Horn2VMT: Translating Horn Reachability into Transition Systems

Denis Bueno
University of Michigan
dlbueno@umich.edu

Karem A. Sakallah
University of Michigan
karem@umich.edu

1 Introduction

Relational transition system models have for decades been the bread and butter representation of hardware and software model checking. The VMT format[1] provides a means of describing sequential model checking problems using SMT-LIB-compatible combinational formulas. MathSAT [4], EUForia [3], and other tools support VMT natively, but unfortunately they do not support Horn clause formats. While it is well known that linear Horn clauses can be expressed as transition systems, we are not aware of a precise description in the literature. This paper contributes a procedure, proof of correctness, and prototype implementation for translating linear Horn clauses into VMT. Our procedure is able to convert SV-COMP [2] and SeaHorn [6] benchmarks into VMT for use with various model checkers. Our prototype implementation will be available at `https://github.com/dbueno/horn2vmt`.

2 Background

Consider a first-order language with equality with signature $\mathscr{S}$ and two common sorts, BOOLS and INTS. Our language also uses a set of relation symbols $\mathscr{R}$. We say a relation R of arity $n = |R|$ has *n places*; each place refers to a dimension of the relation. For instance, in the formula $R(1,2)$, 1 is in the first place of the relation and 2 is in the second. The semantics of these formulas is standard. A *Horn clause* or *rule* is a universally quantified formula with a *body* part and a *head* part:

$$\forall x_1,\ldots,x_m. \underbrace{\bigwedge_{k=1}^{j} P_k(\bar{x}_k) \wedge \phi(x_1,\ldots,x_m)}_{\text{body}} \Rightarrow \mathsf{head} \tag{1}$$

where for every k, $P_k \in \mathscr{R}$ is an uninterpreted predicate symbol, $\bar{x}_k \subseteq \{x_1,\ldots,x_m\}$, and $|\bar{x}_k| = |P_k|$ [8]. The constraint ϕ is a formula over interpreted atoms. head must either be an application of an uninterpreted predicate or an interpreted formula. In this paper, $j = 1$, which corresponds to linear Horn clauses.

A *transition system* [5, 2] is a tuple (X,Y,I,T) consisting of (non-empty) sets of *state variables* $X = \{x_1,\ldots,x_n\}$ and corresponding *next-state variables* $X' = \{x'_1, x'_2, \ldots, x'_n\}$, a (possibly empty) set of *input variables* $Y = \{y_1,\ldots,y_m\}$, and two formulas: $I(X)$, the *initial states*, and $T(X,Y,X')$, the transition relation. For every formula σ, the set $\mathrm{Vars}(\sigma)$ denotes the set of state variables free in σ. The X in $\sigma(X)$ indicates that the free variables in σ are drawn solely from the set X; we may omit the argument and

[1]`https://es-static.fbk.eu/tools/nuxmv/index.php?n=Languages.VMT`
[2]`https://sv-comp.sosy-lab.org/2020/`

Submitted to:
HCVS 2020

write σ. Prime$(\sigma(X))$ stands for the formula $\sigma(X')$, that is, all state variable occurrences are replaced with primed (i.e., next state) state variables. In this paper, we will also provide a *property* formula $P(X)$ that we wish to prove is an invariant of T. We write $\sigma \xrightarrow{T} \omega$ if each state in σ transitions to some state in ω under T, i.e., $\sigma \wedge T \models \omega'$. A transition system *execution* is a (possibly-infinite) sequence of transitions $\sigma_0 \xrightarrow{T} \sigma_1 \xrightarrow{T} \sigma_2, \ldots$ such that $\sigma_0 \models I$. Every *reachable* state occurs in some execution.

3 Translating Horn to Transition Systems

We begin with a motivating example: a program expressed as Horn clauses which we will translate into a transition system. The program's Horn clauses are defined over interpreted symbols $\{<, +\}$ and relation symbols $\mathscr{R} = \{E, L, M, U\}$; relations model the program's control locations and state (see [1] for program encoding details):

$$\text{true} \Rightarrow E \tag{2}$$
$$E \Rightarrow L(0) \tag{3}$$
$$\forall x.\ L(x) \wedge (x < 5) \Rightarrow L(x+3) \tag{4}$$
$$\forall x.\ L(x) \wedge \neg(x < 5) \Rightarrow M(x) \tag{5}$$
$$\forall x.\ M(x) \wedge \neg(x < 7) \Rightarrow U \tag{6}$$

