Neural Network Action Policy Verification via Predicate Abstraction

Marcel Vinzent,[1] Marcel Steinmetz,[1] Jörg Hoffmann[1,2]

[1] Saarland University, Saarland Informatics Campus, Saarbrücken, Germany

[2] German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany

{vinzent, steinmetz, hoffmann}@cs.uni-saarland.de

SIC Saarland Informatics Campus

State Space Representation

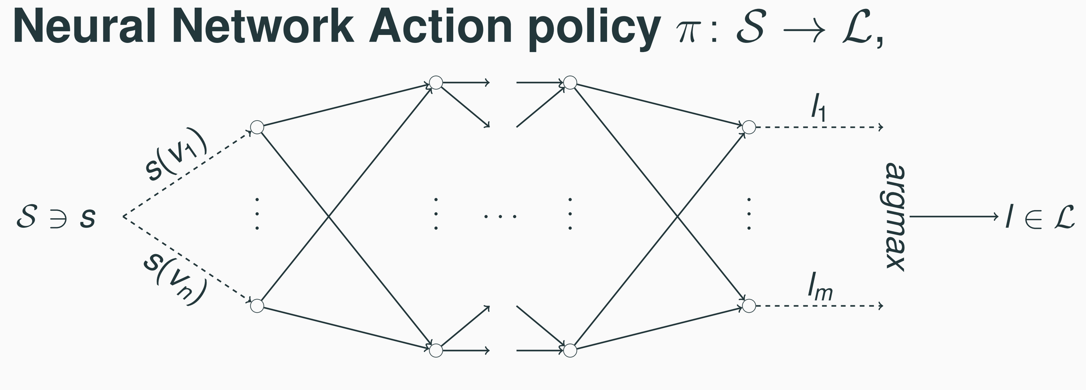
- **State variables** $\mathcal{V}$ with a bounded-integer domain.
- **Linear integer expressions** *Exp* over $\mathcal{V}$, $d_1 \cdot v_1 + \cdots + d_r \cdot v_r + c$ with $d_1, \ldots, d_r, c \in \mathbb{Z}$ and $v_1, \ldots, v_r \in \mathcal{V}$.
- **Linear integer constraints** and conjunctions thereof C, $e_1 \bowtie e_2$ with $e_1, e_2 \in Exp$ and $\bowtie \in \{\leq, =, \geq\}$.
- Labeled **operators** $\mathcal{O}$ of the form (g, l, u), with **action label** $l \in \mathcal{L}$, guard $g \in C$ and (partial) update $u \colon \mathcal{V} \to Exp$.

(Non-deterministic) state space LTS $\Theta = \langle \mathcal{S}, \mathcal{L}, \mathcal{T} \rangle$:

- **States** $\mathcal{S}$: complete state variable assignments over $\mathcal{V}$.
- **Transition** $(s, l, s') \in \mathcal{T}$ iff $s \models g$ (also: $s \models o$) and $s' = s[u(s)]$ (also: $s' = s[\![o]\!]$) for some operator $o = (g, l, u)$ in $\mathcal{O}$.
- State-dependent effects: (g_1, l, u_1) and (g_2, l, u_2) with $s_1 \models g_1$ but $s_2 \not\models g_1$ and $s_2 \models g_2$ but $s_1 \not\models g_2$.
- Action outcome non-determinism: $(s, l, s_1) \in \mathcal{T}$ induced by (g_1, l, u_1) and $(s, l, s_2) \in \mathcal{T}$ induced by (g_2, l, u_2).

Neural Network Action Policy

- **Neural Network Action policy** $\pi \colon \mathcal{S} \to \mathcal{L}$,



ReLU activation : $\max(x, 0)$

- **Policy restriction** $\Theta^\pi = \langle \mathcal{S}, \mathcal{L}, \mathcal{T}^\pi \rangle$ with $\mathcal{T}^\pi = \{(s, l, s') \in \mathcal{T} \mid \pi(s) = l\}$.
- **Safety property** $\rho = (\phi_0, \phi_U)$ with **start condition** $\phi_0 \in C$ and **unsafety condition** $\phi_U \in C$. π is **unsafe** iff there exist states $s_0 \models \phi_0$, $s_U \models \phi_U$ such that s_U is reachable from s_0 in Θ^π.

