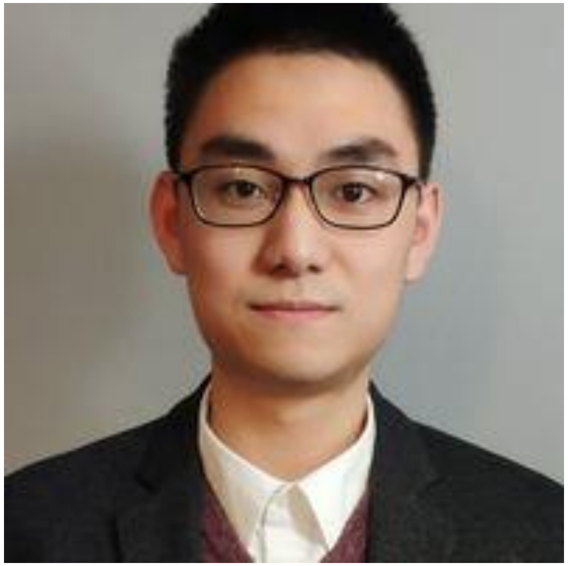
Matching μ-Logic



Xiaohong Chen

Grigore Rosu

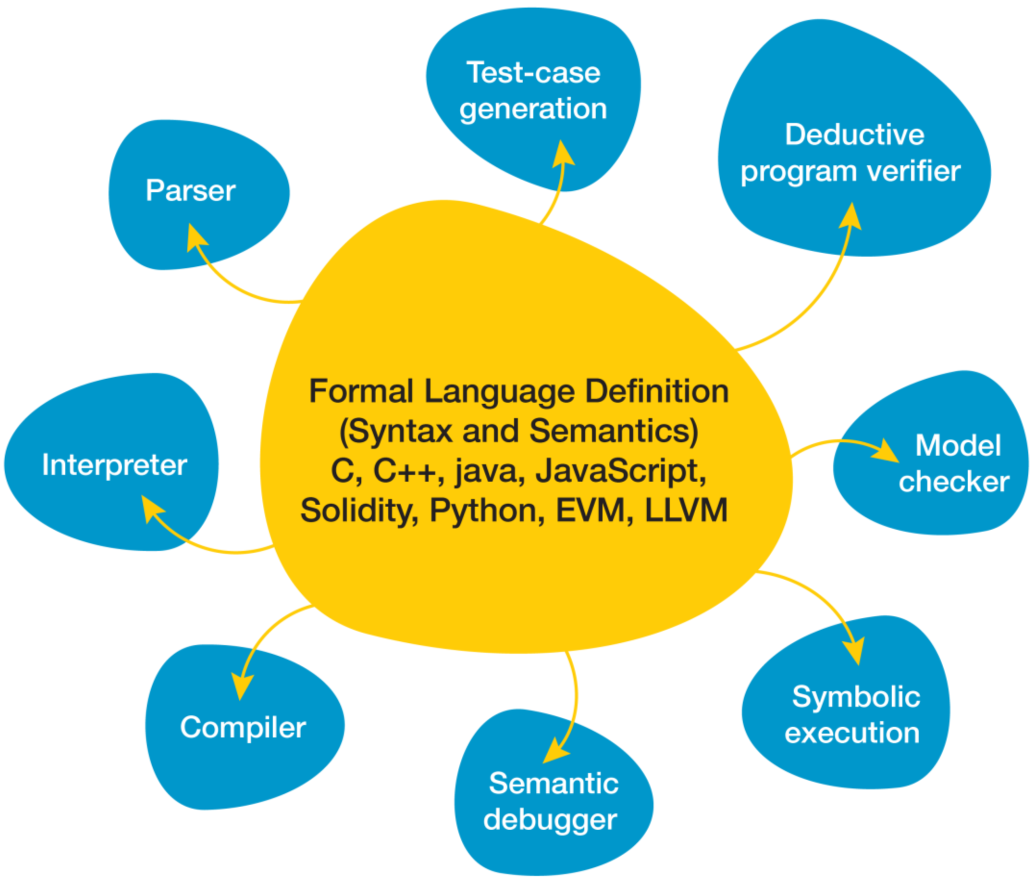
University of Illinois at Urbana-Champaign

Runtime Verification, Inc.

LICS – 2019/06/27

An Ideal Language Framework Vision

We pursue it with the K framework [www.kframework.org]



K scales.

