Connecting Constrained Constructor Patterns and Matching Logic

Xiaohong Chen[1], Dorel Lucanu[2], Grigore Rosu[1]

[1]University of Illinois at Urbana-Champaign, USA

[2]Alexandru Ioan Cuza University of Iasi, Romania

{xc3,grosu}@illinois.edu; dlucanu@info.uaic.ro

Introduction

- Jose Meseguer: "I strongly conjecture that there is a deep connection between **matching logic** and the **constrained constructor patterns**. It would be great to better understand the details of such a connection."
- A constrained constructor pattern is a pair $u \mid \varphi$ consisting of
  - a constructor term u
  - a constraint φ (quantifier-free FOL formula)
- A matching logic unifies the FOL syntax of formulas and terms, allowing us to build formulas called **patterns** that are matched by elements.
  - E.g., $\varphi_1 \wedge \varphi_2$ is matched by those matching φ_1 and φ_2.
- **Main idea:**
  Constrained constructor pattern $u \mid \varphi$ is the same as the pattern $u \wedge \varphi$.

Challenges and Contributions

- How to define the OS spec $(\Sigma, E \cup B)$ using a matching logic theory?
- How to capture the canonical term model of the OS spec?
- Is $u \mid \varphi$ the same as matching logic pattern $u \wedge \varphi$? To what extends?
    - Semantically, $\|u \mid \varphi\| = \exists \vec{x}. (u \wedge \varphi)$ is the set of terms u where φ holds.
- This semantic equivalence reduces properties of constrained constructor patterns into matching logic theorems and/or meta-theorems.

Challenges and Contributions

- Studying the relation between constrained constructor patterns and matching logic has a mutual benefit:
  - Matching logic can benefit from borrowing the computationally efficient reasoning modulo axioms.
  - The theory of constrained constructor patterns can get more expressiveness from its formalization in matching logic.
- We use an example (Sieve of Eratosthenes) to present matching logic and its connection with constrained patterns in an intuitive way.

An Example: Sieve of Eratosthenes

Source: https://www.pinterest.com/pin/461619649106536o1/ by Robert Butterfield

An Example: Sieve of Eratosthenes

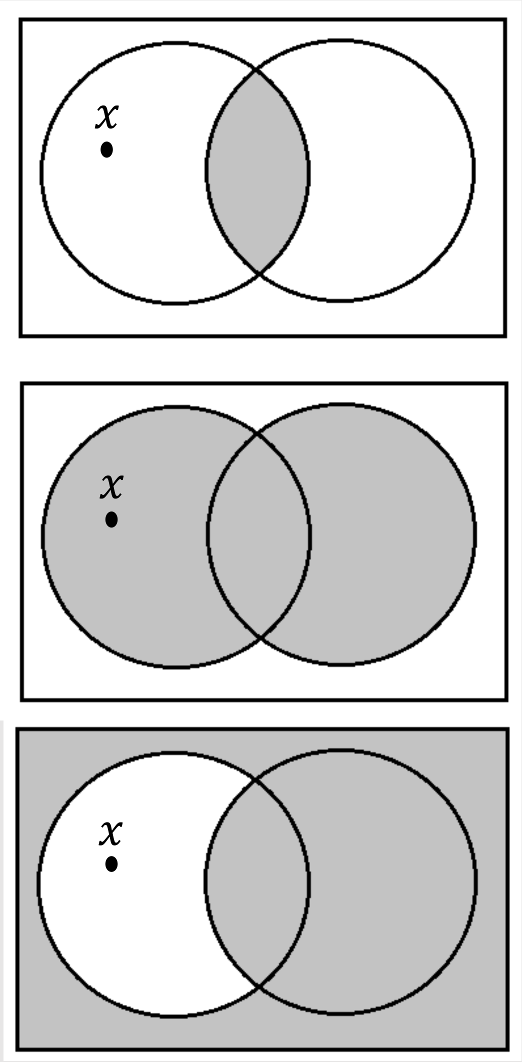
In **constrained constructor patterns**:

- configurations: $\text{cfg}(l_1, l_2) \mid primes?\,(l_1) \wedge sorted?\,(l_2) \wedge lt?\,(l_1, l_2)$
- transition: $\text{cfg}(l_1, cons(k, l_2)) \Rightarrow \text{cfg}\big(cons(k, l_1), removeMult(l_2, k)\big)$
- sorts and subsorting: $Int,\ \ List$
- constructors: $nil,\ \ cons$
- defined functions/predicates: $removeMult,\ \ primes?,\ \ sorted?\,, lt?,$
- subsumption (for rewriting, defined at the meta-level)
  $\|\text{cfg}(l_1, l_2) \mid primes?\,(l_1) \wedge sorted?\,(l_2) \wedge lt?\,(l_1, l_2) \wedge l_2 \neq nil\|$
  $\subseteq \|\text{cfg}(l_1, cons(k, l_2))\|$
- unification/conjunction (for narrowing, defined at the meta-level)
  $\|\text{cfg}(l_1, l_2) \mid primes?\,(l_1) \wedge sorted?\,(l_2) \wedge lt?\,(l_1, l_2)\|$
  $\cap \|\text{cfg}(l_1, cons(k, l_2))\|$

