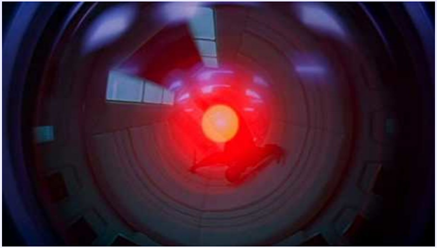
Learning and reasoning with

probabilistic satisfiability modulo theories

Paolo Morettin

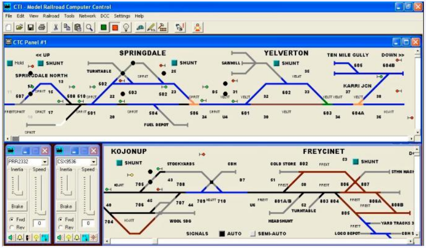
DTAI KU LEUVEN

Concerns on AI



Can we **enforce** and/or **verify**: safety, robustness, ... ?

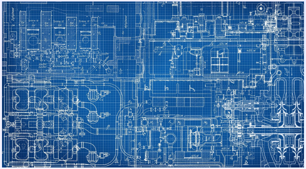
Concerns on ~~AI~~ other systems?



Not as concerned when we: catch a flight / ride a train / get a CT scan

Formal verification - the “traditional” approach

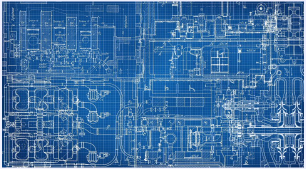
1) **Logical modelling** of the system S and property P:



“The elevator door doesn’t open if the brakes are off.”

Formal verification - the "traditional" approach

1) **Logical modelling** of the system S and property P:



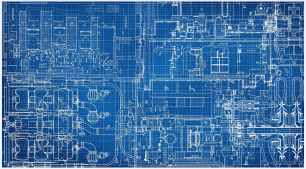
"The elevator door doesn't open if the brakes are off."

$$\downarrow \qquad\qquad \downarrow$$

$$S = ((... \vee ...)... \wedge ...) \qquad\qquad P = \neg(DoorOpen \wedge BrakesOff)$$

Formal verification - the "traditional" approach

1) **Logical modelling** of the system S and property P:



"The elevator door doesn't open if the brakes are off."

$$\downarrow \qquad\qquad\qquad \downarrow$$

$$S = ((\ldots \vee \ldots)\ldots \wedge \ldots) \qquad\qquad P = \neg(DoorOpen \wedge BrakesOff)$$

2) Verification of P is reduced to a **decision problem**:

$$SAT(S \wedge \neg P)? \qquad \rightarrow \qquad \{YES, NO\}$$

Formal verification - continuous/discrete models

Satisfiability Modulo Theories (SMT) = Logic + Specialized theories

$(A \vee read(write(a,i,v),i) = v) \wedge (read(a,j)) \rightarrow A)$ arrays

$(A \vee (a = b \rightarrow f(a) = g(b))) \wedge ((f(.) = g(.)) \rightarrow A)$ uninterpreted functions

...

$(A \vee (10x + 13y \leq z + 17/8)) \wedge ((z \geq 1/3) \rightarrow A)$ **linear algebra**

"If the elevator speed is greater than k, the alarm is on."

$$P = \neg(speed > k) \vee AlarmOn$$

When YES/NO are not enough

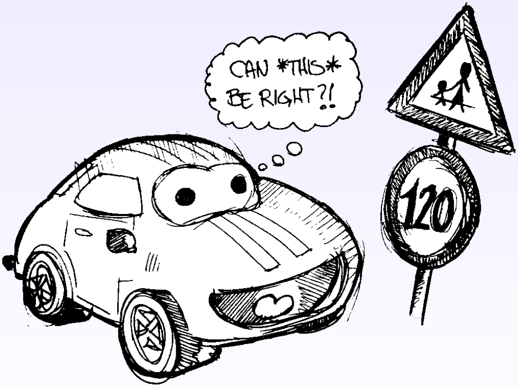
...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

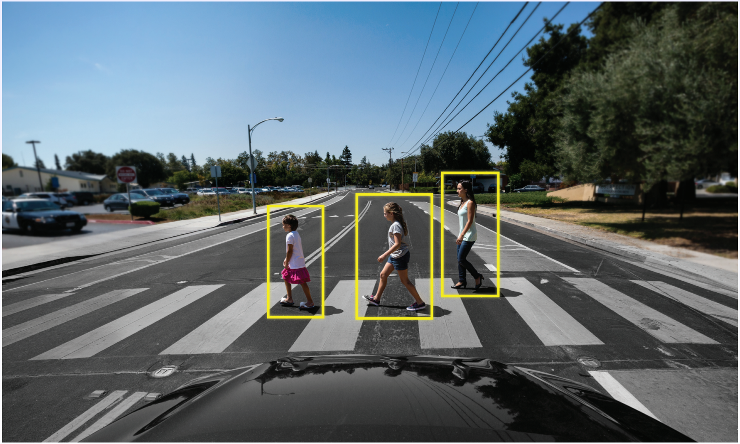


$$\neg((SpeedLimit = 120) \wedge SchoolCrossing)$$

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?



When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

- $Pr($$) = Pr($$) = \epsilon$ ... we don't care about these inputs!

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

- $Pr($$) = Pr($$) = \epsilon$ ... we don't care about these inputs!
- We need a probabilistic model of the environment $P(X)$

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

- $Pr($$) = Pr($$) = \epsilon$ ... we don't care about these inputs!
- We need a probabilistic model of the environment $P(X)$
- ...and verify whether properties hold with probability $\geq k$

When YES/NO are not enough

...What if:

- we need to *quantify* properties like robustness/fairness?
- the system is nondeterministic?
- the environment/input is high-dimensional and uncertain?

