Non-Linear Arithmetic

Master Project

Simon Legner
Supervisor: Dr. Harald Zankl

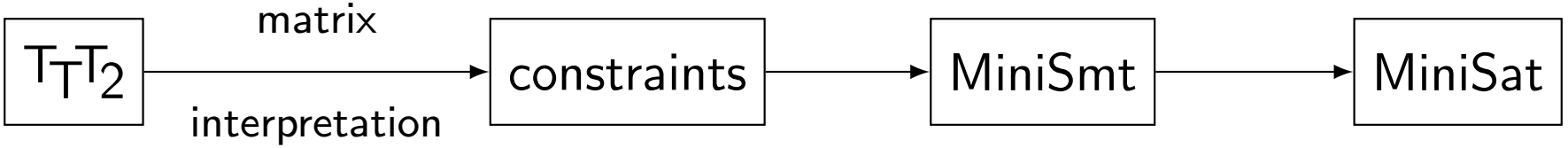
Computational Logic
Institute of Computer Science
University of Innsbruck

November 26, 2013

Problem Context

MiniSmt

- SMT solver for non-linear arithmetic
- domains $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, "$\mathbb{R}$" $(a + b\sqrt{2})$
  (decidable for $\mathbb{R}$, undecidable for $\mathbb{N}$)
- developed at Computational Logic group



Problem Context – Matrix Interpretation

- Term rewrite system

$$\mathsf{a}(\mathsf{a}(x)) \to \mathsf{a}(\mathsf{b}(\mathsf{a}(x)))$$

- Matrix interpretation $\mathcal{M}$

$$\mathsf{a}_{\mathcal{M}}(\vec{x}) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad \mathsf{b}_{\mathcal{M}}(\vec{x}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \vec{x}$$

orients rule strictly:

$$\mathsf{a}_{\mathcal{M}}(\mathsf{a}_{\mathcal{M}}(\vec{x})) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 3 \\ 1 \end{pmatrix} > \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \mathsf{a}_{\mathcal{M}}(\mathsf{b}_{\mathcal{M}}(\mathsf{a}_{\mathcal{M}}(\vec{x})))$$

- How to find this interpretation?

Problem Context – Matrix Interpretation

Arithmetic encoding of 2-dimensional interpretation


```
:formula (and (and (and (and (>= (+ x0 (+ (* x2 x0) (* x3 x1))) (+ x0 (+ (* x2 (+ x6 (+ (* x8 x0) (* x9
x1)))) (* x3 (+ x7 (+ (* x10 x0) (* x11 x1)))))))) (>= (+ x1 (+ (* x4 x0) (* x5 x1))) (+ x1 (+ (* x4 (+
x6 (+ (* x8 x0) (* x9 x1)))) (* x5 (+ x7 (+ (* x10 x0) (* x11 x1)))))))) (and (and (and (>= (+ (* x2 x2)
(* x3 x4)) (+ (* x2 (+ (* x8 x2) (* x9 x4))) (* x3 (+ (* x10 x2) (* x11 x4))))) (>= (+ (* x2 x3) (* x3
x5)) (+ (* x2 (+ (* x8 x3) (* x9 x5))) (* x3 (+ (* x10 x3) (* x11 x5)))))) (>= (+ (* x4 x2) (* x5 x4)) (+
(* x4 (+ (* x8 x2) (* x9 x4))) (* x5 (+ (* x10 x2) (* x11 x4)))))) (>= (+ (* x4 x3) (* x5 x5)) (+ (* x4 (+
(* x8 x3) (* x9 x5))) (* x5 (+ (* x10 x3) (* x11 x5))))))) (and (and (> (+ x0 (+ (* x2 x0) (* x3 x1))) (+
x0 (+ (* x2 (+ x6 (+ (* x8 x0) (* x9 x1)))) (* x3 (+ x7 (+ (* x10 x0) (* x11 x1))))))) (and (>= (+ x0 (+
(* x2 x0) (* x3 x1))) (+ x0 (+ (* x2 (+ x6 (+ (* x8 x0) (* x9 x1)))) (* x3 (+ x7 (+ (* x10 x0) (* x11
x1))))))) (>= (+ x1 (+ (* x4 x0) (* x5 x1))) (+ x1 (+ (* x4 (+ x6 (+ (* x8 x0) (* x9 x1)))) (* x5 (+ x7
(+ (* x10 x0) (* x11 x1)))))))) (and (and (and (>= (+ (* x2 x2) (* x3 x4)) (+ (* x2 (+ (* x8 x2) (* x9
x4))) (* x3 (+ (* x10 x2) (* x11 x4))))) (>= (+ (* x2 x3) (* x3 x5)) (+ (* x2 (+ (* x8 x3) (* x9 x5))) (*
x3 (+ (* x10 x3) (* x11 x5)))))) (>= (+ (* x4 x2) (* x5 x4)) (+ (* x4 (+ (* x8 x2) (* x9 x4))) (* x5 (+ (*
x10 x2) (* x11 x4)))))) (>= (+ (* x4 x3) (* x5 x5)) (+ (* x4 (+ (* x8 x3) (* x9 x5))) (* x5 (+ (* x10 x3)
(* x11 x5))))))))) (and (>= x2 1) (>= x8 1))))
```