JavaScript ES5: by Park etal [PLDI'15]
- Passes existing conformance test suite (2872 programs)
- Found (confirmed) bugs in Chrome, IE, Firefox, Safari

Java 1.4: by Bogdanas etal [POPL'15]

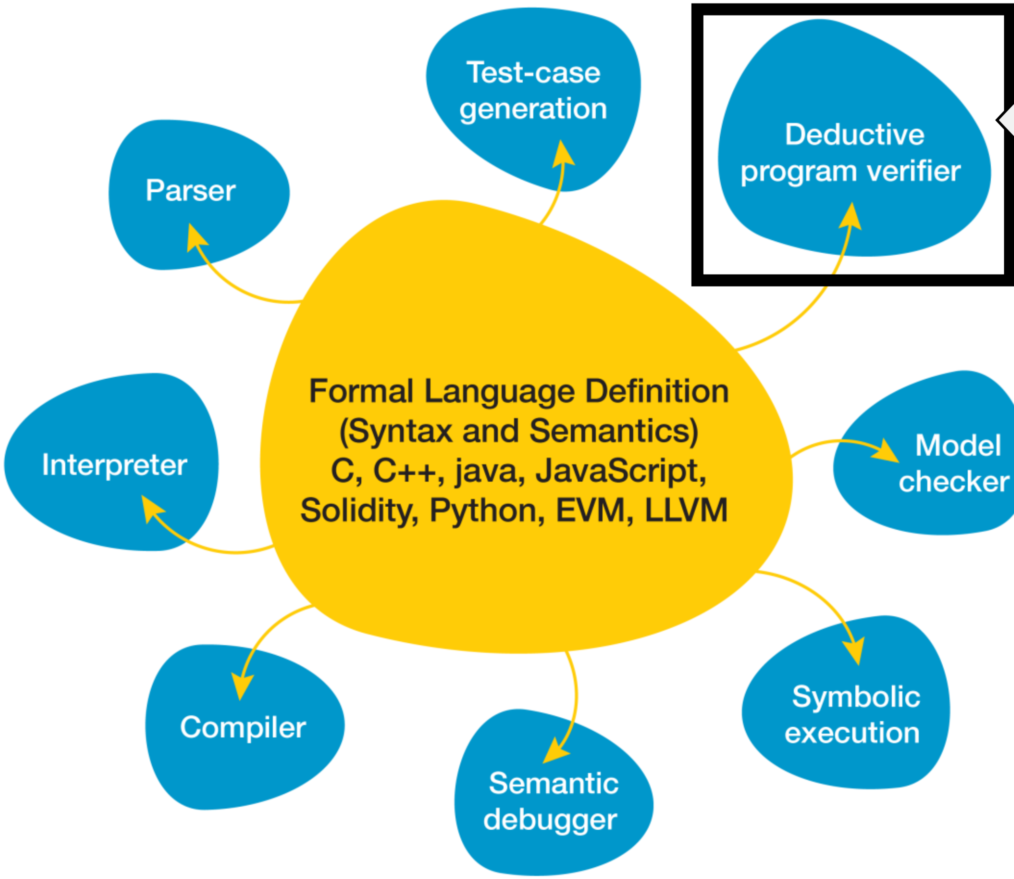
x86: by Dasgupta etal [PLDI'19]

C11: Ellison etal [POPL'12, PLDI'15]
- 192 different types of undefined behavior
- 10,000+ program tests (gcc torture tests, obfuscated C, …)
- Commercialized by startup (Runtime Verification, Inc.)

EVM [CSF'18], Solidity, IELE, Plutus, Vyper, …

An Ideal Language Framework Vision

We pursue it with the K framework [www.kframework.org]



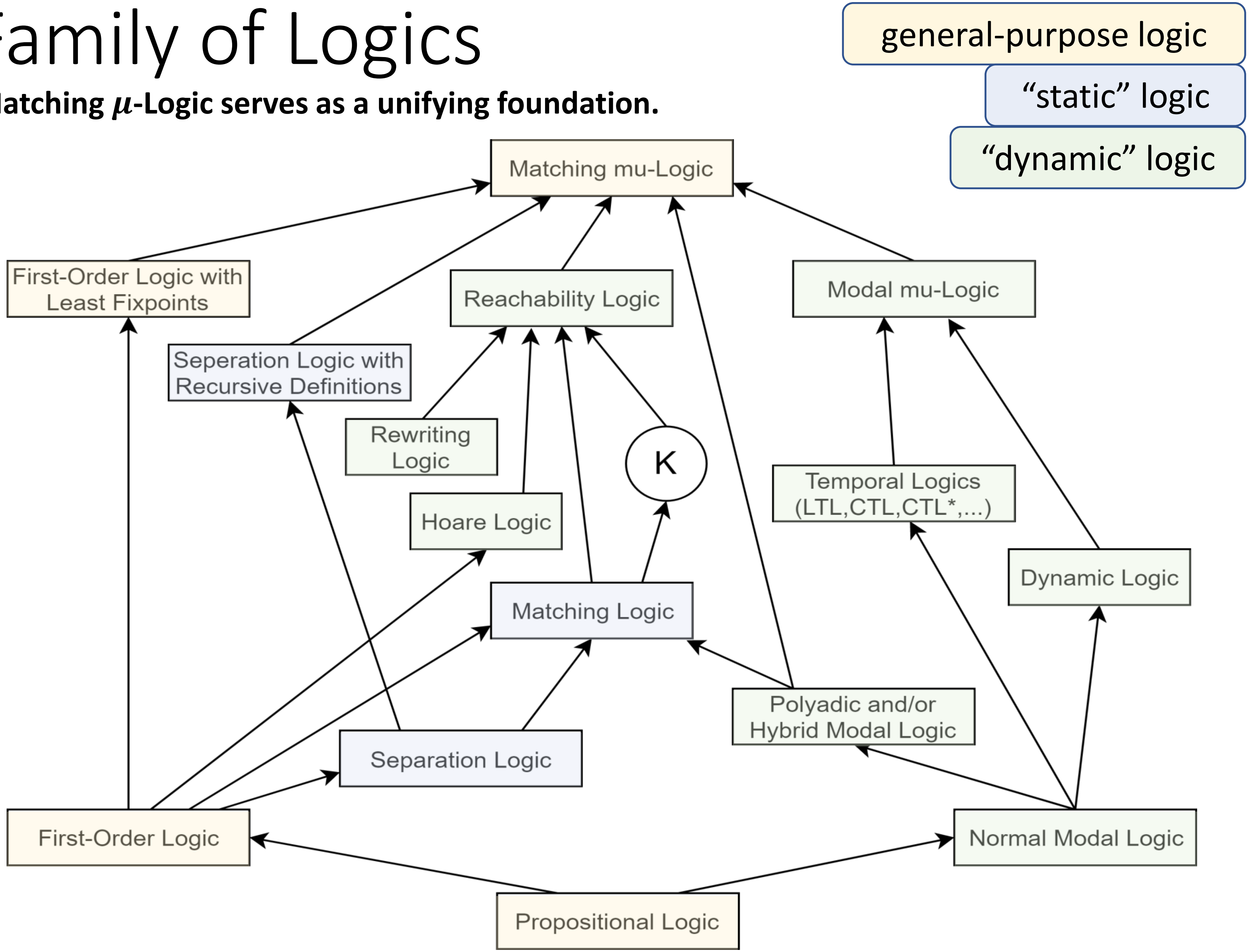
Currently, K uses Matching Logic to specify static structures of programs and Reachability Logic to specify dynamic (reachability) properties.