- $Pr($$) = Pr($$) = \epsilon$ ... we don't care about these inputs!
- We need a probabilistic model of the environment $P(X)$
- ...and verify whether properties hold with probability $\geq k$

probabilistic formal verification!

Weighted Model Counting to the rescue!

Weighted sum of the models of a logical formula

$$WMC(\Delta, w) = \sum_{\mu \models \Delta} w(\mu)$$

Weighted Model Counting to the rescue!

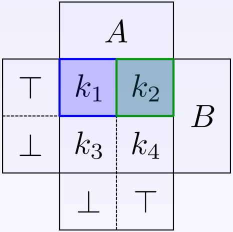
Weighted sum of the models of a logical formula

$$WMC(\Delta, w) = \sum_{\mu \models \Delta} w(\mu)$$

w factorizes over the literals:

$$w(\overbrace{\mu_1}^{A \wedge \neg B}) = k_1 = w(A) \cdot w(\neg B)$$

$$w(\overbrace{\mu_2}^{A \wedge B}) = k_2 = w(A) \cdot w(B)$$



Weighted Model Counting to the rescue!

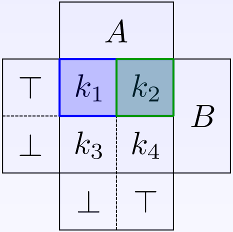
Weighted sum of the models of a logical formula

$$WMC(\Delta, w) = \sum_{\mu \models \Delta} w(\mu)$$

w factorizes over the literals:

$$w(\overbrace{\mu_1}^{A\wedge\neg B}) = k_1 = w(A)\cdot w(\neg B)$$

$$w(\overbrace{\mu_2}^{A\wedge B}) = k_2 = w(A)\cdot w(B)$$



Marginal inference via WMC

$$Pr(B|A) = \frac{WMC(A\wedge B, w)}{WMC(A, w)} = \frac{k_2}{k_1 + k_2}$$

WMC for safe AI

Our model: a (parametric) PLP P

θ_1 :: stress(X) $\leftarrow$ person(X).

θ_2 :: influences(X, Y) $\leftarrow$ person(X),
person(Y).

smokes(X) $\leftarrow$ stress(X).

smokes(X) $\leftarrow$ friend(X, Y),
influences(Y, X),
smokes(Y).

person(a). person(b). person(c).

friend(a, b).

...

evidence(friend(a, b)).

evidence(smokes(c)).

...

WMC for safe AI

Our model: a (parametric) PLP P

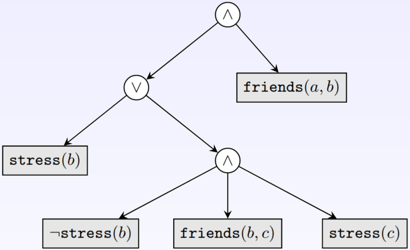
1) Conversion in weighted prop. logic

$$P \to P_{\text{ground}} \to \langle \Delta, w \rangle$$

$$\begin{aligned}
\Delta = \ & person(a) \land person(b) \\
& \ldots \\
& \land (aux_1 \land person(b) \to stress(b)) \lor \\
& \ldots
\end{aligned}$$

$$\begin{aligned}
& w(person(a)) = 1 \\
& w(person(b)) = 1 \\
& \ldots \\
& w(aux_1) = \theta_1 \\
& w(aux_2) = \theta_2
\end{aligned}$$

WMC for safe AI



Our model: a (parametric) PLP P

1) Conversion in weighted prop. logic

$$P \rightarrow P_{\text{ground}} \rightarrow \langle \Delta, w \rangle$$

2) Knowledge compilation:

$$\Delta \rightarrow C_\Delta$$

- $WMC(\Delta, w)$ computed in $\Theta(|C_\Delta|)$
- C_Δ is differentiable w.r.t. w

WMC for safe AI

Query ▾	Probability
funds(fwo,paolo)	0
influences(paolo,pollo)	0.2
smokes(paolo)	0.3

Our model: a (parametric) PLP P

1) Conversion in weighted prop. logic

$$P \rightarrow P_{\text{ground}} \rightarrow \langle \Delta, w \rangle$$

2) Knowledge compilation:

$$\Delta \rightarrow C_\Delta$$

- $WMC(\Delta, w)$ computed in $\Theta(|C_\Delta|)$
- C_Δ is differentiable w.r.t. w

Inference (and learning) in a model that

satisfy constraints **by construction**!

WMC for safe AI

WMC-based inference can be expensive (even with neural predicates)..

WMC for safe AI

WMC-based inference can be expensive (even with neural predicates)..

..what if we can't afford it at inference time?

WMC for safe AI

WMC-based inference can be expensive (even with neural predicates)..

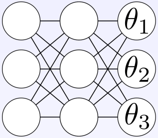
..what if we can't afford it at inference time?

Can we still leverage our background knowledge Δ?

WMC for safe AI

Our model: a differentiable function

$f(\mathbf{x}) = Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$ where $Pr(Y_i = \top|\mathbf{X}) = \theta_i$

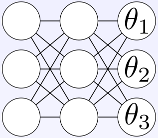


WMC for safe AI

Our model: a differentiable function

$$f(\mathbf{x}) = Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x}) \text{ where } Pr(Y_i = \top|\mathbf{X}) = \theta_i$$

Regularize f w.r.t. a constraint Δ over $\mathbf{Y}$:

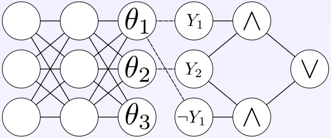


$\mathbf{Y} \models \Delta?$

$$\begin{aligned} \mathrm{SL}_{\Delta}(\Theta) &= -log WMC(\Delta, \Theta) \\ &= -log \sum_{\mu \models \Delta} \prod_{Y_i \in \mu} \theta_i \prod_{\neg Y_j \in \mu} (1 - \theta_j) \end{aligned}$$

...it's a **big** log-polynomial!