In order to use a VMT-capable model checker to answer the question, "is the relation U inhabited?" we show below how to encode Horn clauses (2)–(6) as a transition system $\mathbb{A} = (X, \emptyset, I, T)$ where $X = \{\ell_E, \ell_L, \ell_M, \ell_U, P_{L,1}, P_{M,1}\}$, $I = (\neg\ell_E \wedge \neg\ell_L \wedge \neg\ell_M \wedge \neg\ell_U)$, and T is defined below. The property $P = (\neg\ell_U)$. The variables $\ell_E, \ell_L, \ell_M, \ell_U$ are Boolean *relation variables* that correspond to the relation symbols in the Horn clauses. $P_{L,1}, P_{M,1}$ are integer *place variables* that correspond to elements inhabiting Horn clause relations. We use the function $\pi[S] \equiv (\bigwedge_{x \in S} x' = x)$ to express that values of variables in S are preserved, i.e., they don't change. $\mathbb{A}$'s transition relation T, then, is defined as the disjunction of the following constraints:

$$(\ell'_E \wedge \pi[X \setminus \{\ell_E\}]) \tag{2*}$$
$$(\ell_E \wedge \ell'_L \wedge (P'_{L,1} = 0) \wedge \pi[X \setminus \{\ell_L, P_{L,1}\}]) \tag{3*}$$
$$(\ell_L \wedge (P_{L,1} < 5) \wedge \ell'_L \wedge (P'_{L,1} = P_{L,1} + 3) \wedge \pi[X \setminus \{\ell_L, P_{L,1}\}]) \tag{4*}$$
$$(\ell_L \wedge \neg(P_{L,1} < 5) \wedge \ell'_M \wedge (P'_{M,1} = P_{L,1}) \wedge \pi[X \setminus \{\ell_M, P_{M,1}\}]) \tag{5*}$$
$$(\ell_M \wedge \neg(P_{M,1} < 7) \wedge \ell'_U \wedge \pi[X \setminus \{\ell_U\}]) \tag{6*}$$

Each disjunct of T corresponds to a single Horn rule; (2*) corresponds to (2), (3*) to (3), and so on. It is possible in $\mathbb{A}$ to reach states $(\ell_L \wedge P_{L,1} = \mathbf{0})$, $(\ell_L \wedge P_{L,1} = \mathbf{3})$, and $(\ell_L \wedge P_{L,1} = \mathbf{6})$[3], meaning $\{0, 3, 6\} \subseteq L$. Moreover, every reachable state satisfies P, implying that clauses (2)–(6) are not satisfiable.

3.1 General Translation

We now present our general translation from set of n linear Horn clauses over $\mathscr{R}$ into a transition system $\mathbb{G} = (X, Y, I, T)$ such that a reachability query (i.e., whether a relation is derivable) holds if and only if

[3]Boldface indicates the only difference among the three formulas.

a safety property on T fails. Without loss of generality, we assume each Horn clause head is a relation occurence and that we wish to solve a single query, a 0-ary relation U.[4]

The states of the resulting transition system are defined over a finite set of state variables $X = \{\ell_R \mid R \in \mathscr{R}\} \cup \{P_{R,i} \mid R \in \mathscr{R}, 1 \leq i \leq k$ where R has arity $k\}$: Boolean relation variables ℓ_R and place variables $P_{R,i}$; and fresh primary inputs of the form $\{Y_{j,x} \mid 1 \leq j \leq n\}$, as explained below. Horn clauses are translated with the help of the syntactic mapping $[\![\cdot]\!]$ defined over quantifier-free formulas. Let (possibly-subscripted) e, f, g, s, t be expressions:

$$[\![x]\!] \equiv Y_{i,x} \qquad \text{quantified variable } x \text{ occurs in rule } i \tag{7}$$

$$[\![R(x_1,\ldots,x_k)]\!] \equiv \ell_R \wedge P_{R,1} = [\![x_1]\!] \wedge \cdots \wedge P_{R,k} = [\![x_k]\!] \tag{8}$$

$$[\![F(e_1,\ldots,e_k)]\!] \equiv F([\![e_1]\!],\ldots,[\![e_k]\!]) \qquad \text{for interpreted } F \tag{9}$$

$$[\![s = t]\!] \equiv [\![s]\!] = [\![t]\!] \qquad [\![f \wedge g]\!] \equiv [\![f]\!] \wedge [\![g]\!] \qquad [\![\neg f]\!] \equiv \neg [\![f]\!] \tag{10}$$

During translation, T is treated as a disjunction. For every Horn clause with atom A in its body, $(\forall x_1,\ldots,x_k.A \wedge \phi \Rightarrow \mathsf{head})$, add the following disjunct to T: $[\![A \wedge \phi]\!] \wedge \mathrm{Prime}([\![\mathsf{head}]\!]) \wedge \pi[X \setminus \mathrm{Vars}(\mathsf{head})]$. The initial state $I = (\bigwedge_{\ell_R} \neg \ell_R)$ and the property $P = \neg \ell_U$. By cases it can be tediously but straightforwardly shown that if a single Horn clause is satisfiable, the resulting transition system has a corresponding satisfying assignment.

