How to Define Domain Specific Logics using Matching Logic

Dorel Lucanu

Alexandru Ioan Cuza University, Iași, Romania
dorel.lucanu@gmail.com

Matching logic [4,3,2] is a logic that allows to uniformly specify and reason about programming languages and properties of their programs. The syntax of matching logic is simple and compact:

$$\varphi ::= x \mid X \mid \sigma \mid \varphi_1\varphi_2 \mid \bot \mid \varphi_1 \to \varphi_2 \mid \exists.\, x\varphi \mid \mu X.\, \varphi$$

These eigth syntax constructs build matching logic formulas, called *patterns*, which, semantically speaking, can be matched by a set of elements. Patterns can match structures that are of certain shapes, satisfy certain dynamic properties, or meet certain logical constraints, usually all of these together.

The matching logic is endowed with a proof system that defines the provability relation, written $\Gamma \vdash_{\mathsf{ML}} \varphi$, which means that φ is formally derivable from the axioms in Γ, using the matching logic (Hilbert-style) proof system [2].

Many important logics and/or formal systems have been shown to be definable in matching logic as logical theories. In this we consider a different approach: starting from a matching logic theory specifying a domain D, we derive a logic (proof system) $\vdash_D$ that can be used independently to reason within D.

Next we present two matching logic theories: DEF and NAT. DEF introduces a new symbol def, called the *definedness* symbol, and defines the (Definedness) axiom. This symbol and its axioms is all it is needed to define predicates, its possible values being $\bot$ or $\top \equiv \neg\bot$. Then, the equality, the inclusion, and the membership are introduced as notations for patterns using the new symbol.

theory DEF
Symbols: def
Notations: $\lceil\varphi\rceil \equiv \mathsf{def}\ \varphi$
Axioms: (Definedness) $\forall x.\, \lceil x\rceil$
Notations:
(totality) $\lfloor\varphi\rfloor \equiv \neg\lceil\neg\varphi\rceil$
(equality) $\varphi_1 = \varphi_2 \equiv \lfloor\varphi_1 \leftrightarrow \varphi_2\rfloor$
(inlclusion) $\varphi_1 \subseteq \varphi_2 \equiv \lfloor\varphi_1 \to \varphi_2\rfloor$
(membership) $x \in \varphi \equiv x \subseteq \varphi$
endtheory

theory NAT Imports: DEF
Symbols: $\mathbb{N}, \mathit{zero}, \mathit{succ}$
Notations:
$0 \equiv \mathit{zero}, 1 \equiv \mathit{succ}\, 0, 2 \equiv \mathit{succ}\, 1, \ldots$
$\forall x{:}\mathbb{N}.\, \varphi \equiv x \in \mathbb{N} \to \varphi$
$\exists x{:}\mathbb{N}.\, \varphi \equiv x \in \mathbb{N} \wedge \varphi$
Axioms:
(Zero) $\exists x{:}\mathbb{N}.\, \mathit{zero} = x$
(Succ) $\forall x{:}\mathbb{N}.\, \exists y{:}\mathbb{N}.\, \mathit{succ}\ x = y$
(Succ.1) $\mathit{succ}\ \mathit{zero} \neq \mathit{zero}$
(Succ.2) $\forall x{:}\mathbb{N}.\, \forall y{:}\mathbb{N}.\, \mathit{succ}\, x = \mathit{succ}\, y \to x{=}y$
(Domain) $\mathbb{N} = \mu D.\, \mathit{zero} \vee \mathit{succ}\ D$
endtheory

The theory NAT specifies the natural numbers up to an isomorphism [1]. Note the 1-1 correspondence between the NAT axioms and the Peano axioms (see, e.g., https://www.britannica.com/science/Peano-axioms).