WMC for safe AI



but we can use KC!

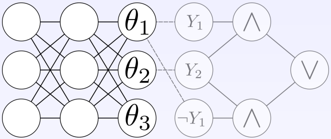
Our model: a differentiable function

$f(\mathbf{x}) = Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$ where $Pr(Y_i = \top|\mathbf{X}) = \theta_i$

Regularize f w.r.t. a constraint Δ over $\mathbf{Y}$:

$$\begin{aligned} \mathrm{SL}_\Delta(\Theta) &= -log WMC(\Delta, \Theta) \\ &= -log \sum_{\mu \models \Delta} \prod_{Y_i \in \mu} \theta_i \prod_{\neg Y_j \in \mu} (1 - \theta_j) \end{aligned}$$

WMC for safe AI



Our model: a differentiable function

$f(\mathbf{x}) = Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$ where $Pr(Y_i = \top|\mathbf{X}) = \theta_i$

Regularize f w.r.t. a constraint Δ over $\mathbf{Y}$:

$$\begin{aligned} \mathrm{SL}_\Delta(\Theta) &= -log WMC(\Delta, \Theta) \\ &= -log \sum_{\mu \models \Delta} \prod_{Y_i \in \mu} \theta_i \prod_{\neg Y_j \in \mu} (1 - \theta_j) \end{aligned}$$

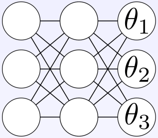
Once f is trained, we can discard C_Δ

WMC for safe AI

Our model: a differentiable function

$$f(\mathbf{x}) = Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x}) \text{ where } Pr(Y_i = \top|\mathbf{X}) = \theta_i$$

Regularize f w.r.t. a constraint Δ over $\mathbf{Y}$:



$\mathbf{Y} \models \Delta!$

$$\begin{aligned}\mathrm{SL}_{\Delta}(\Theta) &= -log WMC(\Delta, \Theta) \\ &= -log \sum_{\mu \models \Delta} \prod_{Y_i \in \mu} \theta_i \prod_{\neg Y_j \in \mu} (1 - \theta_j)\end{aligned}$$

Once f is trained, we can discard C_{Δ}

Efficient inference that

satisfy constraints **in expectation**!

What does *"in expectation"* mean?

What does *"in expectation"* mean?

Point-wise evaluations can be misleading..

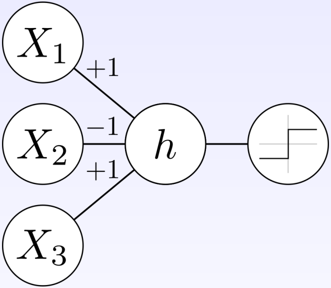
WMC for safe AI

What does *"in expectation"* mean?

Point-wise evaluations can be misleading..

..how can we be sure that Δ is satisfied enough in the real world?

WMC for safe AI

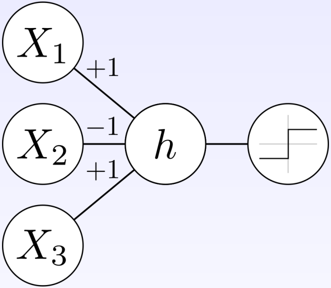


$$Y = sign\left(\langle w, \mathbf{X}\rangle + b\right)$$

Our model: a binarized NN

- $\mathcal{N} : \mathbb{Z}^N \rightarrow \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

WMC for safe AI



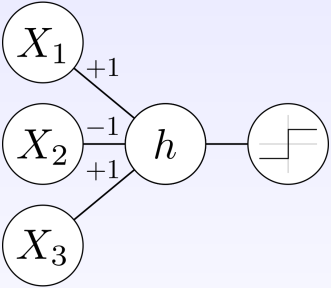
$Y = sign\left(\langle w, \mathbf{X}\rangle + b\right)$

Our model: a binarized NN

- $\mathcal{N} : \mathbb{Z}^N \rightarrow \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

1) Conversion $\mathcal{N} \rightarrow \Delta$

WMC for safe AI



$$Y = sign\left(\langle w, \mathbf{X}\rangle + b\right)$$

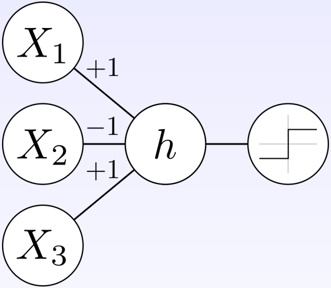
Our model: a binarized NN

- $\mathcal{N} : \mathbb{Z}^N \rightarrow \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

1) Conversion $\mathcal{N} \rightarrow \Delta$

- *trivial* encoding Δ is exponentially large

WMC for safe AI



$$Y = sign\left(\langle w, \mathbf{X} \rangle + b\right)$$

Our model: a binarized NN

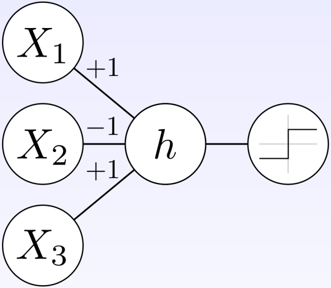
- $\mathcal{N} : \mathbb{Z}^N \rightarrow \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