Can we unify Matching Logic and Reachability Logic so that K has a uniform foundation?

Can we have a more powerful logic that goes beyond Reachability Logic (eg, liveness properties as in LTL/CTL)?

Family of Logics

Matching μ-Logic serves as a unifying foundation.



Talk Overview

- Background:
  - Towards an Ideal Language Framework
  - The need for a uniform and more powerful logic
- Matching μ-Logic:
  - Syntax, Semantics, Proof System
- Applications ("*static*" and "*dynamic*")
  - Reasoning about Constructors and Term Algebras;
  - Reasoning about Transition Systems;
    - Subsuming Modal Logic variants
    - Subsuming Reachability Logic
- Conclusion
  - **Matching μ-Logic serves as a unifying foundation**

Matching μ-Logic Has Simple Syntax (7 syntactic constructs)

Patterns (of each sort *s*)

Element variables

Set variables

Structures

$$\begin{aligned}
\varphi_s ::= {} & x : s \in \mathrm{EVar}_s \mid X : s \in \mathrm{SVar}_s \\
\mid {} & \sigma(\varphi_{s_1}, \ldots, \varphi_{s_n}) \quad \text{with } \sigma \in \Sigma_{s_1 \ldots s_n, s} \\
\mid {} & \varphi_s \wedge \varphi_s \mid \neg \varphi_s \\
\mid {} & \exists z : s' \,.\, \varphi_s \mid \mu Z : s \,.\, \varphi_s \quad \text{if } \varphi_s \text{ is positive in } Z : s
\end{aligned}$$

Logic Constraints

FOL binders, abstraction

(Least) fixpoint binders

Matching μ-Logic Semantics: Pattern Matching

$$\varphi_s ::= x : s \in \mathrm{EVar}_s \mid X : s \in \mathrm{SVar}_s \mid \sigma(\varphi_{s_1}, \ldots, \varphi_{s_n}) \mid \varphi_s \wedge \varphi_s \mid \neg\varphi_s \mid \exists z : s' \,.\, \varphi_s \mid \mu Z : s \,.\, \varphi_s$$

A Model

Sorted Carrier Sets

Symbol Interpretations

$$M = (\{M_s\}_{s \in \mathrm{Sorts}}, \{\sigma_M\}_{\sigma \in \mathrm{Symbols}})$$

$$\sigma_M : M_{s_1} \times \cdots \times M_{s_n} \to \mathcal{P}(M_s)$$

Pattern = The set of elements that ***match*** it.

$\wedge$ as intersection, $\neg$ as complement, $\exists$ as union over all x,
μ as the least fixpoint (examples given later)

Derived Constructs

$$\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$$

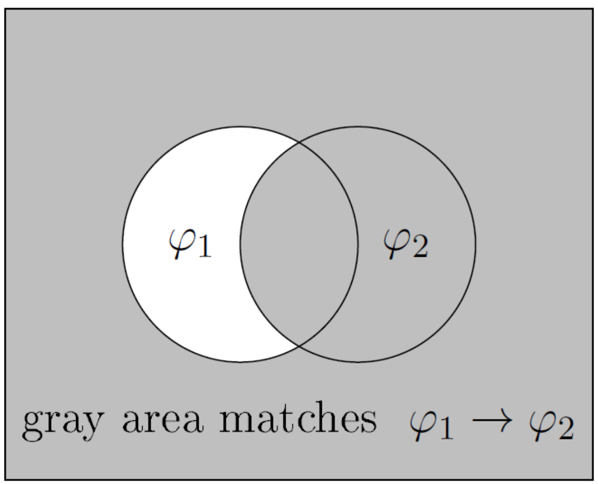
$$\varphi_1 \rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$$