From the theory DEF we may derive the following the following inference system that can be used to reason about the equality and the membership:

$$\frac{}{\vdash_{\mathsf{DEF}} \varphi = \varphi} \qquad \frac{}{\vdash_{\mathsf{DEF}} \varphi_1 = \varphi_2 \wedge \psi[\varphi_1/x] \to \psi[\varphi_2/x]}$$

$$\frac{\vdash_{\mathsf{DEF}} \varphi}{\vdash_{\mathsf{DEF}} \forall x. x \in \varphi} x \notin FV(\varphi) \qquad \frac{\vdash_{\mathsf{DEF}} \forall x. x \in \varphi}{\vdash_{\mathsf{DEF}} \varphi} x \notin FV(\varphi)$$

$$\frac{}{\vdash_{\mathsf{DEF}} x \in y = (x = y)} \qquad \frac{}{\vdash_{\mathsf{DEF}} x \in \neg\varphi = \neg(x \in \varphi)}$$

$$\frac{}{\vdash_{\mathsf{DEF}} (x \in \varphi_1 \wedge \varphi_2) = (x \in \varphi_1) \wedge (x \in \varphi_2)} \qquad \frac{}{\vdash_{\mathsf{DEF}} (x \in \exists y.\, \varphi) = \exists y.\, (x \in \varphi)}$$

$$\frac{\vdash_{\mathsf{DEF}} \varphi_1 = \varphi_2}{\vdash_{\mathsf{DEF}} \varphi_2 = \varphi_1} \qquad \frac{\vdash_{\mathsf{DEF}} \varphi_1 = \varphi_2 \quad \vdash_{\mathsf{DEF}} \varphi_2 = \varphi_3}{\vdash_{\mathsf{DEF}} \varphi_1 = \varphi_3}$$

The derived inference system for NAT imports $\vdash_{\mathsf{DEF}}$ (the first rule), includes the axioms of NAT as rules (the next four rules), and rules for inductive reasoning (the last three rules), obtained using the (PreFixpoint) and (Knaster-Tarski) from the matching logic proof system [2]:

$$\frac{\vdash_{\mathsf{DEF}} \varphi}{\vdash_{\mathsf{NAT}} \varphi} \qquad \frac{}{\vdash_{\mathsf{NAT}} 0 \in \mathbb{N}} \qquad \frac{\vdash_{\mathsf{NAT}} \varphi \in \mathbb{N}}{\vdash_{\mathsf{NAT}} \mathit{succ}\ \varphi \in \mathbb{N}}$$

$$\frac{}{\vdash_{\mathsf{NAT}} \mathit{succ}\ 0 \neq 0} \qquad \frac{}{\vdash_{\mathsf{NAT}} \forall x{:}\mathbb{N}.\, \forall y{:}\mathbb{N}.\, \mathit{succ}\, x = \mathit{succ}}$$

$$\frac{\vdash_{\mathsf{NAT}} \mathit{succ}\ \varphi \to \varphi}{\vdash_{\mathsf{NAT}} \forall x{:}\mathbb{N}.\, (x \in \varphi \to \mathit{succ}\ x \in \varphi)} \qquad \frac{\vdash_{\mathsf{NAT}} \forall x{:}\mathbb{N}.\, (x \in \varphi \to \mathit{succ}\ x \in \varphi)}{\vdash_{\mathsf{NAT}} \mathit{succ}\ \varphi \to \varphi}$$

$$\frac{\vdash_{\mathsf{NAT}} \mathit{zero} \to \varphi \quad \vdash_{\mathsf{NAT}} \mathit{succ}\ \varphi \to \varphi}{\vdash_{\mathsf{NAT}} \mathbb{N} \to \varphi}$$

We obviously have $\vdash_{\mathsf{DEF}} \varphi$ implies $\mathsf{DEF} \vdash_{\mathsf{ML}} \varphi$ and $\vdash_{\mathsf{NAT}} \varphi$ implies $\mathsf{NAT} \vdash_{\mathsf{ML}} \varphi$.

We start with a gentle introduction of matching logic, including its proof system, and then we use several canonical examples of domains specified in matching logic to show how we can derive their specific logics. These examples will involve both the inductive and coinductive reasoning.

References

1. Xiaohong Chen, Dorel Lucanu, and Grigore Roşu. Initial algebra semantics in matching logic. Technical Report http://hdl.handle.net/2142/107781, University of Illinois at Urbana-Champaign, July 2020. submitted.
2. Xiaohong Chen, Dorel Lucanu, and Grigore Roşu. Matching logic explained. *Journal of Logical and Algebraic Methods in Programming*, 120:100638, 2021.
3. Xiaohong Chen and Grigore Roşu. Matching mu-logic. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'19) (to appear)*, 2019.
4. Grigore Roşu. Matching logic. *Logical Methods in Computer Science*, 13(4):1–61, December 2017.