_\uparrow$.
Moreover, suppose that $\mathcal{T}_\uparrow$ is a ***tight overapproximation*** *of $\mathcal{T}_\downarrow$.*

For any $g_\downarrow : X \to T_\downarrow X$, $V_\downarrow(g_\downarrow) : \Omega^X$ can be computed in the following way.
For each ordinal α, compute

- $p_\alpha = (g_\downarrow)^\alpha_{\mathcal{T}_\downarrow}(\bot) : \Omega^X$,
- $g_\alpha : X \to T_\uparrow X$ *such that* $g_\downarrow \lhd g_\alpha$ *and* $(g_\downarrow)^*_{\mathcal{T}_\downarrow}(p_\alpha) = (g_\alpha)^*_{\mathcal{T}_\uparrow}(p_\alpha)$,
- $V_\uparrow(g_\alpha)$.

Then,

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\left((g_\downarrow)^*_{\mathcal{T}_\downarrow}\right) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

- **Application:** to solve a problem $T_\downarrow$ by BVI, check that is is solvable by VI (leaves), and find a (tightly connected) surrogate problem $T_\uparrow$.

Shapes, connections and BVI algorithms (3) – Comment

Theorem (Double approximation)

Let $\mathcal{T}_\downarrow$, $\mathcal{T}_\uparrow$ be ppt with both shape and leaf structure such that $\mathcal{T}_\downarrow \lhd \mathcal{T}_\uparrow$. Moreover, suppose that $\mathcal{T}_\uparrow$ is a ***tight overapproximation*** *of $\mathcal{T}_\downarrow$.*

For any $g_\downarrow : X \to T_\downarrow X$, $V_\downarrow(g_\downarrow) : \Omega^X$ can be computed in the following way. For each ordinal α, compute

- $p_\alpha = (g_\downarrow)^\alpha_{\mathcal{T}_\downarrow}(\bot) : \Omega^X$,
- $g_\alpha : X \to T_\uparrow X$ *such that* $g_\downarrow \lhd g_\alpha$ *and* $(g_\downarrow)^*_{\mathcal{T}_\downarrow}(p_\alpha) = (g_\alpha)^*_{\mathcal{T}_\uparrow}(p_\alpha)$,
- $V_\uparrow(g_\alpha)$.

Then,

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\big((g_\downarrow)^*_{\mathcal{T}_\downarrow}\big) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

- **Application:** to solve a problem $T_\downarrow$ by BVI, check that is is solvable by VI (leaves), and find a (tightly connected) surrogate problem $T_\uparrow$.
- $V_\uparrow(g_\alpha)$ must be easy to compute.

Shapes, connections and BVI algorithms (3) – Comment

Theorem (Double approximation)

Let $\mathcal{T}_\downarrow$, $\mathcal{T}_\uparrow$ be ppt with both shape and leaf structure such that $\mathcal{T}_\downarrow \lhd \mathcal{T}_\uparrow$. Moreover, suppose that $\mathcal{T}_\uparrow$ is a ***tight overapproximation*** *of $\mathcal{T}_\downarrow$.*

For any $g_\downarrow : X \to T_\downarrow X$, $V_\downarrow(g_\downarrow) : \Omega^X$ can be computed in the following way. For each ordinal α, compute

- $p_\alpha = (g_\downarrow)^\alpha_{\mathcal{T}_\downarrow}(\bot) : \Omega^X$,
- $g_\alpha : X \to T_\uparrow X$ *such that* $g_\downarrow \lhd g_\alpha$ *and* $(g_\downarrow)^*_{\mathcal{T}_\downarrow}(p_\alpha) = (g_\alpha)^*_{\mathcal{T}_\uparrow}(p_\alpha)$,
- $V_\uparrow(g_\alpha)$.

Then,

$$V_\downarrow(g_\downarrow) := \mathrm{lfp}\big((g_\downarrow)^*_{\mathcal{T}_\downarrow}\big) = \sup_{\alpha:\mathbf{Ord}} p_\alpha = \inf_{\alpha:\mathbf{Ord}} V_\uparrow(g_\alpha).$$

- **Application:** to solve a problem $T_\downarrow$ by BVI, check that is is solvable by VI (leaves), and find a (tightly connected) surrogate problem $T_\uparrow$.
- $V_\uparrow(g_\alpha)$ must be easy to compute.
- Shape structures must be carefully constructed to enable connections.

Conclusion

Process	$\leftrightarrow$	Process type T : **Set** $\to$ **Set**
Problem	$\leftrightarrow$	Process predicate type $\mathcal{T}$
Possibility of **VI** for $\mathcal{T}$	$\Leftrightarrow$	$\mathcal{T}$ can be given **leaf structure**
Possibility of **surrogate models** of $\mathcal{T}$	$\Leftarrow$	$\mathcal{T}$ can be given **shape structure**
Possibility of **BVI** for $\mathcal{T}$	$\Leftarrow$	$\mathcal{T}$ can be given **both** and tight **connections** can be found

"Semantics applied to algorithmic concepts"

Future work

Future work

What has been done

- Categorical explanations of
  fixed-point theory and surrogate models for coalgebra
- Thus, recipes for VI and BVI algorithms in model checking

Future work

- Avoid *transfinite* value iteration:

$$(\text{Monotone } g_\tau^*) \to (\text{Scott-continuous } g_\tau^*)$$

- Accomodate more kinds of surrogate models!
  Make the theory more useful with new connection types, such as

$$\mathit{MDP} \to \mathit{WG}$$

  We have some ideas!

Also: new instances, fibrations, alternating fixed points, approximate connections, $\mathbb{N}_\infty$ truth object, ...

Conclusion

Thank you !