Calculational Design of Hyperlogics by Abstract Interpretation

Patrick Cousot & Jeffery Wang

Courant Institute, New York University

Objective

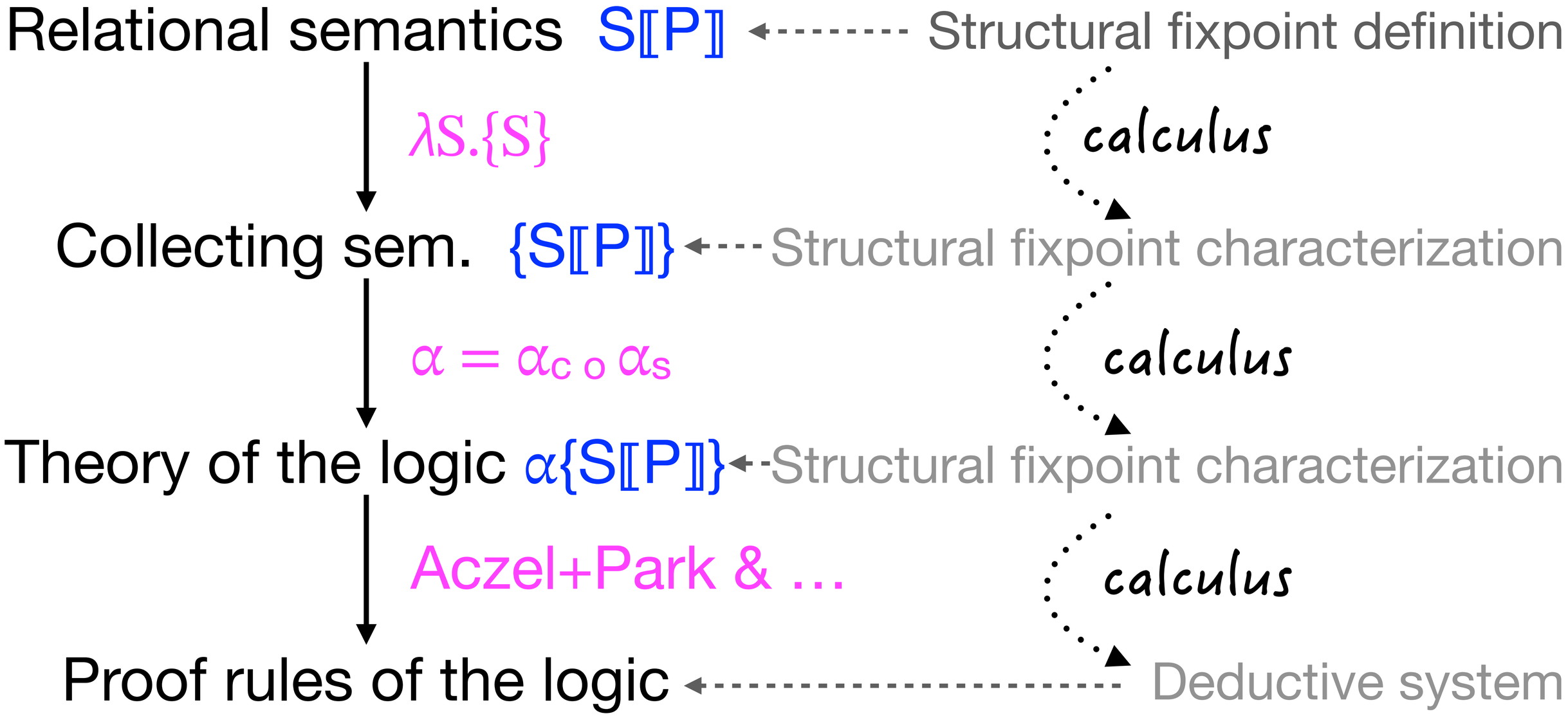
Conceive a method to design program transformational hyperlogics

Transformational logic = Hoare style logics {P} S {Q}

Understanding a program logic

- What is the program semantics? $S[\![P]\!]$
- What is the strongest program semantic property (collecting semantics)? $\{S[\![P]\!]\}$
- What is the strongest program property of interest? $\alpha_s\{S[\![P]\!]\}$
- The properties of interest derive by implication (consequence rule) $\alpha_c \circ \alpha_s\{S[\![P]\!]\}$ (theory of the logic)
- What are the proof rules?