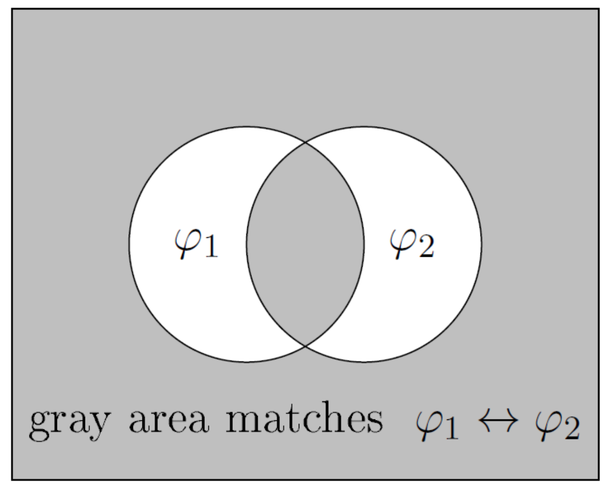
$$\varphi_1 \leftrightarrow \varphi_2 \equiv (\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)$$

$$\forall x{:}s.\varphi \equiv \neg\exists x{:}s.\neg\varphi$$

$$\top_s \equiv \exists x{:}s.x{:}s$$

$$\bot_s \equiv \neg\top_s$$





$$\nu X{:}s.\varphi_s \equiv \neg\mu X{:}s.\neg\varphi_s[\neg X{:}s/X{:}s]$$

Recovering Known Notions, Axiomatically

Definedness

$\lceil \_ \rceil_s^{s'} \in \Sigma_{s,s'}$

Definedness symbol

$\lceil x{:}s \rceil_s^{s'}$

Definedness pattern/axiom

$\lceil \varphi \rceil_s^{s'}$

Either the empty set (when φ empty) or otherwise the total set (when φ non-empty)

Totality

$\lfloor \varphi \rfloor_s^{s'} \equiv \neg \lceil \neg \varphi \rceil_s^{s'}$

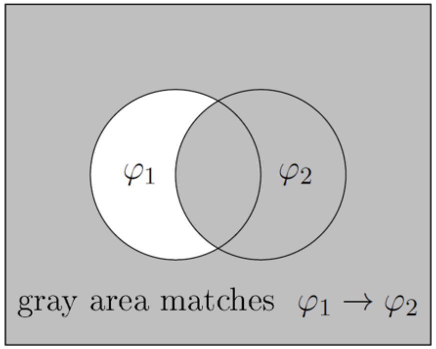
Sort subscripts and superscripts can be inferred from the context, so we do not write them

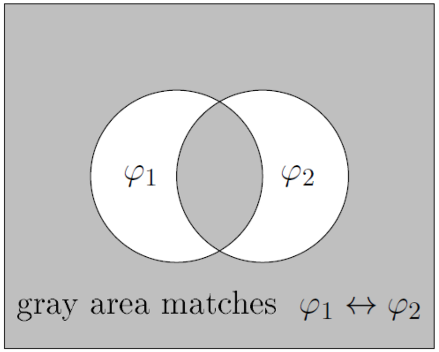
Equality, Membership, Inclusion

$$x \in_s^{s'} \varphi \equiv \lceil x \wedge \varphi \rceil_s^{s'}$$

$$\varphi_1 \subseteq_s^{s'} \varphi_2 \equiv \lfloor \varphi_1 \rightarrow \varphi_2 \rfloor_s^{s'}$$

$$\varphi_1 =_s^{s'} \varphi_2 \equiv \lfloor \varphi_1 \leftrightarrow \varphi_2 \rfloor_s^{s'}$$





Recovering Known Notions, Axiomatically

Functions (recovering "operation" symbols)

$$\sigma \in \Sigma_{s_1 \ldots s_n, s}$$

$$\exists y \, . \, \sigma(x_1, \ldots, x_n) = y$$

We write it using the functional notation:

$$\sigma : s_1 \times \cdots \times s_n \to s$$

Similarly we can define partial functions, total relations, etc. Algebraic specification and FOL subsumed notationally. For example:

$$0 : \to Nat$$

$$succ : Nat \to Nat$$

$$plus : Nat \times Nat \to Nat$$

$$plus(0, y) = y$$

$$plus(succ(x), y) = succ(plus(x, y))$$

Examples of Patterns

- $x : s$ or simply x matched by singletons (of sort s);
- $X : s$ or simply X matched by any sets (of sort s);
- $succ(x)$ matched by the successor of x;
- $0 \vee \exists x. succ(x)$ matched by zero or successors;
- $\mu N. 0 \vee succ(N)$ matched by all natural numbers;
  - The <u>smallest solution</u> of the equation $N = 0 \vee succ(N)$
  - $N = \{0, succ(0), succ\big(succ(0)\big), \ldots\}$

Term Algebras in Matching μ-Logic

Consider the *term algebra* generated by a set of *constructors*

$$C = \{c_i \in \Sigma_{\underbrace{Term \ldots Term}_{arity(c_i) \text{ times}}, Term}\}$$

For example,
Natural numbers = Term algebra freely generated by $\{0, succ\}$.
Lists (inductive data structure) = Term algebra freely generate by $\{nil, cons\}$.

