On Finding Minimum Satisfying Assignments

Alexey Ignatiev[1,2], Alessandro Previti[1], and Joao Marques-Silva[1]

September 9, 2016

[1] LaSIGE, FC, University of Lisbon, Portugal
[2] ISDCT SB RAS, Irkutsk, Russia

Problem definition

Minimum satisfying assignment

what is a *minimum satisfying assignment* (MSA)?

Minimum satisfying assignment

what is a *minimum satisfying assignment* (MSA)?

$$x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$$w, x, y, z \in \mathbb{Z}$$

Minimum satisfying assignment

what is a *minimum satisfying assignment* (MSA)?

$$x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$$w, x, y, z \in \mathbb{Z}$$

$w = 5 \quad x = -1 \quad y = -1 \quad z = -1$ — *SA (satisfying assignment)*

Minimum satisfying assignment

what is a ***minimum satisfying assignment*** (MSA)?

$$x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$$w, x, y, z \in \mathbb{Z}$$

$w = 5 \quad x = -1 \quad y = -1 \quad z = -1$ — *SA (satisfying assignment)*

$w = 2 \quad x = 1 \quad y = 0 \quad z = 0$ — *SA*

Minimum satisfying assignment

what is a ***m**inimum **s**atisfying **a**ssignment* (MSA)?

$$x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$$w, x, y, z \in \mathbb{Z}$$

	$w = 5$	$x = -1$	$y = -1$	$z = -1$	— *SA (satisfying assignment)*
	$w = 2$	$x = 1$	$y = 0$	$z = 0$	— *SA*
⋆	~~$w = 2$~~	$x = 1$	$y = 0$	$z = 0$	— *minimal SA (mSA)*

Minimum satisfying assignment

what is a ***m**inimum **s**atisfying **a**ssignment* (MSA)?

$$x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$$w, x, y, z \in \mathbb{Z}$$

	$w = 5$	$x = -1$	$y = -1$	$z = -1$	— *SA (satisfying assignment)*
	$w = 2$	$x = 1$	$y = 0$	$z = 0$	— *SA*
⋆	~~$w = 2$~~	$x = 1$	$y = 0$	$z = 0$	— *minimal SA (mSA)*
⋆	$w = 2$	~~$x = 1$~~	~~$y = 0$~~	~~$z = 0$~~	— *minimum SA (MSA)*

$$(x + y + z > 0 \quad \vee \quad x + y + z < 3)$$

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$\{a, b, c\}$ $\{d, e, f\}$ $\{a, f, g\}$ $\{b, g, h\}$ $\{d, f, g\}$

Hitting sets

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$$\{a, b, c\} \quad \{d, e, f\} \quad \{a, f, g\} \quad \{b, g, h\} \quad \{d, f, g\}$$

$\{b, e, f, g\}$ — *HS (hitting set)*

Hitting sets

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$$\{a, b, c\} \quad \{d, e, f\} \quad \{a, f, g\} \quad \{b, g, h\} \quad \{d, f, g\}$$

$\{b, e, f, g\}$ — *HS (hitting set)*

$\{a, d, g, h\}$ — *HS*

Hitting sets

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$$\{a, b, c\} \quad \{d, e, f\} \quad \{a, f, g\} \quad \{b, g, h\} \quad \{d, f, g\}$$

$\{b, e, f, g\}$ — *HS (hitting set)*

$\{a, d, g, h\}$ — *HS*

⋆ $\{a, d, g, \not{h}\}$ — *minimal HS (mHS)*

Hitting sets

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$$\{a, b, c\} \quad \{d, e, f\} \quad \{a, f, g\} \quad \{b, g, h\} \quad \{d, f, g\}$$

$\{b, e, f, g\}$ — *HS (hitting set)*

$\{a, d, g, h\}$ — *HS*

⋆ $\{a, d, g, \not{h}\}$ — *minimal HS (mHS)*

⋆ $\{a, d, \not{g}, h\}$ — *minimal HS (mHS)*

Hitting sets

given $\mathbb{U} = \{a, b, c, d, e, f, g, h\}$ and its subsets

$\{a, b, c\}$ $\{d, e, f\}$ $\{a, f, g\}$ $\{b, g, h\}$ $\{d, f, g\}$

	$\{b, e, f, g\}$	— *HS (hitting set)*
	$\{a, d, g, h\}$	— *HS*
⋆	$\{a, d, g, \cancel{h}\}$	— *minimal HS (mHS)*
⋆	$\{a, d, \cancel{g}, h\}$	— *minimal HS (mHS)*
⋆	$\{b, \cancel{e}, f, \cancel{g}\}$	— *minimum HS (MHS)*

Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

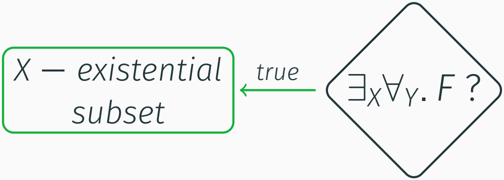
Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

$$\exists_X \forall_Y .\, F\ ?$$

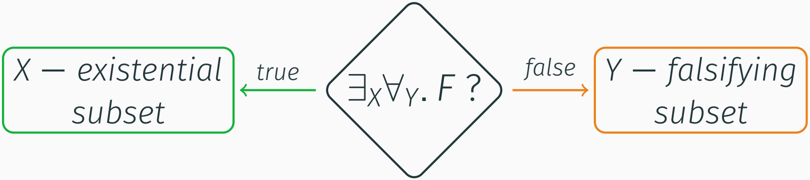
Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,



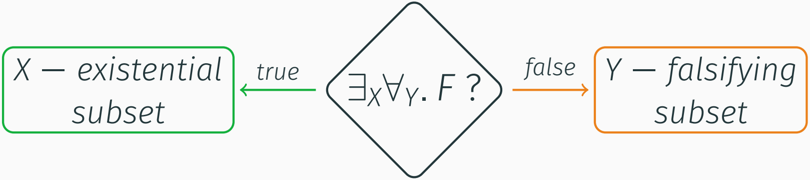
Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,



Existential and falsifying subsets

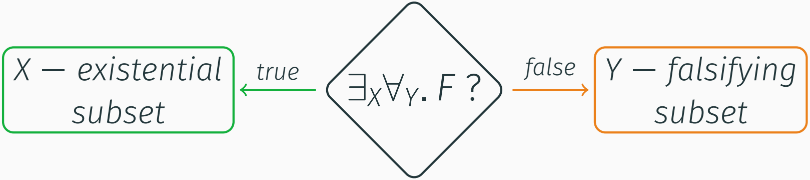
given F s.t. $\mathrm{var}(F) = X \cup Y$,



$$F = x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

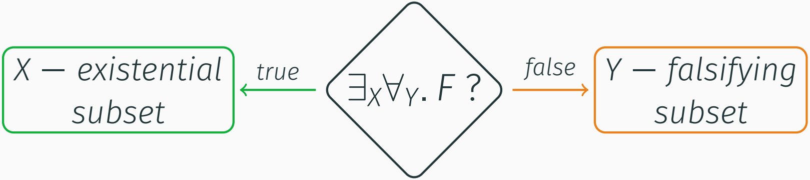


$$F = x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

$\{x\} \cup \{w, y, z\}$ $\quad \exists_x \forall_{w,y,z} .\, F = false$ $\quad \{w, y, z\}$ $\quad$ — *falsifying subset (FS)*

Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

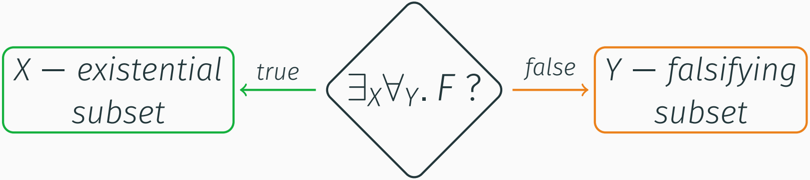


$$F = x + y + z > 0 \;\vee\; w + x + y + z < 5$$

	$\{x\} \cup \{w, y, z\}$	$\exists_x \forall_{w,y,z}. F = \mathit{false}$	$\{w, y, z\}$	— *falsifying subset (FS)*
⋆	$\{x, y\} \cup \{w, z\}$	$\exists_{x,y} \forall_{w,z}. F = \mathit{false}$	$\{w, z\}$	— *minimal FS (mFS)*

Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

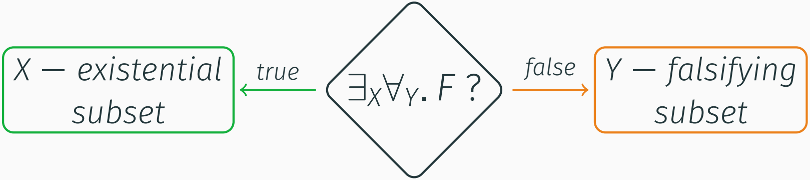


$$F = x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

	$\{x\} \cup \{w, y, z\}$	$\exists_x \forall_{w,y,z}.\, F = \mathit{false}$	$\{w, y, z\}$	— *falsifying subset (FS)*
⋆	$\{x, y\} \cup \{w, z\}$	$\exists_{x,y} \forall_{w,z}.\, F = \mathit{false}$	$\{w, z\}$	— *minimal FS (mFS)*
	$\{w, x, y, z\} \cup \emptyset$	$\exists_{w,x,y,z}.\, F = \mathit{true}$	$\{w, x, y, z\}$	— *existential set (ES)*

Existential and falsifying subsets

given F s.t. $\mathrm{var}(F) = X \cup Y$,

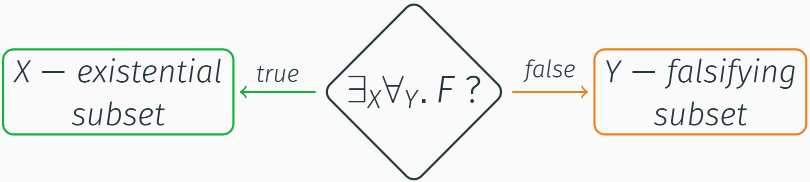


$$F = x + y + z > 0 \;\vee\; w + x + y + z < 5$$

	$\{x\} \cup \{w, y, z\}$	$\exists_x \forall_{w,y,z}.\, F = \textit{false}$	$\{w, y, z\}$	— *falsifying subset (FS)*
⋆	$\{x, y\} \cup \{w, z\}$	$\exists_{x,y} \forall_{w,z}.\, F = \textit{false}$	$\{w, z\}$	— *minimal FS (mFS)*
	$\{w, x, y, z\} \cup \emptyset$	$\exists_{w,x,y,z}.\, F = \textit{true}$	$\{w, x, y, z\}$	— *existential set (ES)*
⋆	$\{x, y, z\} \cup \{w\}$	$\exists_{x,y,z} \forall_{w}.\, F = \textit{true}$	$\{x, y, z\}$	— *minimal ES (mES)*

Existential and falsifying subsets

given F s.t. $\text{var}(F) = X \cup Y$,



$$F = x + y + z > 0 \quad \vee \quad w + x + y + z < 5$$

	$\{x\} \cup \{w, y, z\}$	$\exists_x \forall_{w,y,z}.\ F = \mathit{false}$	$\{w, y, z\}$	— *falsifying subset (FS)*
⋆	$\{x, y\} \cup \{w, z\}$	$\exists_{x,y} \forall_{w,z}.\ F = \mathit{false}$	$\{w, z\}$	— *minimal FS (mFS)*
	$\{w, x, y, z\} \cup \emptyset$	$\exists_{w,x,y,z}.\ F = \mathit{true}$	$\{w, x, y, z\}$	— *existential set (ES)*
⋆	$\{x, y, z\} \cup \{w\}$	$\exists_{x,y,z} \forall_w.\ F = \mathit{true}$	$\{x, y, z\}$	— *minimal ES (mES)*
⋆	$\{w\} \cup \{x, y, z\}$	$\exists_w \forall_{x,y,z}.\ F = \mathit{true}$	$\{w\}$	— *minimum ES (MES)*

Approach

Minimal hitting set duality

given F,

Minimal hitting set duality

given F,

$\mathcal{E}$ — set of **all mESes** for F

given F,

$\mathcal{E}$ — set of **all mESes** for F

$\mathcal{F}$ — set of **all mFSes** for F

Minimal hitting set duality

given F,

$\mathcal{E}$ — set of **all mESes** for F

$\mathcal{F}$ — set of **all mFSes** for F

1. set $e \in \mathcal{E}$

Minimal hitting set duality

given F,

$\mathcal{E}$ — set of **all mESes** for F

$\mathcal{F}$ — set of **all mFSes** for F

1. set $e \in \mathcal{E} \;\Leftrightarrow\; e$ is an *mHS of* $\mathcal{F}$

Minimal hitting set duality

given F,

$\mathcal{E}$ — set of **all mESes** for F

$\mathcal{F}$ — set of **all mFSes** for F

1. set $e \in \mathcal{E}$ $\Leftrightarrow$ e is an *mHS of* $\mathcal{F}$
2. set $f \in \mathcal{F}$