Reminder (POPL 2024)



Methodology

Can we calculate hyperlogics proof systems by structural abstractions of the program semantics?

We will conclude that ``Yes'', but

- For hyperlogics, the strongest program property of interest is the collecting semantics itself $\{S[\![P]\!]\}$
- There is no abstraction α_s *(in general)*
- Any proof of a *general* hyperproperty must characterize the program semantics exactly!
- Unmanageable in practice!
- The only workaround is to consider only *abstract* hyperproperties!

Which semantics?

Which semantics?

- Hoare logic soundness/completeness for invariants is with respect to a relational semantics
- The logic would be essentially the same with execution traces (but for primitives)
- Is there a semantics covering both cases (and even many others)?

Algebraic semantics: a structural fixpoint definition

Algebraic semantics

- Parameterized by an abstract semantic domain providing the model of executions and effect of primitives

$$\mathbb{D}^\sharp_+ \triangleq \langle \mathbb{L}^\sharp_+, \sqsubseteq^\sharp_+, \bot^\sharp_+, \sqcup^\sharp_+, \mathrm{init}^\sharp, \mathrm{assign}^\sharp[\![\mathrm{x}, \mathrm{A}]\!], \mathrm{rassign}^\sharp[\![\mathrm{x}, a, b]\!], \mathrm{test}^\sharp[\![\mathrm{B}]\!], \mathrm{break}^\sharp, \mathrm{skip}^\sharp, \fatsemi^\sharp \rangle$$

$$\mathbb{D}^\sharp_\infty \triangleq \langle \mathbb{L}^\sharp_\infty, \sqsubseteq^\sharp_\infty, \top^\sharp_\infty, \sqcap^\sharp_\infty, \fatsemi^\sharp \rangle$$

Algebraic semantics (cont'd)

- Structural fixpoint definition of the effect of commands
- E.g. assignment

$$[\![\mathtt{x = A}]\!]^\sharp_e \triangleq \mathrm{assign}^\sharp[\![\mathtt{x}, \mathtt{A}]\!]$$
$$[\![\mathtt{x = A}]\!]^\sharp_b \triangleq \bot^\sharp_+$$
$$[\![\mathtt{x = A}]\!]^\sharp_\bot \triangleq \bot^\sharp_\infty$$

- E.g. break

$$[\![\mathtt{break}]\!]^\sharp_e \triangleq \bot^\sharp_+$$
$$[\![\mathtt{break}]\!]^\sharp_b \triangleq \mathrm{break}^\sharp$$
$$[\![\mathtt{break}]\!]^\sharp_\bot \triangleq \bot^\sharp_\infty$$

Algebraic semantics (cont'd)

- E.g. iteration `while (B) S`

$$\overleftarrow{F}^\sharp_e \triangleq \boldsymbol{\lambda} X \in \mathbb{L}^\sharp_+ \bullet \mathsf{init}^\sharp \sqcup^\sharp_+ (\llbracket \mathtt{B;S} \rrbracket^\sharp_e \mathbin{\fatsemi}^\sharp X)$$

$$F^\sharp_\perp \triangleq \boldsymbol{\lambda} X \in \mathbb{L}^\sharp_\infty \bullet \llbracket \mathtt{B;S} \rrbracket^\sharp_e \mathbin{\fatsemi}^\sharp X$$

$$\llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_e \triangleq (\mathsf{lfp}^{\sqsubseteq^\sharp_+} \overleftarrow{F}^\sharp_e) \mathbin{\fatsemi}^\sharp (\llbracket \neg \mathtt{B} \rrbracket^\sharp_e \sqcup^\sharp_e \llbracket \mathtt{B;S} \rrbracket^\sharp_b)$$

$$\llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_b \triangleq \bot^\sharp_+$$

$$\llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_{bi} \triangleq (\mathsf{lfp}^{\sqsubseteq^\sharp_+} \overleftarrow{F}^\sharp_e) \mathbin{\fatsemi}^\sharp \llbracket \mathtt{B;S} \rrbracket^\sharp_\perp$$

$$\llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_{li} \triangleq \mathsf{gfp}^{\sqsubseteq^\sharp_\infty} F^\sharp_\perp$$

$$\llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_\perp \triangleq \llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_{bi} \sqcup^\sharp_\infty \llbracket \mathtt{while\ (B)\ S} \rrbracket^\sharp_{li}$$