The set of all terms is *precisely captured* by the following pattern/axiom:

$$(\textsc{Inductive Domain}) \quad \mu D. \bigvee_{c \in C} c(D, \ldots, D)$$

For example,
The set of all natural numbers $\mu N. 0 \vee succ(N)$
The set of all lists $\mu L. nil \vee cons(\top_N, L)$.

Separation logic = Matching logic [Map]

- Consider map model, with some useful axioms

$$
\begin{aligned}
&\_ \mapsto \_ : Nat \times Nat \rightharpoonup Map && emp * H = H \\
&emp : \to Map && H_1 * H_2 = H_2 * H_1 \\
&\_ * \_ : Map \times Map \rightharpoonup Map && (H_1 * H_2) * H_3 = H_1 * (H_2 * H_3) \\
&0 \mapsto a = \bot && x \mapsto a * x \mapsto b = \bot
\end{aligned}
$$

$$\_ \mapsto [\_] : Nat \times Seq \to Map \qquad x \mapsto [\epsilon] = emp \qquad x \mapsto [a, S] = x \mapsto a * (x+1) \mapsto [S]$$

- Then we can define map patterns "a la SL"

$$list \in \Sigma_{Nat \times Seq, Map}$$

$$
\begin{aligned}
list(0, \epsilon) &= emp \\
list(x, n \cdot S) &= \exists z \,.\, x \mapsto [n, z] * list(z, S)
\end{aligned}
$$

Separation Logic Derivations Using the Generic Matching Logic Proof System (to be shown next)

- Sample derivation for the "separation logic" theory:

$$
\begin{array}{lcll}
1 \mapsto 5 * 2 \mapsto 0 * 7 \mapsto 9 * 8 \mapsto 1 & = & 1 \mapsto [5,0] * 7 \mapsto [9,1] & = \\
1 \mapsto [5,0] * \mathit{list}(0,\epsilon) * 7 \mapsto [9,1] & \to & (\exists z \,.\, 1 \mapsto [5,z] * \mathit{list}(z,\epsilon)) * 7 \mapsto [9,1] & = \\
\mathit{list}(1, 5 \cdot \epsilon) * 7 \mapsto [9,1] & = & \mathit{list}(1,5) * 7 \mapsto [9,1] & \to \\
\exists z \,.\, 7 \mapsto [9,z] \wedge \mathit{list}(z,5) & = & \mathit{list}(7, 9 \cdot 5) &
\end{array}
$$

- Local reasoning globalized ("structural framing" for free!)
  - Above derivation can be lifted to whole configuration

$$
\forall c : \mathit{Cfg}. \forall h : \mathit{Map} \,.\, (\langle\langle 1 \mapsto 5 * 2 \mapsto 0 * 7 \mapsto 9 * 8 \mapsto 1 * h \rangle_{\mathsf{heap}}\ c \rangle_{\mathsf{cfg}}
$$

$$
\to\ \langle\langle \mathit{list}(7, 9 \cdot 5) * h \rangle_{\mathsf{heap}}\ c \rangle_{\mathsf{cfg}})
$$

Matching μ-Logic Has Simple Proof System (FOL plus 9 additional proof rules)

(PROPOSITIONAL TAUTOLOGY)	φ if φ is a propositional tautology over patterns
(MODUS PONENS)	$\dfrac{\varphi_1 \quad \varphi_1 \to \varphi_2}{\varphi_2}$
($\exists$-QUANTIFIER)	$\varphi[y/x] \to \exists x.\varphi$
($\exists$-GENERALIZATION)	$\dfrac{\varphi_1 \to \varphi_2}{(\exists x.\varphi_1) \to \varphi_2}$ if $x \notin FV(\varphi_2)$
(PROPAGATION$_\bot$)	$C_\sigma[\bot] \to \bot$
(PROPAGATION$_\vee$)	$C_\sigma[\varphi_1 \vee \varphi_2] \to C_\sigma[\varphi_1] \vee C_\sigma[\varphi_2]$
(PROPAGATION$_\exists$)	$C_\sigma[\exists x.\varphi] \to \exists x.C_\sigma[\varphi]$ if $x \notin FV(C_\sigma[\exists x.\varphi])$
(FRAMING)	$\dfrac{\varphi_1 \to \varphi_2}{C_\sigma[\varphi_1] \to C_\sigma[\varphi_2]}$
(EXISTENCE)	$\exists x.\, x$
(SINGLETON VARIABLE)	$\neg(C_1[x \wedge \varphi] \wedge C_2[x \wedge \neg\varphi])$ where C_1 and C_2 are nested symbol contexts.
(SET VARIABLE SUBSTITUTION)	$\dfrac{\varphi}{\varphi[\psi/X]}$
(PRE-FIXPOINT)	$\varphi[\mu X.\,\varphi/X] \to \mu X.\varphi$
(KNASTER-TARSKI)	$\dfrac{\varphi[\psi/X] \to \psi}{\mu X.\,\varphi \to \psi}$