1) Conversion $\mathcal{N} \rightarrow \Delta$

- *trivial* encoding Δ is exponentially large
- *equicardinal* Δ' is found using MILP

$$\#SAT(\Delta) = \#SAT(\Delta')$$

WMC for safe AI



$$Y = sign\left(\langle w, \mathbf{X} \rangle + b\right)$$

Our model: a binarized NN

- $\mathcal{N} : \mathbb{Z}^N \rightarrow \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

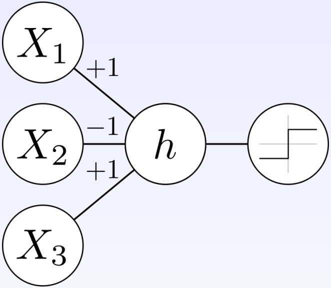
1) Conversion $\mathcal{N} \rightarrow \Delta$

- *trivial* encoding Δ is exponentially large
- *equicardinal* Δ' is found using MILP

$$\#SAT(\Delta) = \#SAT(\Delta')$$

2) Prior $P(\mathbf{X})$? Reduction WMC $\rightarrow$ #SAT

WMC for safe AI



$$Y = sign\left(\langle w, \mathbf{X}\rangle + b\right)$$

Our model: a binarized NN

- $\mathcal{N} : \mathbb{Z}^N \to \mathbb{B}^M$
- weights $w \in \{-1, 1\}$ and step activations

1) Conversion $\mathcal{N} \to \Delta$

- *trivial* encoding Δ is exponentially large
- *equicardinal* Δ' is found using MILP

$$\#SAT(\Delta) = \#SAT(\Delta')$$

2) Prior $P(\mathbf{X})$? Reduction WMC $\to$ #SAT

Approximate $\langle \epsilon, \delta \rangle$-counting for

verifying quantitative properties!

WMC for safe AI

In discrete settings, WMC can be used for:

learning models that satisfy constraints **by construction**

learning models that satisfy constraints **in expectation**

verifying quantitative properties of learned models

WMC for safe AI

In discrete settings, WMC can be used for:

learning models that satisfy constraints **by construction**

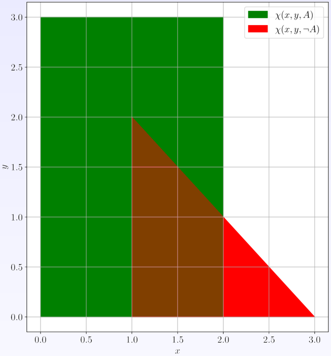
learning models that satisfy constraints **in expectation**

verifying quantitative properties of learned models

..but what about continuous or hybrid models?

Generalizing WMC

SMT formulas
w/ algebraic constraints

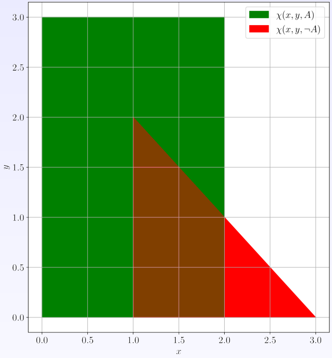


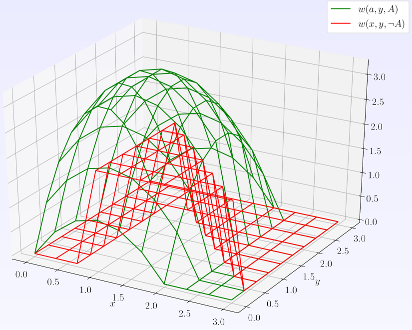
$$\chi = (0 \leq y) \wedge (y \leq 3) \wedge (0 \leq x)$$
$$\wedge\, (A \rightarrow (x \leq 2))$$
$$\wedge\, (\neg A \rightarrow ((1 \leq x) \wedge (x + y \leq 3)))$$

Generalizing WMC

SMT formulas
w/ algebraic constraints

+ density functions
over continuous variables





$$\chi = (0 \leq y) \wedge (y \leq 3) \wedge (0 \leq x)$$
$$\wedge (A \rightarrow (x \leq 2))$$
$$\wedge (\neg A \rightarrow ((1 \leq x) \wedge (x + y \leq 3)))$$

$$w(x, y, A) = [\![A]\!](-x^2 - y^2 + 2x + 3y)$$
$$+ [\![\neg A]\!](-2x - 2y + 6)$$

Weighted Model Integration

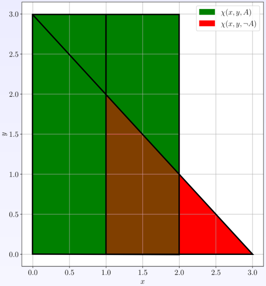
$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu)$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$

Weighted Model Integration

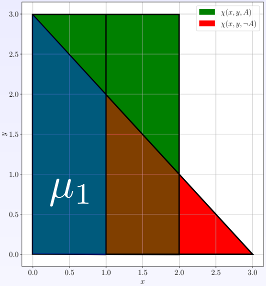
$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_\mu f(\mathbf{x}) d\mathbf{x}$$



$$WMI(\chi, w) =$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$



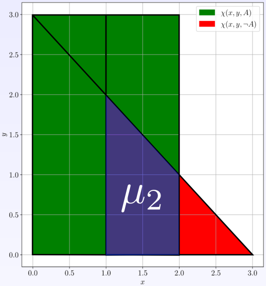
$$\mu_1 = \begin{cases} A \\ \neg(1 \leq x) \\ (x \leq 2) \\ (x + y \leq 3) \\ \dots \end{cases}$$

$$w(\mu_1) = \int_0^1 \int_0^{3-x} -x^2 - y^2 + 2x + 3y \, dy \, dx$$

$$WMI(\chi, w) = w(\mu_1) +$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$