Problem Context – Matrix Interpretation

SAT encoding

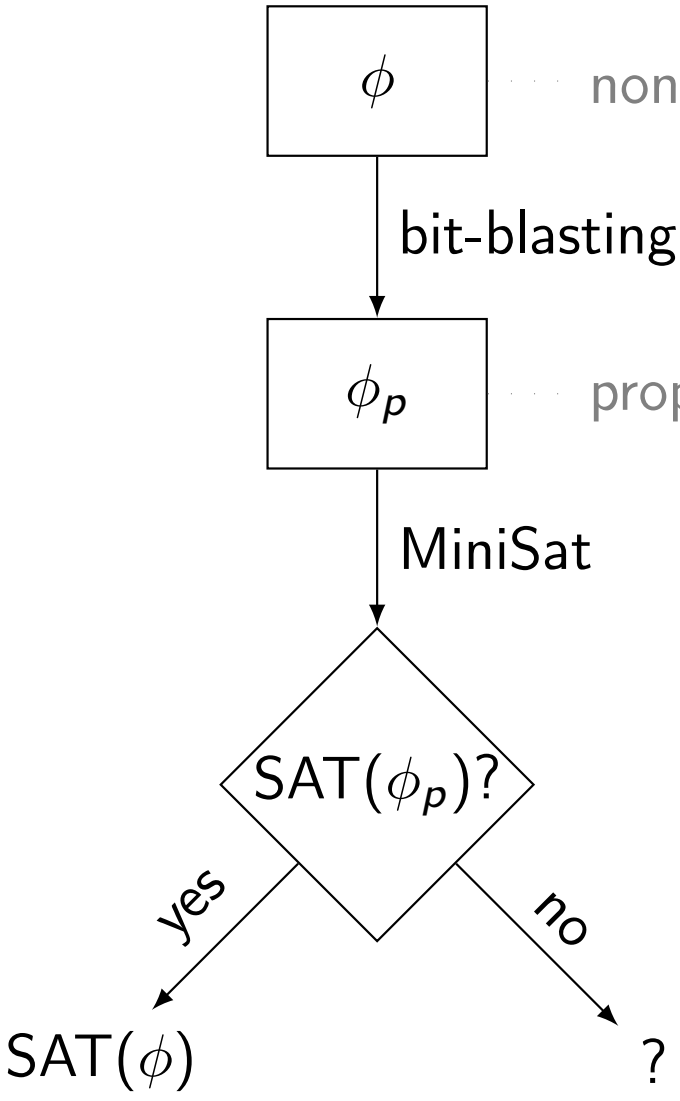
(and (and (and (and (and (or (or (or (and (and (not p65) (not p64)) (and p20 (not p62))) (and (and (and (not p65) (not p64)) (or (not p62) p20)) (and p21 (not p60)))) (and (and (and (and (not p65) (not p64)) (or (not p62) p20)) (or (not p60) p21)) (and p19 (not p58)))) (and (and (and (and (iff p19 p58) (iff p21 p60)) (iff p20 p62)) (not p64)) (not p65))) (or (or (or (and (and (not p99) (not p98)) (and p72 (not p96))) (and (and (and (not p99) (not p98)) (or (not p96) p72)) (and p73 (not p94)))) (and (and (and (and (not p99) (not p98)) (or (not p96) p72)) (or (not p94) p73)) (and p71 (not p92)))) (and (and (and (and (iff p71 p92) (iff p73 p94)) (iff p72 p96)) (not p98)) (not p99)))) (and (and (and (or (or (and (not p114) (and p100 (not p113))) (and (and (not p114) (or (not p113) p100)) (and p101 (not p111)))) (and (and (iff p101 p111) (iff p100 p113)) (not p114))) (or (or (and (not p129) (and p115 (not p128))) (and (and (not p129) (or (not p128) p115)) (and p116 (not p126)))) (and (and (iff p116 p126) (iff p115 p128)) (not p129)))) (or (or (and (not p140) (and p130 (not p139))) (and (and (not p140) (or (not p139) p130)) (and p131 (not p137)))) (and (and (iff p131 p137) (iff p130 p139)) (not p140)))) (or (or (and (not p151) (and p141 (not p150))) (and (and (not p151) (or (not p150) p141)) (and p142 (not p148)))) (and (and (iff p142 p148) (iff p141 p150)) (not p151))))) (and (and (or (or (and (and (not p65) (not p64)) (and p20 (not p62))) (and (and (and (not p65) (not p64)) (or (not p62) p20)) (and p21 (not p60)))) (and (and (and (and (not p65) (not p64)) (or (not p62) p20)) (or (not p60) p21)) (and p19 (not p58)))) (and (or (or (or (and (and (not p65) (not p64)) (and p20 (not p62))) (and (and (and (not p65) (not p64)) (or (not p62) p20)) (and p21 (not p60)))) (and (and (and (and (not p65) (not p64)) (or (not