Standard FOL reasoning

$C_\sigma[\blacksquare] \equiv \sigma(\dots, \blacksquare, \dots)$

Frame reasoning about symbols

Technical rules (completeness)

Standard fixpoint reasoning

Matching μ-Logic Reasoning is Powerful

FOL formula $\varphi(x)$

Peano Induction Rule:

$$\boxed{\varphi(0) \wedge \forall y\,.\,(\varphi(y) \to \varphi(succ(y))) \to \forall x\,.\,\varphi(x)}$$

… is a theorem in Matching μ-Logic:

Let $\boxed{\Phi \equiv \exists z.\left(z \wedge \varphi(z)\right)}$.

matched by all numbers where φ holds, i.e., φ(z) iff z∈Φ

$\forall x.\,\varphi(x)$

$\Longleftarrow \forall x.\, x \in \Phi,$

(Inductive Domain)

$\Longleftarrow \boxed{\left(\mu N.\, 0 \vee succ(N)\right)} \to \Phi$, by

(Knaster-Tarski)

$$\frac{\varphi[\psi/X] \to \psi}{\mu X\,.\,\varphi \to \psi}$$

$\Longleftarrow 0 \vee succ(\Phi) \to \Phi$

$\Longleftarrow$ *both* $0 \to \Phi$, i.e., $\varphi(0)$ holds,

and $succ(\Phi) \to \Phi$, i.e., $\varphi(y) \to \varphi(succ(y))$

Outline

We've seen that matching μ-Logic supports

✓Heap patterns $list(x)$

✓Term algebras; inductive structures and reasoning

✓Standard FOL reasoning and fixpoint reasoning

✓Frame reasoning for free

✓Peano induction as an instance

We'll see next…

- Transition systems
- Modal mu-logic, LTL, CTL as instances
- Reachability logic as instances

Transition Systems

$\mathbb{S} = (S, R)$ with a *binary transition relation* $R \subseteq S \times S$

In matching μ-Logic, define:

- A sort $State$ of states with carrier set $M_{State} = S$;
- A symbol $\bullet \in \Sigma_{State,State}$, called *one-path next,* with interpretation

$$\bullet_{\mathbb{S}} : S \to \mathcal{P}(S) \quad \text{with } \bullet_{\mathbb{S}}(t) = \{s \in S \mid s\, R\, t\}$$

$$\begin{array}{lllllllr} \cdots & s & \xrightarrow{R} & s' & \xrightarrow{R} & s'' & \cdots & \text{// states} \\ & \bullet\bullet\varphi & & \bullet\varphi & & \varphi & & \text{// patterns} \end{array}$$

Transition Systems
= Matching μ-Logic with one sort and one unary symbol.

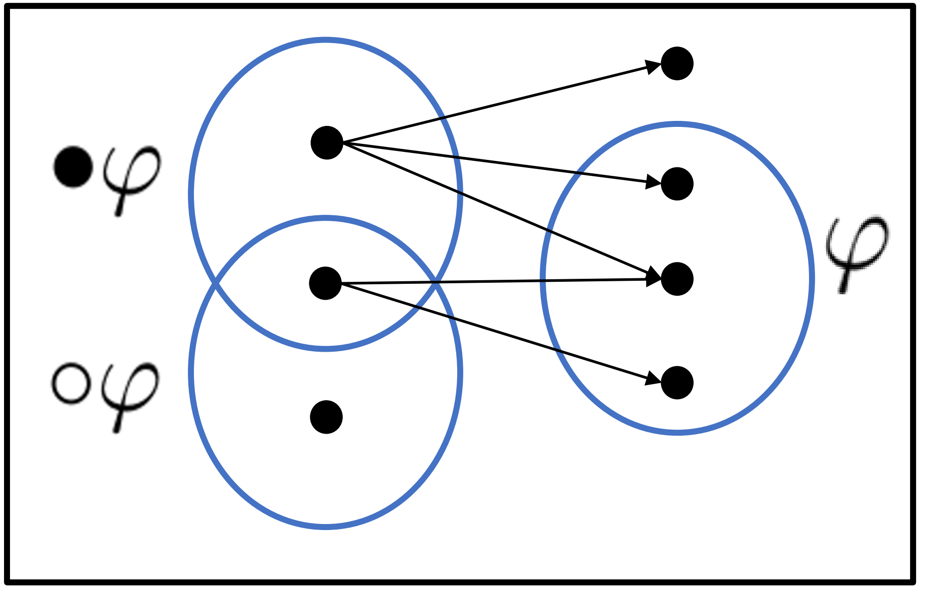
Useful Sugar about Transition Systems

"one-path next" $\bullet\varphi$

"all-path next" $\circ\varphi \equiv \neg\bullet\neg\varphi$

"exists a next state in φ"