Minimal hitting set duality

given F,

$\mathcal{E}$ — set of **all mESes** for F

$\mathcal{F}$ — set of **all mFSes** for F

1. set $e \in \mathcal{E}$ $\Leftrightarrow$ e is an *mHS of* $\mathcal{F}$
2. set $f \in \mathcal{F}$ $\Leftrightarrow$ f is an *mHS of* $\mathcal{E}$

Algorithm

input : formula F
output: an MSA of F

```
𝓗 ← ∅
while true:
    X ← MinHS(𝓗)                      # get a new MHS with MaxSAT
    Y ← var(F) \ X                     # take complement of X
    (st, μ_X) ← Solve(∃_X ∀_Y. F)      # check if X is a minimum ES
    if st:
        break
    else:
        I ← Reduce(Y)                  # reduce counterexample
        𝓗 ← 𝓗 ∪ I                      # hit counterexample next time
return MSA ← μ_X
```

Algorithm

input : formula F
output: an MSA of F

$\mathcal{H} \leftarrow \emptyset$
while true:
    $X \leftarrow$ `MinHS`$(\mathcal{H})$ # get a new MHS with MaxSAT
    $Y \leftarrow \mathrm{var}(F) \setminus X$ # take complement of X
    $(\mathrm{st}, \mu_X) \leftarrow$ `Solve`$(\exists_X \forall_Y . F)$ # check if X is a minimum ES
    if st:
        break
    else:
        $I \leftarrow$ `Reduce`(Y) # reduce counterexample
        $\mathcal{H} \leftarrow \mathcal{H} \cup I$ # hit counterexample next time
return MSA $\leftarrow \mu_X$

Algorithm

input : formula F

output: an MSA of F

$\mathcal{H} \leftarrow \emptyset$
while true:
    $X \leftarrow \texttt{MinHS}(\mathcal{H})$ # get a new MHS with MaxSAT
    $Y \leftarrow \mathrm{var}(F) \setminus X$ # take complement of X
    $(\mathrm{st}, \mu_X) \leftarrow \texttt{Solve}(\exists_X \forall_Y . F)$ # check if X is a minimum ES
    if st:
        break
    else:
        $I \leftarrow \texttt{Reduce}(Y)$ # reduce counterexample
        $\mathcal{H} \leftarrow \mathcal{H} \cup I$ # hit counterexample next time
return $\mathrm{MSA} \leftarrow \mu_X$

input : formula F
output: an MSA of F

$\mathcal{H} \leftarrow \emptyset$
while true:
    $X \leftarrow \texttt{MinHS}(\mathcal{H})$ `# get a new MHS with MaxSAT`
    $Y \leftarrow \text{var}(F) \setminus X$ `# take complement of X`
    $(\text{st}, \mu_X) \leftarrow \texttt{Solve}(\exists_X \forall_Y . F)$ `# check if X is a minimum ES`
    if st:
        break
    else:
        $I \leftarrow \texttt{Reduce}(Y)$ `# reduce counterexample`
        $\mathcal{H} \leftarrow \mathcal{H} \cup I$ `# hit counterexample next time`
return $\text{MSA} \leftarrow \mu_X$

Algorithm

input : formula F
output: an MSA of F

```
ℋ ← ∅
while true:
    X ← MinHS(ℋ)                          # get a new MHS with MaxSAT
    Y ← var(F) \ X                        # take complement of X
    (st, μ_X) ← Solve(∃_X ∀_Y . F)        # check if X is a minimum ES
    if st:
        break
    else:
        I ← Reduce(Y)                     # reduce counterexample
        ℋ ← ℋ ∪ I                         # hit counterexample next time
return MSA ← μ_X
```

Algorithm

input : formula F
output: an MSA of F

```
ℋ ← ∅
while true:
    X ← MinHS(ℋ)                          # get a new MHS with MaxSAT
    Y ← var(F) \ X                        # take complement of X
    (st, μ_X) ← Solve(∃_X ∀_Y. F)         # check if X is a minimum ES
    if st:
        break
    else:
        I ← Reduce(Y)                     # reduce counterexample
        ℋ ← ℋ ∪ I                         # hit counterexample next time
return MSA ← μ_X
```

Algorithm

input : formula F
output: an MSA of F

```
ℋ ← ∅
while true:
    X ← MinHS(ℋ)                                # get a new MHS with MaxSAT
    Y ← var(F) \ X                              # take complement of X
    (st, μ_X) ← Solve(∃_X ∀_Y. F)               # check if X is a minimum ES
    if st:
        break
    else:
        I ← Reduce(Y)                           # no need to reduce counterexample
        ℋ ← ℋ ∪ Y                               # hit counterexample next time
return MSA ← μ_X
```