An Example: Sieve of Eratosthenes

In **matching logic**:

- Formulas are called patterns
- **A pattern is matched by a set of elements.**
- Conjunction=Intersection, Disjunction=Union, Negation=Complement, etc.
- Definedness (matched by at least one element) and totality (matched by all elements):
  - $\lceil \varphi \rceil$ is total iff φ is defined; otherwise, it is empty.
  - $\lfloor \varphi \rfloor$ is total iff φ is total; otherwise, it is empty.
- Membership and inclusion are expressed by patterns:
  - $x \in \varphi \equiv \lceil x \wedge \varphi \rceil$
  - $\varphi_1 \subseteq \varphi_2 \equiv \lfloor \varphi_1 \rightarrow \varphi_2 \rfloor$



$\varphi_1 \wedge \varphi_2$

$\varphi_1 \vee \varphi_2$

$\varphi_1 \rightarrow \varphi_2$

An Example: Sieve of Eratosthenes

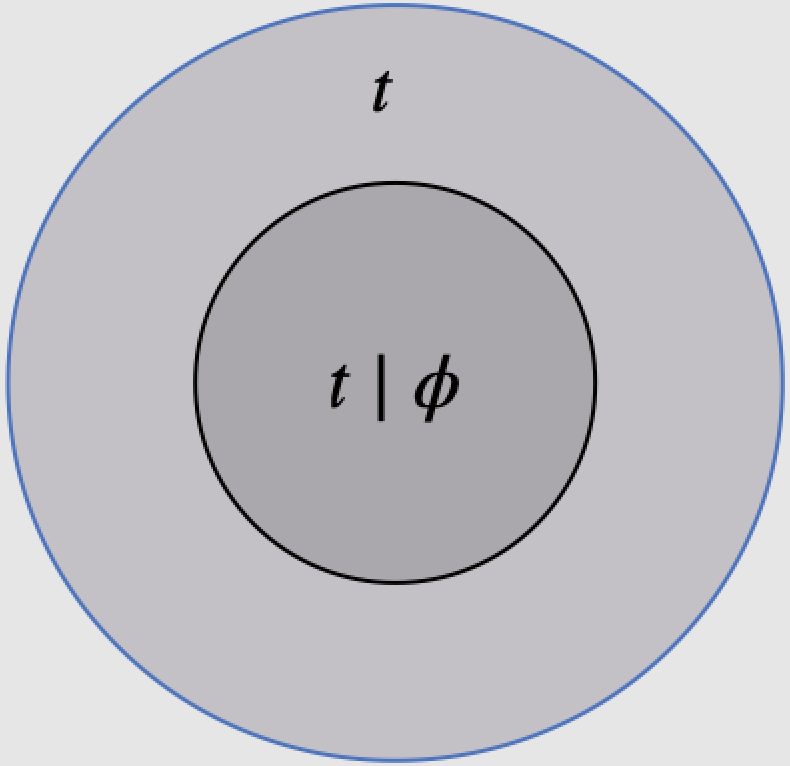
- Sorts are (matching logic) symbols satisfying proper axioms:
  - $Int \in Sort, List \in Sort$
  - $x{:}Int \equiv x \wedge (x \in [\![Int]\!])$
- Functions/predicates are symbols satisfying proper axioms:
  - $\exists l.\, nil = l$
  - $\forall n{:}Int.\, l'{:}List.\, \exists l.\, conc(n, l') = l$
- Constructors: “no-junk no-confusion”.
  - $[\![List]\!] = \mu L.\, nil \vee \exists n{:}Int.\, \exists l{:}List\, .\, cons(n, l)$ “inductive principle”
  - $cons(n_1, l_1) = cons(n_2, l_2) \rightarrow n_1 = n_2 \wedge l_1 = l_2$
- Constrained constructor pattern $\|u \mid \varphi\| \equiv \exists \vec{x}.\, (u \wedge \varphi)$
- Subsumption expressed by a pattern:

$$\|\mathrm{cfg}(l_1, l_2) \mid primes?\,(l_1) \wedge sorted?\,(l_2) \wedge lt?\,(l_1, l_2) \wedge l_2 \neq nil\|$$
$$\subseteq \|\mathrm{cfg}(l_1, cons(k, l_2))\|$$