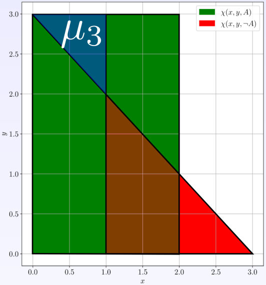
$$\mu_2 = \begin{cases} A \\ (1 \leq x) \\ (x \leq 2) \\ (x + y \leq 3) \\ \dots \end{cases}$$

$$w(\mu_2) = \int_1^2 \int_0^{3-x} -x^2 - y^2 + 2x + 3y \, dy \, dx$$

$$WMI(\chi, w) = w(\mu_1) + w(\mu_2) +$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$



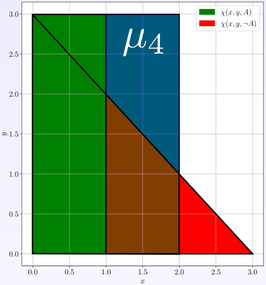
$$\mu_3 = \begin{cases} A \\ \neg(1 \leq x) \\ (x \leq 2) \\ \neg(x + y \leq 3) \\ \dots \end{cases}$$

$$w(\mu_3) = \int_0^1 \int_{3-x}^3 -x^2 - y^2 + 2x + 3y \; dy \; dx$$

$$WMI(\chi, w) = w(\mu_1) + w(\mu_2) + w(\mu_3) +$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$



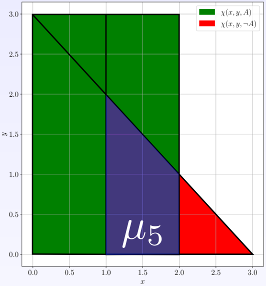
$$\mu_4 = \begin{cases} A \\ (1 \leq x) \\ (x \leq 2) \\ \neg(x + y \leq 3) \\ \dots \end{cases}$$

$$w(\mu_4) = \int_1^2 \int_{3-x}^3 -x^2 - y^2 + 2x + 3y \; dy \; dx$$

$$WMI(\chi, w) = w(\mu_1) + w(\mu_2) + w(\mu_3) + w(\mu_4) +$$

Weighted Model Integration

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$

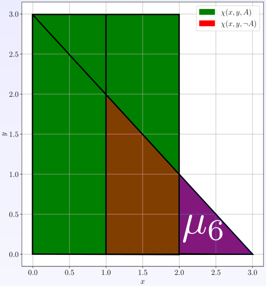


$$\mu_5 = \begin{cases} \neg A \\ (1 \leq x) \\ (x \leq 2) \\ \neg(x + y \leq 3) \\ \ldots \end{cases}$$

$$w(\mu_5) = \int_1^2 \int_0^{3-x} -2x - 2y + 6 \, dy \, dx$$

$$WMI(\chi, w) = w(\mu_1) + w(\mu_2) + w(\mu_3) + w(\mu_4) + w(\mu_5) +$$

$$WMI(\Delta, w) = \sum_{\mu \models \Delta} w(\mu) = \sum_{\mu \models \Delta} \int_{\mu} f(\mathbf{x}) d\mathbf{x}$$



$$\mu_6 = \begin{cases} \neg A \\ (1 \leq x) \\ \neg(x \leq 2) \\ \neg(x + y \leq 3) \\ \dots \end{cases}$$

$$w(\mu_6) = \int_2^3 \int_0^{3-x} -2x - 2y + 6 \, dy \, dx$$

$$WMI(\chi, w) = w(\mu_1) + w(\mu_2) + w(\mu_3) + w(\mu_4) + w(\mu_5) + w(\mu_6)$$

Weighted Model Integration

Two subtasks:

Weighted Model Integration

Two subtasks:

1) **Enumerating** convex integration regions μ (λ-SMT)
$\rightarrow$ as hard as #SAT (#P-complete)

Weighted Model Integration

Two subtasks:

1) **Enumerating** convex integration regions μ (λ-SMT)
$\rightarrow$ as hard as #SAT (#P-complete)

2) **Continuous integration**
$\rightarrow$ as hard as computing the volume of a polytope (#P-hard)

Weighted Model Integration

Two subtasks:

1) **Enumerating** convex integration regions μ (λ-SMT)
$\rightarrow$ as hard as #SAT (#P-complete)

2) **Continuous integration**
$\rightarrow$ as hard as computing the volume of a polytope (#P-hard)

The **combination** of 1) and 2) makes it very tricky
$\rightarrow$ successful ideas in WMC do not always apply

Weighted Model Integration

Two subtasks:

1) **Enumerating** convex integration regions μ (λ-SMT)
$\rightarrow$ as hard as #SAT (#P-complete)

2) **Continuous integration**
$\rightarrow$ as hard as computing the volume of a polytope (#P-hard)

The **combination** of 1) and 2) makes it very tricky
$\rightarrow$ successful ideas in WMC do not always apply

Why even bother?

Weighted Model Integration

Two subtasks:

1) **Enumerating** convex integration regions μ (λ-SMT)
$\rightarrow$ as hard as #SAT (#P-complete)

2) **Continuous integration**
$\rightarrow$ as hard as computing the volume of a polytope (#P-hard)

The **combination** of 1) and 2) makes it very tricky
$\rightarrow$ successful ideas in WMC do not always apply

Why even bother?