"all next states in φ"



Standard definitions …

"eventually" $\Diamond\varphi \equiv \mu X.\, \varphi \vee \bullet X$

"always" $\Box\varphi \equiv \nu X.\, \varphi \wedge \circ X$

"(strong) until" $\varphi_1 \; U \; \varphi_2 \equiv \mu X.\, \varphi_2 \vee (\varphi_1 \wedge \bullet X)$

"well-founded" $\mathsf{WF} \equiv \mu X.\circ X$ // no infinite paths

Modal μ-Logic, LTL, CTL, …
= Matching μ-Logic theories

$$(\textsc{Inf}) \quad \bullet\top$$

All states are some states' predecessors:
no stopped states

$$(\textsc{Fin}) \quad \mathsf{WF} \equiv \mu X \,.\, \circ X$$

All states are well-founded

$$(\textsc{Lin}) \quad \bullet X \to \circ X$$

If s can go to t then s can ***only*** go to t:
linear future (no branching)

Target logic	**Assumption on traces**	**Axioms required**
Modal μ-Logic	Any traces	No axioms
Infinite-trace LTL	Infinite and linear traces	(INF)+(LIN)
Finite-trace LTL	Finite and linear traces	(FIN)+(LIN)
CTL	Infinite traces	(INF)

Reachability Logic: Semantics of K

[LICS'13, RTA'14, RTA'15,OOPLSA'16]

A *reachability rule*:

$$\varphi \Rightarrow \varphi'$$ with φ and φ' configuration patterns

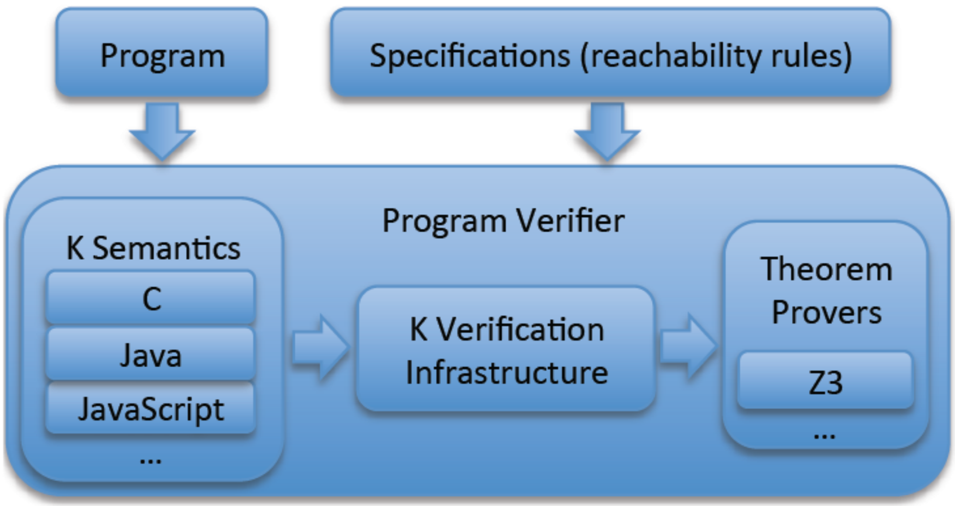
Can express *operational semantics*:

$$\langle\langle \mathtt{x} = i\,;\mathtt{s}\rangle_{\mathsf{k}}\ \langle \mathtt{x} \mapsto j,\ e\rangle_{\mathsf{env}}\ c\rangle_{\mathsf{cfg}}$$
$$\Rightarrow \langle\langle \mathtt{s}\rangle_{\mathsf{k}}\ \langle \mathtt{x} \mapsto i,\ e\rangle_{\mathsf{env}}\ c\rangle_{\mathsf{cfg}}$$

Can express *Hoare triples*: $\{\psi\}\,\mathsf{code}\,\{\psi'\}$

$$\exists X_{\mathsf{code}}(\langle \mathsf{code}, \sigma_{X_{\mathsf{code}}}\rangle \wedge \psi_X)$$
$$\Rightarrow \exists X_{\mathsf{code}}(\langle \mathtt{skip}, \sigma_{X_{\mathsf{code}}}\rangle \wedge \psi'_X)$$

K = (Best Effort) Implementation of Reachability Logic



Evaluated it with the existing semantics of C, Java, JavaScript, Ethereum VM, …, and many tricky programs. Performance acceptable.