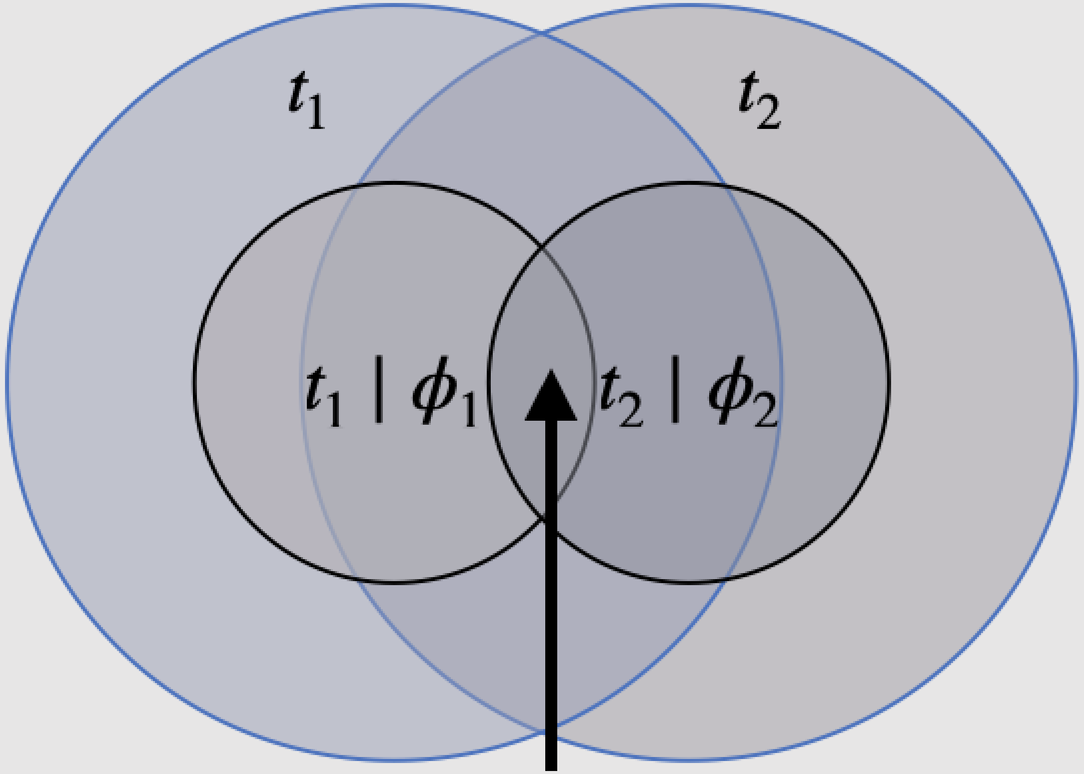
Quick Overview: Constrained Constructor Patterns

- An order-sorted signature $\Sigma = (S, \leq, F)$, where $F = \Omega \cup \Delta$.
  - Ω: the set of constructors.
  - Δ: the set of defined functions.
- An order-sorted (equational) theory $(\Sigma, B \cup E)$.
  - B: the set of basic axioms like associativity, commutativity, etc.
  - E: the set of other axioms, often oriented as rewrite rules, written $\vec{E}$.
- $u \mid \varphi$, where $u \in T_\Omega(X)$ is a constructor (with variables in X).
- $\|u \mid \varphi\| = \left\{canf(u\rho) \mid \rho: X \to T_\Omega, C_{\Sigma/E,B} \vDash \varphi\rho\right\}$

Constrained Constructor Patterns



$t \mid \phi$



$t_1 \mid \phi_1 \wedge t_2 \mid \phi_2$

Matching Logic Patterns

- Matching logic signature Σ is a set of symbols.
- Matching logic patterns:
  - $\varphi ::= \sigma \in \Sigma \mid x \mid X \mid \varphi_1\varphi_2 \mid \bot \mid \varphi_1 \to \varphi_2 \mid \exists x.\varphi \mid \mu X.\varphi$
  - x: element variable ranging over elements (like FOL variables).
  - X: set variable ranging over sets (like propositional variables in modal logic).
  - $\varphi_1\varphi_2$: application (left associative).
  - $\bot$ and $\varphi_1 \to \varphi_2$: propositional connectives (others can be derived).
  - $\exists x.\varphi$: FOL-quantification.
  - $\mu X.\varphi$: least fixpoint (like in modal μ-calculus).
- Example: $\exists x.\, cons\big(x, cons(x, l)\big)$