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\text{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$			

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*		

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y. F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$			

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$	*false*		

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$	*false*	$I \leftarrow \{a\}$	

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

MinHS($\mathcal{H}$)	Solve($\exists_X \forall_Y . F$)	$I \leftarrow$ Reduce(Y)	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$	*false*	$I \leftarrow \{a\}$	$\{\{b, c\}, \{a\}\}$

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\mathtt{MinHS}(\mathcal{H})$	$\mathtt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \mathtt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$	*false*	$I \leftarrow \{a\}$	$\{\{b, c\}, \{a\}\}$
$X \leftarrow \{a, c\}$			

Example

$$F = ((a + b \geqslant 0) \vee (c \leqslant 0)) \wedge ((a + b \geqslant 0) \vee (b - a \leqslant 0))$$

$$\mathrm{var}(F) = \{a, b, c\}$$

$\texttt{MinHS}(\mathcal{H})$	$\texttt{Solve}(\exists_X \forall_Y . F)$	$I \leftarrow \texttt{Reduce}(Y)$	$\mathcal{H} = \mathcal{H} \cup I$
			$\emptyset$
$X \leftarrow \{\emptyset\}$	*false*	$I \leftarrow \{b, c\}$	$\{\{b, c\}\}$
$X \leftarrow \{b\}$	*false*	$I \leftarrow \{a\}$	$\{\{b, c\}, \{a\}\}$
$X \leftarrow \{a, c\}$	*true*	$\{a = 1, c = 0\}$ is an MSA of F	

$$((b + 1 \geqslant 0) \vee (0 \leqslant 0)) \wedge ((b + 1 \geqslant 0) \vee (b - 1 \leqslant 0))$$

Experimental results

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ $=$ MINT $+$ bootstrapping with unit sets:

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. **MINT+** = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

- MISTRAL — state of the art
  - implemented in C++
  - branch-and-bound approach
  - targets LIA formulas

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

- MISTRAL — state of the art
  - implemented in C++
  - branch-and-bound approach
  - targets LIA formulas

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

- MISTRAL — state of the art
  - implemented in C++
  - branch-and-bound approach
  - targets LIA formulas

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
  - running Ubuntu Linux

Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

- MISTRAL — state of the art
  - implemented in C++
  - branch-and-bound approach
  - targets LIA formulas

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
  - running Ubuntu Linux
  - 3600s timeout

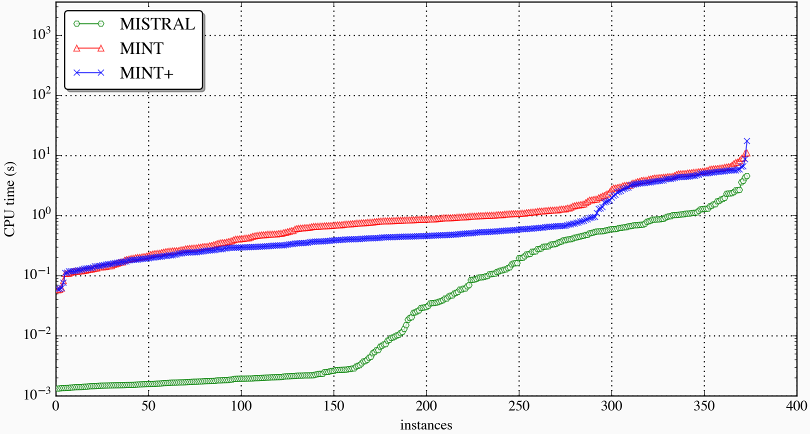
Experimental evaluation

- new IHS-based approach:
  1. MINT
  2. MINT+ = MINT + bootstrapping with unit sets:
- minimum hitting set engine — incremental MaxSAT
- implemented in PySMT:
  - supports Z3, CVC4, Yices2, etc.
  - not limited to LIA formulas

- MISTRAL — state of the art
  - implemented in C++
  - branch-and-bound approach
  - targets LIA formulas

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
  - running Ubuntu Linux
  - 3600s timeout
  - 10GByte memout

Original CAV12 benchmarks



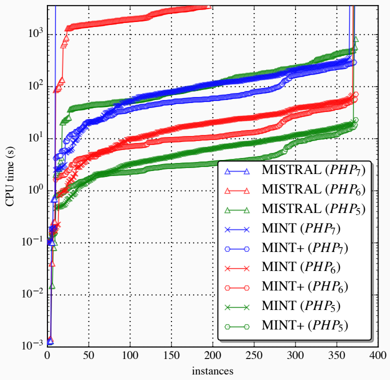
too easy!