Algebraic semantics (cont'd)

- The classic postulated presentation by equational axioms [(*)] can be calculated by
  - structural induction
  - Aczel correspondence between fixpoints and deductive systems (see POPL 2024)

(*) C. A. R. Hoare, Ian J. Hayes, Jifeng He, Carroll Morgan, A. W. Roscoe, Jeff W. Sanders, Ib Holm Sørensen, J. Michael Spivey, and Bernard Sufrin. 1987. Laws of Programming. *Commun. ACM* 30, 8 (1987), 672–686. https://doi.org/10.1145/27651.27653

How to express program properties?

“Programs are predicates” (*)

- We are only interested in properties of programs (not in arbitrary properties)
- A program encodes a program execution property defined by its semantics
- So defining properties as programs, we don’t need a language for programs + another language for predicates!
- Other encodings of properties are mere abstractions.

(*) Eric C. R. Hehner. 1990. A Practical Theory of Programming. *Sci. Comput. Program.* 14, 2-3 (1990), 133–158. https://doi.org/10.1016/0167-6423(90)90018-9

Property transformer

Algebraic property transformer

- Forward property transformer:

$$\mathrm{post}^\sharp \in \mathbb{L}^\sharp \nrightarrow \mathbb{L}^\sharp \nrightarrow \mathbb{L}^\sharp$$

$$\mathrm{post}^\sharp(S)P \quad \triangleq \quad P \mathbin{\fatsemi}^\sharp S$$

A structural fixpoint characterization of the property transformer

A calculus of algebraic execution properties

- Galois connection

$$\forall S \in \mathbb{L} \,.\, \langle \mathbb{L}, \sqsubseteq \rangle \underset{\overrightarrow{\mathrm{post}(S)}}{\overset{\overleftarrow{\widetilde{\mathrm{pre}}(S)}}{\leftrightarrows}} \langle \mathbb{L}, \sqsubseteq \rangle \qquad (\langle \mathbb{L}, \sqsubseteq, \sqcup \rangle \text{ is a poset})$$

- Using the abstraction methodology of POPL 2024, we generalize POPL 2024 to
  - a structural fixpoint algebraic calculus of execution properties
  - (and the lattice of algebraic transformational logics)

Hyperproperties

Algebraic hyperproperties

- $\mathbb{L}$ is the semantic domain (e.g. set of finite and infinite traces, input-output relation)
- $\wp(\mathbb{L})$ is the set of hyperproperties (defined in extension)
- $\subseteq$ is logical implication

Hyperproperty transformer

Algebraic hyperproperty transformer

- Transformer

$$\begin{aligned} \mathrm{Post}^\sharp &\in \mathbb{L}^\sharp \to \wp(\mathbb{L}^\sharp) \nrightarrow \wp(\mathbb{L}^\sharp) \\ \mathrm{Post}^\sharp(S)\mathcal{P} &\triangleq \{\mathrm{post}^\sharp(S)P \mid P \in \mathcal{P}\} \end{aligned}$$

- Galois connection

$$\langle \wp(\mathbb{L}^\sharp), \subseteq \rangle \underset{\mathrm{Post}^\sharp(S)}{\overset{\mathrm{Pre}(S)}{\leftrightarrows}} \langle \wp(\mathbb{L}^\sharp), \subseteq \rangle$$

Structural fixpoint characterization of the hyperproperty transformer

Incomplete structural characterization of $\mathrm{Post}^\sharp(S)$

- Counter-example

$$\mathrm{Post}^\sharp[\![\texttt{if (B) } S_1 \texttt{ else } S_2]\!]^\sharp\mathcal{P}$$

$$= \{\mathrm{post}^\sharp[\![B; S_1]\!]^\sharp P \sqcup^\sharp \mathrm{post}^\sharp[\![\neg B; S_2]\!]^\sharp P \mid P \in \mathcal{P}\}$$

$$\subseteq \{\mathrm{post}^\sharp[\![B; S_1]\!]^\sharp P_1 \sqcup^\sharp \mathrm{post}^\sharp[\![\neg B; S_2]\!]^\sharp P_2 \mid P_1 \in \mathcal{P} \wedge P_2 \in \mathcal{P}\}$$

$$= \{Q_1 \sqcup^\sharp Q_2 \mid Q_1 \in \mathrm{Post}^\sharp[\![B; S_1]\!]^\sharp\mathcal{P} \wedge Q_2 \in \mathrm{Post}^\sharp[\![\neg B; S_2]\!]^\sharp\mathcal{P}\}$$