Reachability Logic = Matching μ-Logic fragment

Semantically, $\varphi_1 \Rightarrow \varphi_2$ means that

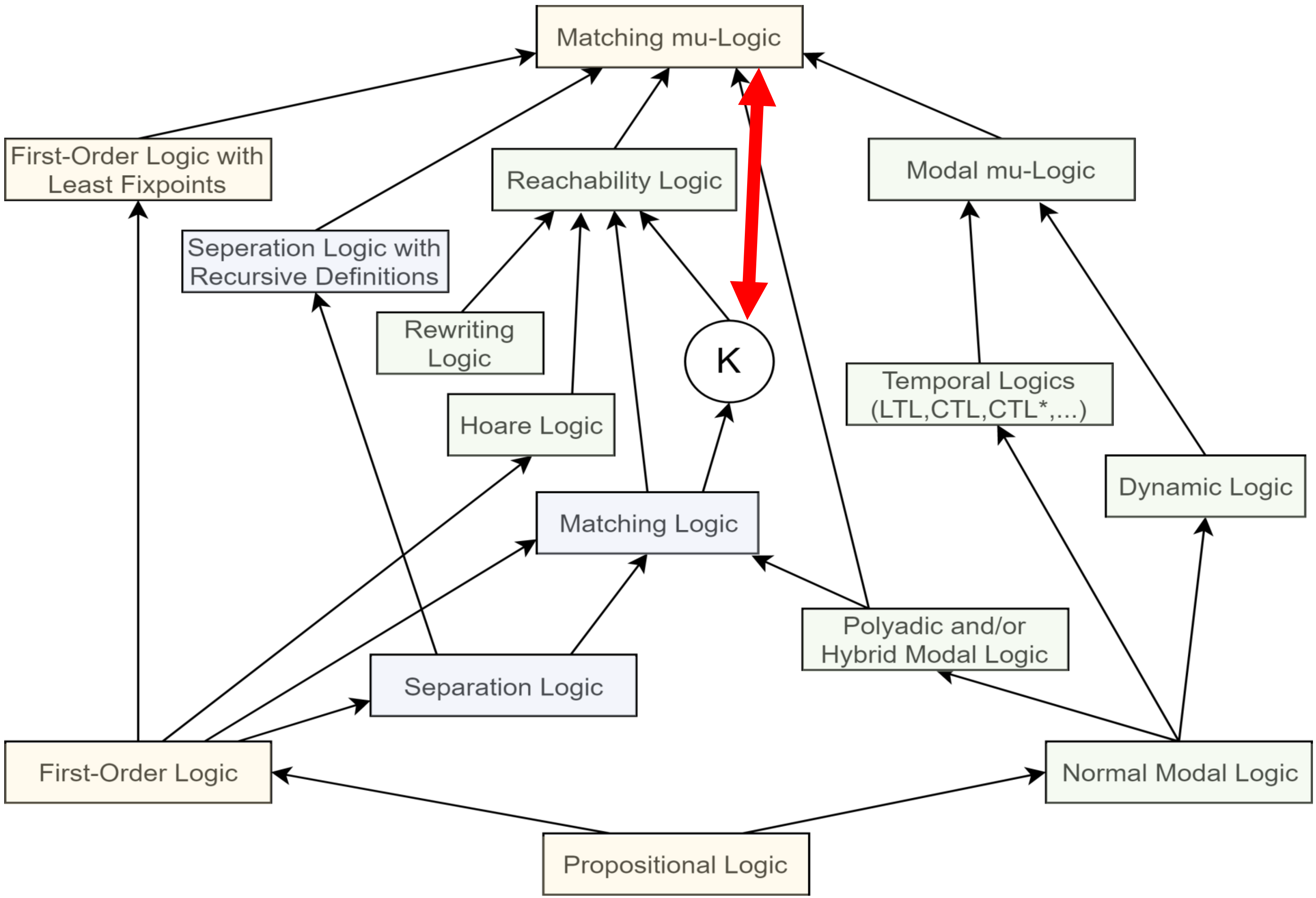
for all configurations γ_1 matching φ_1:

- either there exists γ_2 matching φ_2 such that γ_1 **reaches** γ_2 (in finite steps);
- or there exists an infinite execution path from γ_1;

$$\text{“weak eventually”} \quad \Diamond_w \varphi \equiv \nu X \,.\, \varphi \vee \bullet X$$

$$\text{“reachability”} \quad \varphi_1 \Rightarrow \varphi_2 \equiv \varphi_1 \rightarrow \Diamond_w \varphi_2$$

Matching $\boldsymbol{\mu}$-Logic as unifying semantic foundation



Completeness?

$\lceil \_ \rceil_s^{s'} \in \Sigma_{s,s'}$

$\lceil x{:}s \rceil_s^{s'}$

On the fragment *without* fixpoints μ-binder:

- **Theorem**. For theory Γ that contains *definedness*

$$\Gamma \vDash \varphi \text{ implies } \Gamma \vdash \varphi.$$

- **Theorem**. For the empty theory $\emptyset$,

$$\emptyset \vDash \varphi \text{ implies } \emptyset \vdash \varphi.$$

nontrivial extension of hybrid modal logic

- **Conjecture**. For all theories Γ,

$$\Gamma \vDash \varphi \text{ implies } \Gamma \vdash \varphi.$$

called *global completeness*

On the fragment *without* FOL $\exists$-binders:

- **Conjecture**. $\emptyset \vDash \varphi$ implies $\emptyset \vdash \varphi$.
- **Conjecture.** Fragment is *decidable*.

extension of modal μ-logic

On *full* Matching μ-logic:

- **Conjecture**. $\boxed{\Gamma \vDash_{\text{Henkin}} \varphi}$ implies $\Gamma \vdash \varphi$.

Henkin semantics or General semantics