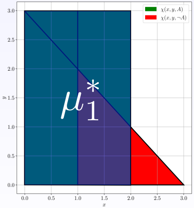
- This is a **very general** framework!
- Many inference algorithms are **specific** polytime WMI "in disguise"

Computing WMI efficiently

$$
\begin{aligned}
WMI(\chi, w) = &\int_0^1 \int_0^{3-x} f_A(x,y)\,dy\,dx + \int_1^2 \int_0^{3-x} f_A(x,y)\,dy\,dx \\
&+ \int_0^1 \int_{3-x}^{3} f_A(x,y)\,dy\,dx + \int_1^2 \int_{3-x}^{3} f_A(x,y)\,dy\,dx \\
&+ \int_1^2 \int_0^{3-x} f_{\neg A}(x,y)\,dy\,dx + \int_2^3 \int_0^{3-x} f_{\neg A}(x,y)\,dy\,dx
\end{aligned}
$$

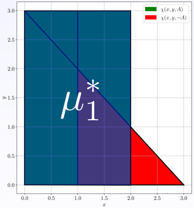
Computing WMI efficiently

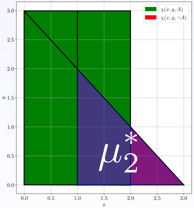
$$
\begin{aligned}
WMI(\chi, w) = & \int_0^1 \int_0^{3-x} f_A(x,y)\, dy\, dx + \int_1^2 \int_0^{3-x} f_A(x,y)\, dy\, dx \\
& + \int_0^1 \int_{3-x}^{3} f_A(x,y)\, dy\, dx + \int_1^2 \int_{3-x}^{3} f_A(x,y)\, dy\, dx \\
& + \int_1^2 \int_0^{3-x} f_{\neg A}(x,y)\, dy\, dx + \int_2^3 \int_0^{3-x} f_{\neg A}(x,y)\, dy\, dx \\
= & \overbrace{\int_0^2 \int_0^3 f_A(x,y)\, dy\, dx}^{w(\mu_1^*)}
\end{aligned}
$$



Computing WMI efficiently

$$
\begin{aligned}
WMI(\chi, w) &= \int_0^1 \int_0^{3-x} f_A(x,y)\,dy\,dx + \int_1^2 \int_0^{3-x} f_A(x,y)\,dy\,dx \\
&+ \int_0^1 \int_{3-x}^{3} f_A(x,y)\,dy\,dx + \int_1^2 \int_{3-x}^{3} f_A(x,y)\,dy\,dx \\
&+ \int_1^2 \int_0^{3-x} f_{\neg A}(x,y)\,dy\,dx + \int_2^3 \int_0^{3-x} f_{\neg A}(x,y)\,dy\,dx \\
&= \overbrace{\int_0^2 \int_0^3 f_A(x,y)\,dy\,dx}^{w(\mu_1^*)} + \overbrace{\int_1^3 \int_0^{3-x} f_{\neg A}(x,y)\,dy\,dx}^{w(\mu_2^*)}
\end{aligned}
$$





Computing WMI efficiently

How?

- Knowledge compilation
- SMT oracles
- Tractable subclasses
- Monte Carlo estimates

Computing WMI efficiently

How?

- Knowledge compilation
- SMT oracles
- Tractable subclasses
- Monte Carlo estimates

Algorithm	Combinatorial Enumeration		Integration			Expressiveness	
	Exact	Method	Exact	Sym.	Method	Parametric form	Assumptions
WMI-CC	✓	DPLL-SMT	✓		LattE + CC	Mul. polynomial	UC
WMI-PA	✓	DPLL-SMT	✓		LattE	Mul. polynomial	-
PRAiSE	✓	DPLL-PIMT	✓	✓	PIMT	Mul. polynomial	-
SVE	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
BR	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
F-XSDD	✓	KC-XSDD	✓	✓	XADD / PSI	Mul. polynomial	-
WMI-SDD	✓	KC-XSDD	✓		Scipy / LattE	Mul. polynomial	-
Symbo	✓	KC-XSDD	✓	✓	PSI	Mul. Gaussian	UC
Sampo	✓	KC-XSDD			MC	Mul. Gaussian	-
SMI	✓	AND/OR search	✓	✓	univ. integration	Biv. monomials	BC, CNF, A
MP-WMI	✓	MP	✓	✓	Sympy	Biv. polynomial	BC, CNF, A
ReCoIn		MP	✓	✓	Sympy	Biv. polynomial	BC, CNF
AprxWMI-CNF		hashing + SMT	✓		LattE	Mul. polynomial	CNF
AprxWMI-DNF		FPRAS	✓		LattE	Mul. polynomial	DNF

Computing WMI efficiently

How?

- Knowledge compilation
- SMT oracles
- Tractable subclasses
- Monte Carlo estimates

Algorithm	Combinatorial Enumeration		Integration			Expressiveness	
	Exact	Method	Exact	Sym.	Method	Parametric form	Assumptions
WMI-CC	✓	DPLL-SMT	✓		LattE + CC	Mul. polynomial	UC
WMI-PA	✓	DPLL-SMT	✓		LattE	Mul. polynomial	-
PRAiSE	✓	DPLL-PIMT	✓	✓	PIMT	Mul. polynomial	-
SVE	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
BR	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
F-XSDD	✓	KC-XSDD	✓	✓	XADD / PSI	Mul. polynomial	-
WMI-SDD	✓	KC-XSDD	✓		Scipy / LattE	Mul. polynomial	-
Symbo	✓	KC-XSDD	✓	✓	PSI	Mul. Gaussian	UC
Sampo	✓	KC-XSDD			MC	Mul. Gaussian	-
SMI	✓	AND/OR search	✓	✓	univ. integration	Biv. monomials	BC, CNF, A
MP-WMI	✓	MP	✓	✓	Sympy	Biv. polynomial	BC, CNF, A
ReCoIn		MP	✓	✓	Sympy	Biv. polynomial	BC, CNF
AprxWMI-CNF		hashing + SMT	✓		LattE	Mul. polynomial	CNF
AprxWMI-DNF		FPRAS	✓		LattE	Mul. polynomial	DNF

Research focused on theory and algorithms

Computing WMI efficiently

How?