- This structural collecting semantics (*) is incomplete

(*) Thibault Dardinier and Peter Müller. 2024. Hyper Hoare Logic: (Dis-)Proving Program Hyperproperties. *Proceedings of the ACM on Programming Languages (PACMPL)* 8, Issue PLDI, Article No.: 207 (June 2024), 1485–1509. https://doi.org/10.1145/3656437

Complete structural characterization of $\mathrm{Post}^\sharp(S)$

$$\{\mathrm{post}^\sharp(S)P\} \quad = \quad \mathrm{Post}^\sharp(S)\{P\}$$

- Example:

$$\mathrm{Post}^\sharp[\![\texttt{if (B) } \mathtt{S}_1 \texttt{ else } \mathtt{S}_2]\!]^\sharp\mathcal{P}$$

$$= \{\mathrm{post}^\sharp[\![\mathtt{B};\mathtt{S}_1]\!]^\sharp P \sqcup^\sharp \mathrm{post}^\sharp[\![\neg\mathtt{B};\mathtt{S}_2]\!]^\sharp P \mid P \in \mathcal{P}\}$$

$$= \{Q_1 \sqcup^\sharp Q_2 \mid Q_1 \in \{\mathrm{post}^\sharp[\![\mathtt{B};\mathtt{S}_1]\!]^\sharp P\} \wedge Q_2 \in \{\mathrm{post}^\sharp[\![\neg\mathtt{B};\mathtt{S}_2]\!]^\sharp P\} \wedge P \in \mathcal{P}\}$$

$$= \{Q_1 \sqcup^\sharp Q_2 \mid Q_1 \in \mathrm{Post}^\sharp[\![\mathtt{B};\mathtt{S}_1]\!]^\sharp\{P\} \wedge Q_2 \in \mathrm{Post}^\sharp[\![\neg\mathtt{B};\mathtt{S}_2]\!]^\sharp\{P\} \wedge P \in \mathcal{P}\}$$

- We get a complete elementwise characterization of Post#(S)

Calculational design of the algebraic hyperlogic rules

Upper and lower algebraic hyperlogics

- Definition

$$\overline{\{\!|\,\mathcal{P}\,|\!\}}\ \mathsf{s}\ \overline{\{\!|\,\mathcal{Q}\,|\!\}} \quad = \quad \mathrm{Post}^\sharp[\![\mathsf{s}]\!]^\sharp\mathcal{P} \subseteq \mathcal{Q}$$

$$\underline{\{\!|\,\mathcal{P}\,|\!\}}\ \mathsf{s}\ \underline{\{\!|\,\mathcal{Q}\,|\!\}} \quad = \quad \mathcal{Q} \subseteq \mathrm{Post}^\sharp[\![\mathsf{s}]\!]^\sharp\mathcal{P}$$

- The proof system is derived by calculational design (as in POPL 2024)

Upper algebraic hyperlogic for iteration

$$\frac{\left(P_e = \mathrm{lfp}^{\sqsubseteq^\sharp_+} \vec{F}^\sharp_{pe}(P') \wedge \overline{\{\!|}\,\{P_e\}\,\overline{|\!\}}\,\neg \mathsf{B}\,\overline{\{\!|}\,\{Q_e\}\,\overline{|\!\}} \wedge \overline{\{\!|}\,\{P_e\}\,\overline{|\!\}}\,\mathsf{B};\mathsf{S}\,\overline{\{\!|}\,\{Q_b\}\,\overline{|\!\}} \wedge \overline{\{\!|}\,\{P_e\}\,\overline{|\!\}}\,\mathsf{B};\mathsf{S}\,\overline{\{\!|}\,\{Q_{\bot\ell}\}\,\overline{|\!\}} \wedge Q_{\bot b} = \mathrm{gfp}^{\sqsubseteq^\sharp_\infty} F^\sharp_{p\bot} \wedge P' \in \mathcal{P}\right) \Rightarrow \left(\langle e : Q_e \sqcup^\sharp_e Q_b,\ \bot : Q_{\bot\ell} \sqcup^\sharp_\infty Q_{\bot b},\ br : P_{br}\rangle \in \mathcal{Q}\right)}{\overline{\{\!|}\,\mathcal{I}\,\overline{|\!\}}\,\texttt{while (B) S}\,\overline{\{\!|}\,\mathcal{Q}\,\overline{|\!\}}}$$