p62) p20)) (or (not p60) p21)) (and p19 (not p58)))) (and (and (and (and (iff p19 p58) (iff p21 p60)) (iff p20 p62)) (not p64)) (not p65))) (or (or (or (and (and (not p99) (not p98)) (and p72 (not p96))) (and (and (and (not p99) (not p98)) (or (not p96) p72)) (and p73 (not p94)))) (and (and (and (and (not p99) (not p98)) (or (not p96) p72)) (or (not p94) p73)) (and p71 (not p92)))) (and (and (and (and (iff p71 p92) (iff p73 p94)) (iff p72 p96)) (not p98)) (not p99))))) (and (and (and (or (or (and (not p114) (and p100 (not p113))) (and (and (not p114) (or (not p113) p100)) (and p101 (not p111)))) (and (and (iff p101 p111) (iff p100 p113)) (not p114))) (or (or (and (not p129) (and p115 (not p128))) (and (and (not p129) (or (not p128) p115)) (and p116 (not p126)))) (and (and (iff p116 p126) (iff p115 p128)) (not p129)))) (or (or (and (not p140) (and p130 (not p139))) (and (and (not p140) (or (not p139) p130)) (and p131 (not p137)))) (and (and (iff p131 p137) (iff p130 p139)) (not p140)))) (or (or (and (not p151) (and p141 (not p150))) (and (and (not p151) (or (not p150) p141)) (and p142 (not p148)))) (and (and (iff p142 p148) (iff p141 p150)) (not p151)))))) (and p13 p23)) (and (iff p46 (and p44 p14)) (and (iff p49 (and p43 p14)) (and (iff p109 (and p106 p14)) (and (iff p124 (and p121 p14)) (and (iff p77 p75) (and (iff p95 (and p89 p93)) (and (iff p60 (not (iff p53 p57))) (and (iff p80 (and p44 p67)) (and (iff p83 (and p43 p67)) (and (iff p135 (and p106 p67)) (and (iff p146 (and p121 p67)) (and (iff p39 (and (and p12 p37) (and p15 p38))) (and (iff p32 (and p30 p13)) (and (iff p35 (and p29 p13)) (and (iff p105 (and p102 p13)) (and (iff p120 (and p117 p13)) (and (iff p97 (and p90 p95)) (and (iff p75 (and p30 p66)) (and (iff p78 (and p29 p66)) (and (iff p133 (and p102 p66)) (and (iff p144 (and p117 p66)) (and (iff p74 (and p28 p66)) (and (iff p29 (and p25 p27)) (and (iff p92 (not (iff p15 p85))) (and (iff p149 (or (and