Knowledge compilation
SMT oracles
Tractable subclasses
Monte Carlo estimates

$\Bigg\}$ **guided by applications in safe AI**

Algorithm	Combinatorial Enumeration		Integration			Expressiveness	
	Exact	Method	Exact	Sym.	Method	Parametric form	Assumptions
WMI-CC	✓	DPLL-SMT	✓		LattE + CC	Mul. polynomial	UC
WMI-PA	✓	DPLL-SMT	✓		LattE	Mul. polynomial	-
PRAiSE	✓	DPLL-PIMT	✓	✓	PIMT	Mul. polynomial	-
SVE	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
BR	✓	KC-XADD	✓	✓	XADD	Mul. polynomial	-
F-XSDD	✓	KC-XSDD	✓	✓	XADD / PSI	Mul. polynomial	-
WMI-SDD	✓	KC-XSDD	✓		Scipy / LattE	Mul. polynomial	-
Symbo	✓	KC-XSDD	✓	✓	PSI	Mul. Gaussian	UC
Sampo	✓	KC-XSDD			MC	Mul. Gaussian	-
SMI	✓	AND/OR search	✓	✓	univ. integration	Biv. monomials	BC, CNF, A
MP-WMI	✓	MP	✓	✓	Sympy	Biv. polynomial	BC, CNF, A
ReCoIn		MP	✓	✓	Sympy	Biv. polynomial	BC, CNF
AprxWMI-CNF		hashing + SMT	✓		LattE	Mul. polynomial	CNF
AprxWMI-DNF		FPRAS	✓		LattE	Mul. polynomial	DNF

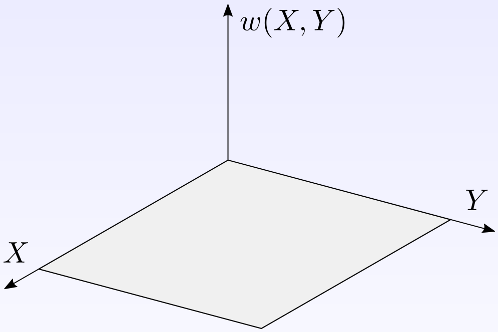
Research focused on theory and algorithms

WMI for safe AI

learning models that satisfy constraints **by construction**?

WMI for safe AI

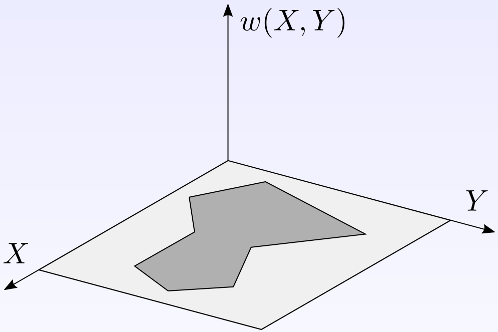
learning models that satisfy constraints **by construction**?



Structure

WMI for safe AI

learning models that satisfy constraints **by construction**?

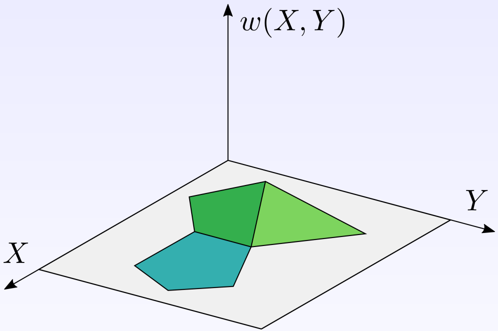


Structure

$\rightarrow$ support/constraints Δ (*if not/partially given*)

WMI for safe AI

learning models that satisfy constraints **by construction**?



Structure

$\rightarrow$ support/constraints Δ (*if not/partially given*)

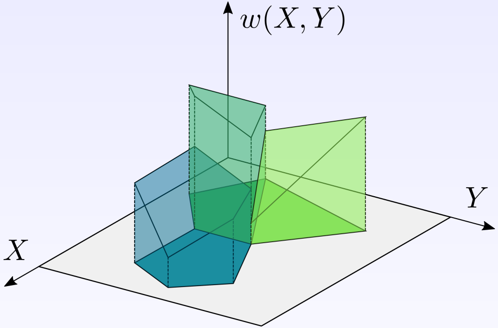
$\rightarrow$ $w(\mathbf{X};\Theta) =$ [if ψ then $w_1(\mathbf{X};\Theta_1)$ else $w_2(\mathbf{X};\Theta_2)$]

$[w_1(\mathbf{X_1};\Theta_1) \times w_2(\mathbf{X_2};\Theta_2)]$

$[\theta_1 \cdot w_1(\mathbf{X};\Theta_1) + \theta_2 \cdot w_2(\mathbf{X};\Theta_2)]$

WMI for safe AI

learning models that satisfy constraints **by construction**?