Policy Predicate Abstraction

- **Idea**: Predicate Abstraction (e.g., Graf and Saïdi (1997)) under π.
- Set of **predicates** $\mathcal{P} \subseteq C$.
- **Abstraction** of concrete state $s \in \mathcal{S}$: $s|_\mathcal{P} \in \mathcal{P} \to \{0, 1\}, p \mapsto p(s)$.
- **Concretization** of abstract state $s_\mathcal{P} \in \mathcal{P} \to \{0, 1\}$: $[s_\mathcal{P}] = \{s' \in \mathcal{S} \mid s'|_\mathcal{P} = s_\mathcal{P}\}$.
- The **policy predicate abstraction** of Θ^π over $\mathcal{P}$ is the LTS $\Theta^\pi_\mathcal{P} = \langle \mathcal{S}_\mathcal{P}, \mathcal{L}, \mathcal{T}^\pi_\mathcal{P} \rangle$, where $\mathcal{S}_\mathcal{P} = \mathcal{P} \to \{0, 1\}$ and $\mathcal{T}^\pi_\mathcal{P} = \{(s|_\mathcal{P}, l, s'|_\mathcal{P}) \mid (s, l, s') \in \mathcal{T}^\pi\}$ (transition preservation).

Motivation: Policy safety verification via (over-approximating) reachability analysis in $\Theta^\pi_\mathcal{P}$.

Transition problem of $\Theta^\pi_\mathcal{P}$:
$(s_\mathcal{P}, l, s'_\mathcal{P}) \in \mathcal{T}^\pi_\mathcal{P}$ iff for some operator $o = (g, l, u)$:
$\exists s \in [s_\mathcal{P}] \colon s \models o \wedge s[\![o]\!] \in [s'_\mathcal{P}] \wedge \pi(s) = l$.

(Without π) routinely encoded in SMT (e.g., *Z3* de Moura and Bjørner (2008)), **but** (under π) expensive due to non-linear NN activation.

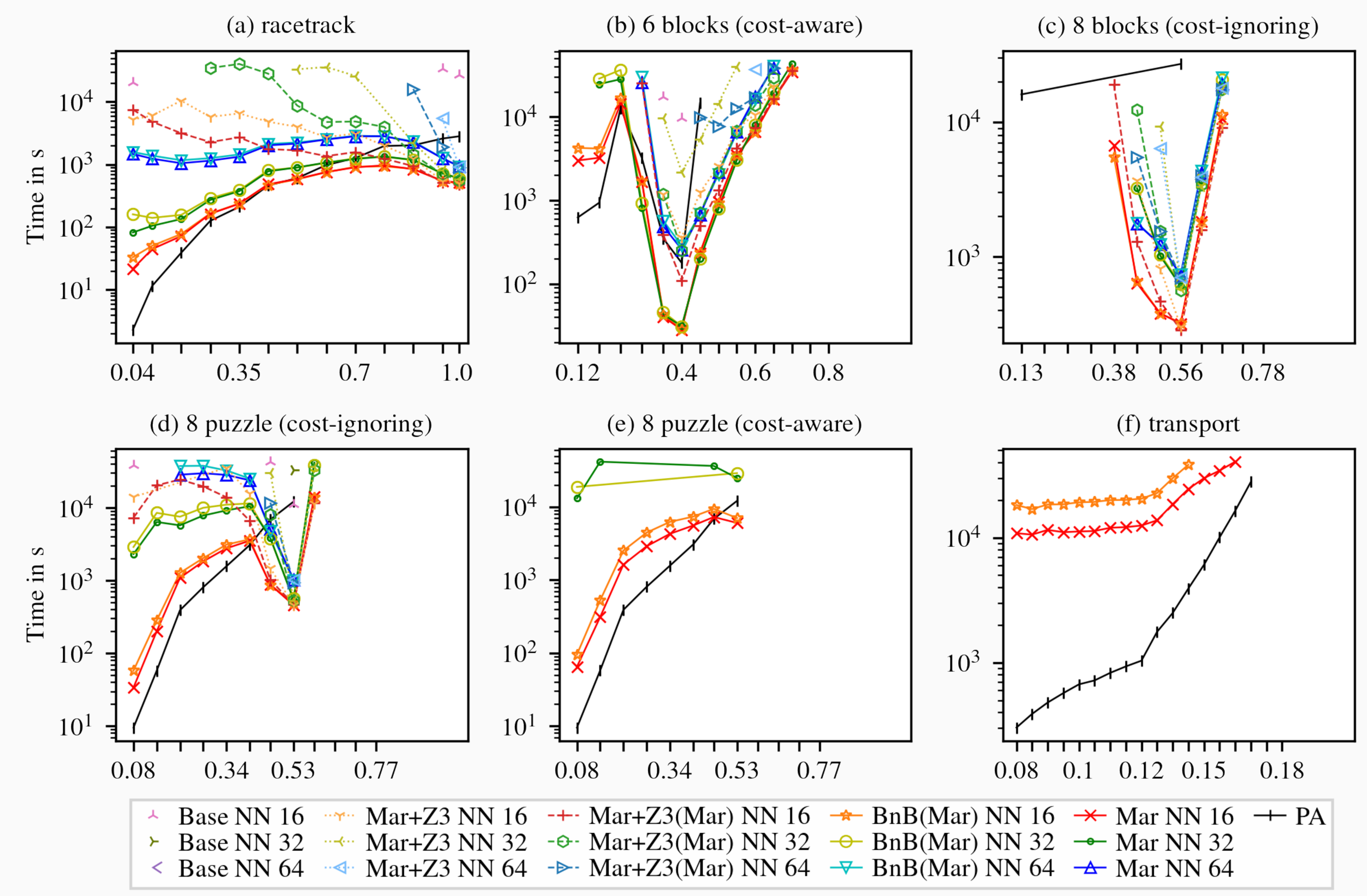
Algorithmic Enhancements: Relaxation + NN Analysis