- Requires an *EXACT* characterization of the program semantics
- *Unmanageable* in practice

Abstractions

Abstractions

- Since proofs of general hyperproperties are unmanageable, we consider abstractions of
  - the algebraic semantics
  - program properties
  - program hyperproperties
  - program logics

Algebraic semantics abstraction

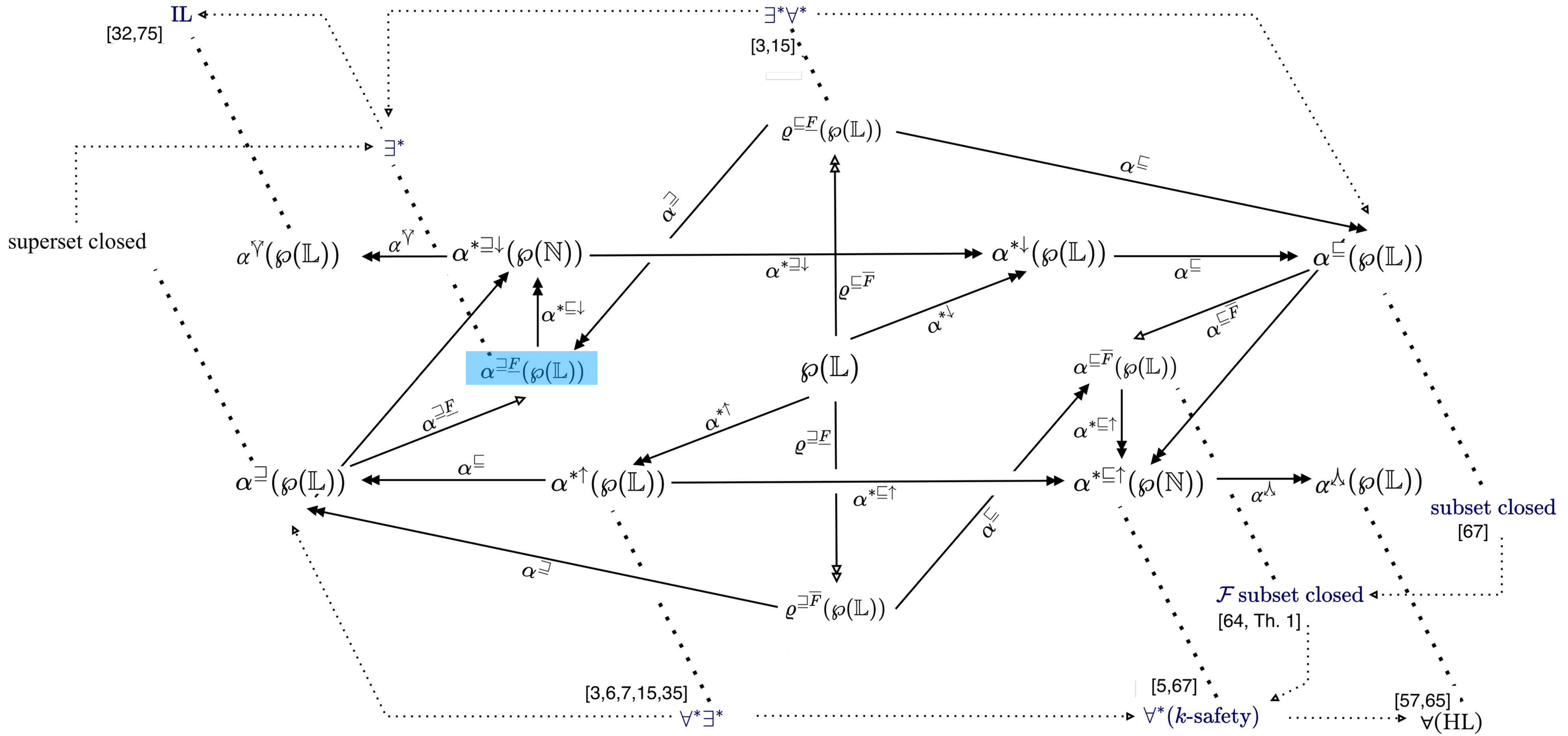
- An abstraction of the algebraic semantics is another instance of the algebraic semantics
  - e.g. trace semantics → relational semantics
- This extends to logics and hyperlogics
- But still proofs require exact characterizations of the (abstract) semantics