Structure

$\rightarrow$ support/constraints Δ (*if not/partially given*)

$\rightarrow$ $w(\mathbf{X};\Theta) = [\text{if } \psi \text{ then } w_1(\mathbf{X};\Theta_1) \text{ else } w_2(\mathbf{X};\Theta_2)]$
$[w_1(\mathbf{X_1};\Theta_1) \times w_2(\mathbf{X_2};\Theta_2)]$
$[\theta_1 \cdot w_1(\mathbf{X};\Theta_1) + \theta_2 \cdot w_2(\mathbf{X};\Theta_2)]$

Parameters

$\rightarrow$ *argmin*$_\Theta$ $\mathcal{L}(\Theta)$

WMI for safe AI

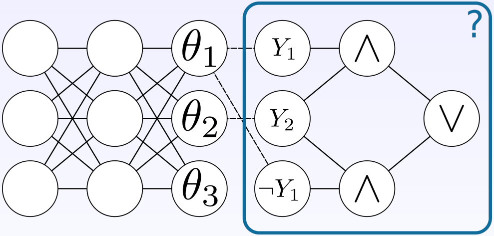
learning models that satisfy constraints **in expectation**?

WMI for safe AI

learning models that satisfy constraints **in expectation**?

Efficient WMI evaluation

Tractable classes?
Knowledge compilation?
Sampling?



WMI for safe AI

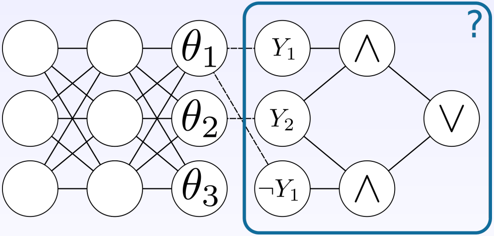
learning models that satisfy constraints **in expectation**?

Efficient WMI evaluation

Tractable classes?
Knowledge compilation?
Sampling?

Constraints

Logical + linear?
Nonlinear? ($E = mc^2$)



WMI for safe AI

learning models that satisfy constraints **in expectation**?

Efficient WMI evaluation

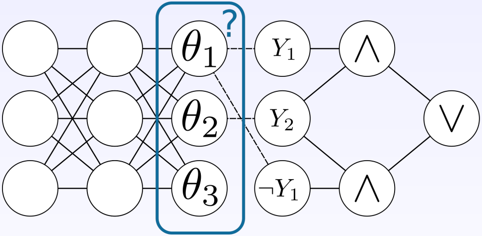
Tractable classes?
Knowledge compilation?
Sampling?

Constraints

Logical + linear?
Nonlinear? ($E = mc^2$)

What models to target

Piecewise polynomials?
Exponential family?

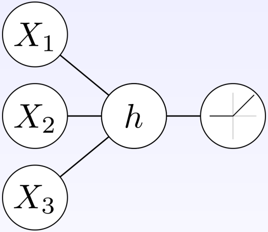


WMI for safe AI

verifying quantitative properties?

verifying quantitative properties?

$Y = max(0, \langle w, \mathbf{X} \rangle + b)$



$$\Delta = (h = \langle w, \mathbf{X} \rangle + b)$$
$$\wedge ((h \leq 0) \rightarrow (Y = 0))$$
$$\wedge ((h > 0) \rightarrow (Y = h))$$

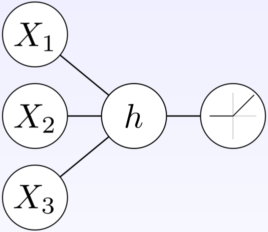
We can encode:

Decision trees
Support vector machines
ReLU networks
Sum-product networks
...

WMI for safe AI

verifying quantitative properties?

$Y = max(0, \langle w, \mathbf{X} \rangle + b)$



$$\begin{aligned}\Delta = & (h = \langle w, \mathbf{X} \rangle + b) \\ & \wedge ((h \leq 0) \rightarrow (Y = 0)) \\ & \wedge ((h > 0) \rightarrow (Y = h))\end{aligned}$$

We can encode:

Decision trees
Support vector machines
ReLU networks
Sum-product networks
...

We can verify properties like:

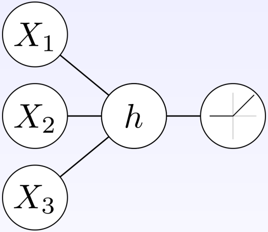
$$Pr(Hire|Female) = Pr(Hire|Male)$$

(with arbitrary priors over $\mathbf{X}$!)

WMI for safe AI

verifying quantitative properties?

$Y = max(0, \langle w, \mathbf{X} \rangle + b)$



$$\Delta = (h = \langle w, \mathbf{X} \rangle + b)$$
$$\wedge ((h \leq 0) \rightarrow (Y = 0))$$
$$\wedge ((h > 0) \rightarrow (Y = h))$$

We can encode:

Decision trees
Support vector machines
ReLU networks
Sum-product networks
...

We can verify properties like:

$$Pr(Hire|Female) = Pr(Hire|Male)$$

(with arbitrary priors over **X**!)

Can we scale?

Possibly, focusing on specific models/properties!

The final slide

The final slide

- WMI is a **very general framework** for constrained inference

The final slide

- WMI is a **very general framework** for constrained inference

...and it is not THAT scary!

The final slide

- WMI is a **very general framework** for constrained inference

...and it is not THAT scary!

- There are **many potential applications** of WMI in safe AI

The final slide

- WMI is a **very general framework** for constrained inference

...and it is not THAT scary!

- There are **many potential applications** of WMI in safe AI

...also many algorithmic challenges (I haven't talked about)

The final slide

- WMI is a **very general framework** for constrained inference

...and it is not THAT scary!

- There are **many potential applications** of WMI in safe AI

...also many algorithmic challenges (I haven't talked about)

Thank you!

questions / feedback / collaborations are welcome!