3.2 Proof of Correctness

The transition system has the property that the state ℓ_R is reachable in T if and only if relation R is derivable under the Horn clauses.

Direction ($\Leftarrow$): We proceed by induction on the length of the derivation of R. All relations are initially empty; this is correctly modeled by the definition of I. Length-1 derivations use a single Horn clause whose body contains no uninterpreted relations with head R. Such a clause translates to a transition that can be similarly satisfied without relation variables and which also satisfies ℓ'_R.

Consider a relation R derivable in $n+1$ steps. Its last derivation step involves some rule with head R; by the induction hypothesis, a state satisfying the body of this rule is reachable in T. Examining T and the definition of $[\![\cdot]\!]$ allows us to conclude that the next state satisfies ℓ'_R.

Direction ($\Rightarrow$): We proceed by induction on the number of transitions. Initially, no relation variable is reached, due to I's definition. Next, suppose that $I \wedge T \wedge \ell'_R$ is satisfied. T guarantees that the body of the Horn rule corresponding to the transition is satisfied, so R is derivable.

Assume that some state $\sigma \wedge \ell'_R$ is reachable after $n+1$ steps. By the induction hypothesis, the current state, which took n steps to reach, has a Horn derivation. The current state corresponds to the body of a rule with head R, since the only transitions to ℓ'_R in T correspond to such rules. Therefore R is derivable. QED.

Our translation takes linear time and uses space linear in the number of Horn clauses and the relation symbols. The number of state variables is proportional to the sum of the relation symbol arities. In addition, an n-step Horn derivation corresponds to an $O(n)$-length execution.

[4]If we wish to solve a more complex query, for example $P(1,4,x)$ (for $P \in \mathscr{R}$), simply modify the Horn clauses as follows: add a fresh relation symbol U to $\mathscr{R}$ and a rule $(\forall x.P(1,4,x) \Rightarrow U)$.

References

[1] Nikolaj Bjørner, Arie Gurfinkel, Kenneth L. McMillan & Andrey Rybalchenko (2015): *Horn Clause Solvers for Program Verification*. In Lev D. Beklemishev, Andreas Blass, Nachum Dershowitz, Bernd Finkbeiner & Wolfram Schulte, editors: *Fields of Logic and Computation II - Essays Dedicated to Yuri Gurevich on the Occasion of His 75th Birthday*, *Lecture Notes in Computer Science* 9300, Springer, pp. 24–51, doi:10.1007/978-3-319-23534-9_2.

[2] Aaron R. Bradley & Zohar Manna (2007): *Checking Safety by Inductive Generalization of Counterexamples to Induction*. In: *Formal Methods in Computer-Aided Design*, IEEE Computer Society, pp. 173–180, doi:10.1109/FAMCAD.2007.15.

[3] Denis Bueno & Karem A. Sakallah (2019): EUFORIA*: Complete Software Model Checking with Uninterpreted Functions*. In Constantin Enea & Ruzica Piskac, editors: *Verification, Model Checking, and Abstract Interpretation - 20th International Conference, VMCAI 2019, Cascais, Portugal, January 13-15, 2019, Proceedings*, *Lecture Notes in Computer Science* 11388, Springer, pp. 363–385, doi:10.1017/S1471068415000204.

[4] Alessandro Cimatti, Alberto Griggio, Bastiaan Schaafsma & Roberto Sebastiani (2013): *The MathSAT5 SMT Solver*. In Nir Piterman & Scott Smolka, editors: *Proceedings of TACAS*, *LNCS* 7795, Springer, doi:10.1007/978-3-642-36742-7_7.

[5] Edmund M. Clarke, Orna Grumberg & David E. Long (1994): *Model Checking and Abstraction*. *ACM Trans. Program. Lang. Syst.* 16(5), pp. 1512–1542, doi:10.1145/3828.3837.

[6] Arie Gurfinkel, Temesghen Kahsai, Anvesh Komuravelli & Jorge A. Navas (2015): *The SeaHorn Verification Framework*. In Daniel Kroening & Corina S. Pasareanu, editors: *Computer Aided Verification*, *Lecture Notes in Computer Science* 9206, Springer, pp. 343–361, doi:10.1007/978-3-642-31424-7_42.

[7] Roope Kaivola & Thomas Wahl, editors (2015): *Formal Methods in Computer-Aided Design*. IEEE, doi:10.5555/2893529.

[8] Anvesh Komuravelli, Nikolaj Bjørner, Arie Gurfinkel & Kenneth L. McMillan (2015): *Compositional Verification of Procedural Programs using Horn Clauses over Integers and Arrays*. In Kaivola & Wahl [7], pp. 89–96, doi:10.5555/2893529.