Hardened instances (with PHP_n)

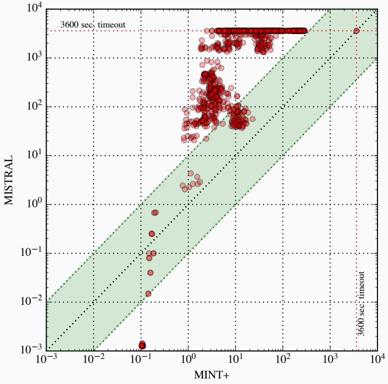
$\forall F \in \text{CAV12}$ consider $F^H = F \vee PHP_n$, $n \in \{5, 6, 7\}$

Hardened instances (with PHP_n)

$\forall F \in$ CAV12 consider $F^H = F \vee PHP_n$, $n \in \{5, 6, 7\}$



(a) cactus plot



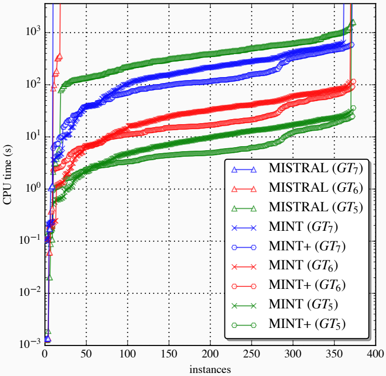
(b) scatter plot (MINT+ vs. MISTRAL)

Hardened instances (with GT_n)

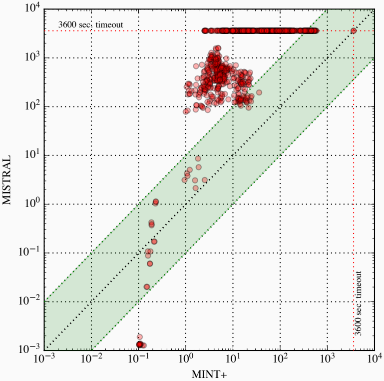
$\forall F \in \text{CAV12}$ consider $F^H = F \vee GT_n$, $n \in \{5, 6, 7\}$

Hardened instances (with GT_n)

$$\forall F \in \text{CAV12} \quad \text{consider } F^H = F \vee GT_n,\ n \in \{5, 6, 7\}$$



(a) cactus plot



(b) scatter plot (MINT+ vs. MISTRAL)

Performance of MINT+ vs. MISTRAL

	PHP_5	PHP_6	PHP_7
MINT+	373 *(4.74s)*	**373** ***(15.3s)***	**371** ***(>84.0s)***
MISTRAL	373 *(149.4s)*	195 *(>2784.1s)*	9 *(>3513.2s)*

	GT_5	GT_6	GT_7
MINT+	373 *(6.9s)*	**373** ***(24.0s)***	**371** ***(>165.2s)***
MISTRAL	373 *(421.9s)*	18 *(>3431.8s)*	9 *(>3474.6s)*

Summary and future work

Summary and future work

- *new* approach for MSA:

Summary and future work

- *new* approach for MSA:
  - based on implicit hitting sets
    - underlying technology — incremental MaxSAT

Summary and future work

- *new* approach for MSA:
  - based on implicit hitting sets
    - underlying technology — incremental MaxSAT
  - uses PySMT
    - supports Z3, CVC4, Yices2, etc.
    - not limited to LIA formulas

Summary and future work

- *new* approach for MSA:
  - based on implicit hitting sets
    - underlying technology — incremental MaxSAT
  - uses PySMT
    - supports Z3, CVC4, Yices2, etc.
    - not limited to LIA formulas
  - outperforms state of the art (B&B)

Summary and future work

- *new* approach for MSA:
  - based on implicit hitting sets
    - underlying technology — incremental MaxSAT
  - uses PySMT
    - supports Z3, CVC4, Yices2, etc.
    - not limited to LIA formulas
  - outperforms state of the art (B&B)

- better IHS-based algorithms

Summary and future work

- *new* approach for MSA:
  - based on implicit hitting sets
    - underlying technology — incremental MaxSAT
  - uses PySMT
    - supports Z3, CVC4, Yices2, etc.
    - not limited to LIA formulas
  - outperforms state of the art (B&B)

- better IHS-based algorithms
- more practical applications of MSA

Questions?