Matching Logic Pattern Semantics

- Matching logic has a **pattern matching semantics**.
- $[\![\varphi]\!]$ is the set of elements that **match** φ.
- Symbol interpretation $\sigma_M \subseteq M$.
- Application interpretation $app_M: M \times M \to \mathcal{P}(M)$
- Given valuation ρ with $\rho(x) \in M$ and $\rho(M) \subseteq M$:

- $[\![x]\!]_\rho = \{\rho(x)\}$
- $[\![X]\!]_\rho = \rho(X)$
- $[\![\sigma]\!]_\rho = \sigma_M \subseteq M$
- $[\![\varphi_1 \varphi_2]\!]_\rho = \bigcup_{a_1 \in [\![\varphi_1]\!]_\rho, a_2 \in [\![\varphi_2]\!]_\rho} app_M(a_1, a_2)$

- $[\![\bot]\!]_\rho = \emptyset$
- $[\![\varphi_1 \to \varphi_2]\!]_\rho = M \setminus \left([\![\varphi_1]\!]_\rho \setminus [\![\varphi_2]\!]_\rho\right)$
- $[\![\exists x. \varphi]\!]_\rho = \bigcup_{a \in M} [\![\varphi]\!]_{\rho[a/x]}$
- $[\![\mu X. \varphi]\!]_\rho = \mathbf{lfp}\left(\lambda A \mapsto [\![\varphi]\!]_{\rho[A/X]}\right)$

Defining OSA in Matching Logic

	Order-Sorted Algebra	**Matching Logic**
Signature	$\Sigma = (S, \leq, F)$	$\Sigma^{\mathsf{ML}} = \{\lceil \_ \rceil, [\![\_]\!], Sort\} \cup S \cup F$
Axioms	OSA metalanguage	ML axioms
	$s \in S$	$s \in Sort$ $\exists y.\, s = y$ $[\![s]\!] \neq \bot$
	$s \leq s'$	$[\![s]\!] \subseteq [\![s']\!]$
	$f \in F_{s_1 \ldots s_n, s}$	$f \colon s_1 \times \cdots \times s_n \to s$
	$x{:}s$ (sorted variable)	$x \in [\![s]\!]$
Terms	t	t^{ML}
	$f(t_1, \ldots, t_n)$	$f\, t_1 \cdots t_n$
Sentences	φ $\{x_1, \ldots, x_n\}$ = variables in φ	φ^{ML} $x_1 \in [\![s_1]\!] \wedge \cdots \wedge x_n \in [\![s_n]\!] \to (\varphi = \top)$
Model	A	$M \equiv A^{\mathsf{ML}}$
	$f_A : A_{s_1} \times \cdots A_{s_n} \to A_s$ $f_A(a_1, \ldots, a_n)$	$f_M\, a_1 \cdots a_n = \{f_A(a_1, \ldots, a_n)\}$

Fig. 1. Given an order-sorted signature $\Sigma = (S, \leq, F)$ and a Σ-OSA A, we derive a ML signature Σ^{ML} and a corresponding Σ^{ML}-model $M \equiv A^{\mathsf{ML}}$.

Constrained Constructor Pattern Subsumption: A Matching Logic View

- Question: Whether $[\![\, u \mid \varphi \,]\!] \subseteq \bigcup_{i\in I} [\![v_i \mid \psi_i]\!]$?
- The answer is given by $(E_\Omega \cup B_\Omega)$-matching:
- $C_{\Sigma,B,E} \vDash \varphi \to \bigvee_{(i,\beta)\in \mathrm{MATCH}(u,\{v_i \mid i\in I\})} \psi_i\beta$
- We can put the above result as an internal theorem in matching logic.

Theorem 2. *The following holds:*

$$C^{\mathsf{ML}}_{\Sigma/E,B} \models \left(\exists \overline{x}{:}\overline{s}.\, u^{\mathsf{ML}} \wedge \varphi^{\mathsf{ML}}\right) \subseteq \left(\bigvee_{i\in I} \exists \overline{y_i}{:}\overline{s_i}.\, v_i^{\mathsf{ML}} \wedge \psi_i^{\mathsf{ML}}\right) \leftrightarrow \left(\varphi^{\mathsf{ML}} \to \bigvee_{(i,\beta)\in\mathrm{MATCH}(u,\{v_i \mid i\in I\})} (\psi_i^{\mathsf{ML}} \wedge \beta^{\mathsf{ML}})\right)$$

Conclusion

- We establish the relationship between two approaches that formalize state predicates of distributed systems:
    - constrained constructor patterns
    - matching logic
- There is a mutual benefit:
    - Matching logic can benefit from borrowing the computationally efficient reasoning modulo axioms.
    - The theory of constrained constructor patterns can get more expressiveness from its formalization as a fragment of matching logic.