Hyperproperty abstraction

Hyperproperty abstraction

- A dozen abstractions are considered in the paper
- This leads to a lattice of hyperlogics

Hierarchy of hyperlogics



Chain limit order ideal abstraction

Chain limit order ideal abstraction (cont'd)

- The chain limit order ideal abstraction of algebraic hyperproperties is an algebraic generalization of the abstraction to ∀*∃* hyperproperties

- ∀*∃* hyperproperties (for traces in Π) $\mathcal{AEH} \triangleq$

$$\{\{P \in \wp(\Pi) \mid \forall \pi_1 \in P \,.\, \exists \pi_2 \in P \,.\, \langle \pi_1, \pi_2 \rangle \in A\} \mid A \in \wp(\Pi \times \Pi)\}$$

Chain limit order ideal abstraction

$$\alpha^{\uparrow}(\mathcal{P}) \triangleq \left\{ \bigsqcup_{i \in \mathbb{N}} P_i \mid \langle P_i,\ i \in \mathbb{N} \rangle \in \mathcal{P} \text{ is an increasing chain with existing lub} \right\}$$

$$\alpha^{\sqsubseteq}(\mathcal{P}) \triangleq \left\{ P' \in \mathbb{L} \mid \exists P \in \mathcal{P} \,.\, P' \sqsubseteq P \right\}$$

$$\alpha^{\sqsubseteq\uparrow} \triangleq \alpha^{\sqsubseteq} \circ \alpha^{\uparrow}$$ (extensive, increasing, not idempotent)

$$\overset{*}{\alpha}^{\sqsubseteq\uparrow}(\mathcal{P}) \triangleq \mathrm{lfp}^{\subseteq}\, \boldsymbol{\lambda} X \bullet \mathcal{P} \cup \alpha^{\sqsubseteq\uparrow}(X)$$ (upper closure operator hence G.C.)

- in particular for traces:

$$\mathcal{AEH} \subseteq \overset{*}{\alpha}^{\uparrow}(\wp(\wp(\Pi)))$$

Conclusion

Conclusion

- We have introduced a new algebraic semantics (instantiable to any classic semantics)
- We have considered programs (i.e. their semantics) as properties
- We have designed by calculus a general algebraic logic (sound & complete and generalizing POPL 2024)
- We have designed by calculus a general algebraic hyperlogic (sound & complete but unmanageable in practice)
- All this for terminating and nonterminating executions

Conclusion (cont'd)

- We have considered abstractions of algebraic hyperproperties :
  - less expressive than general hyperproperties
  - but with sound and complete hyperlogics using only approximations of the program semantics
- This was illustrated by an algebraic generalization of ∀*∃* hyperproperties

More in the paper

- Various instanciations of the algebraic semantics
- Abstractions of the algebraic semantics leading to complete hyperlogics
- A dozen of other abstractions of hyperproperties
- Including algebraic generalizations of ∃*∀* as well as ∀*∀* hyperproperties
- Correction of errors and generalizations of results in the literature
- etc

Conclusion of the conclusion

A transformational hyperlogic
is
an abstract interpretation
of
an hypertransformer
of
an instantiation
of
an algebraic semantics.

(Conclusion of the conclusion)$^{-1}$

A (hyper)logic is

another (complicated) way

of defining

an abstract interpretation

of

an instantiation

of

an algebraic semantics.

The End, Thank You

- Online full version of the clickable paper + appendix:
  - auxiliary material of the ACM digital library
  - my web page (https://cs.nyu.edu/~pcousot/) + slides
  - arXiv https://arxiv.org/abs/2411.11113
  - Zenodo https://zenodo.org/records/14173478