- Tests on **necessary conditions**: label selection ($l \in \pi(s_\mathcal{P})$), operator applicability ($s_\mathcal{P} \models o$), label selection + operator applicability ($\exists s \in [s_\mathcal{P}] \colon \pi(s) = l \wedge s \models o$), and the non-policy-restricted transition ($\exists s \in [s_\mathcal{P}] \colon s \models o \wedge s[\![o]\!] \in [s'_\mathcal{P}]$). If unsat, one can skip all corresponding transition tests. Also: **non-policy-restricted** tests ($\cancel{\pi(s) = l}$) are much cheaper.
- **Continuously-relax** $\mathcal{V}$ to over-approximate the transition problem, plug in existing SMT solvers tailored to NN analysis (e.g., *Marabou* [Katz et al. (2019)]), branch & bound (over $\mathcal{V}$) to solve the exact transition problem.
- **Fixing activation cases** of ReLU towards SMT encoding: if $x \leq 0$ then $ReLU(x) = 0$, if $x \geq 0$ then $ReLU(x) = x$. Here: Extract bounds derived by *Marabou* to solve the exact transition problem.

Experiments

(on planning benchmarks modeled in JANI [Budde et al. (2017)])

- Compute $\Theta^\pi_\mathcal{P}$ reachable from ϕ_0.
- Scaling $|\mathcal{P}|$ as part of problem input (x-axis) for NN policies of different sizes (neurons per hidden layer).
- SMT via *Z3* [de Moura and Bjørner (2008)] & *Marabou* [Katz et al. (2019)].



$\rightarrow$ Algorithmic enhancements are required for practicality.

- Bounded Model Checking (BMC) competitor:

Benchmark \ NN	$L_U^{\min}$ 16|32|64	$t_U^{\min}$ 16|32|64	$L_{checked}^{\max}$ 16|32|64
Racetrack	3|3|3	36.9|40.1|316.5	12|11|7
6 Blocks (cost-awa)	-	-	6|5|4
8 Blocks (cost-ign)	-	-	5|5|4
8-puzzle (cost-ign)	2| - | -	72.8| - | -	7|3|0
8-puzzle (cost-awa)	-	-	3|3|0
Transport	1|1| -	57.0|20548.0| -	2|1|0

$\rightarrow$ PPA outperforms BMC competitor (& explicit enumeration).

References

Carlos E. Budde, Christian Dehnert, Ernst Moritz Hahn, Arnd Hartmanns, Sebastian Junges, and Andrea Turrini. JANI: Quantitative model and tool interaction. In *TACAS*, 2017.

Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *TACAS*, 2008.

S. Graf and H. Saïdi. Construction of abstract state graphs with PVS. In *CAV*, 1997.

Guy Katz, Derek A. Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljic, David L. Dill, Mykel Kochenderfer, and Clark Barrett. The Marabou framework for verification and analysis of deep neural networks. In *CAV*, 2019.