p144 p146) (and (not (iff p144 p146)) p147))) (and (iff p111 (not (iff p104 p108))) (and (iff p106 (and (and p13 p37) (and p66 p38))) (and (iff p148 (not (iff p143 p145))) (and (iff p112 (or (and p105 p109) (and (not (iff p105 p109)) p110))) (and (iff p121 (and (and p14 p37) (and p67 p38))) (and (iff p129 p127) (and (iff p137 (not (iff p132 p134))) (and (iff p118 (not (iff (and p14 p23) (and p67 p24)))) (and (iff p103 (not (iff (and p13 p23) (and p66 p24)))) (and (iff p42 (not (iff p36 p40))) (and (iff p91 (and p15 p85)) (and (iff p43 (and p39 p41)) (and (iff p20 (and p16 p18)) (and (iff p56 p54) (and (iff p126 (not (iff p119 p123))) (and (iff p93 (and p87 p91)) (and (iff p73 (not (iff p68 p70))) (and (iff p26 (not (iff (and p12 p23) (and p15 p24)))) (and (iff p44 (not (iff p39 p41))) (and (iff p76 p74) (and (iff p136 (and p132 p134)) (and (iff p18 (and p12 p17)) (and (iff p51 (not (iff p33 p47))) (and (iff p48 p46) (and (iff p90 p88) (and (iff p34 p32) (and (iff p72 (and p68 p70)) (and (iff p122 (not (iff (and p14 p37) (and p67 p38)))) (and (iff p107 (not (iff (and p13 p37) (and p66 p38)))) (and (iff p123 (and p122 p14)) (and (iff p61 (and p55 p59)) (and (iff p84 (and p76 p81)) (and (iff p87 (not (iff (not (iff p77 p82)) p84))) (and (iff p58 (not (iff p12 p51))) (and (iff p63 (and p56 p61)) (and (iff p108 (and p107 p14)) (and (iff p114 p112) (and (iff p85 (not (iff p76 p81))) (and (iff p25 (and (and p12 p23) (and p15 p24))) (and (iff p50 (and p33 p47)) (and (iff p145 (and p122 p67)) (and (iff p40 (not (iff (and p12 p37) (and p15 p38)))) (and (iff p70 (and p15 p69)) (and (iff p82 p80) (and (iff p116 (not (iff (and p14 p13) (and p67 p14)))) (and (iff p89 (not (iff (not (iff p78 p83)) p86))) (and (iff p101 (not (iff (and p13 p13) (and p66 p14)))) (and (iff p147 (and p143 p145)) (and (iff p21 (not (iff p16 p18))) (and (iff p134 (and p107 p67)) (and (iff p128 (not (iff (not (iff p120 p124)) p125))) (and (iff p28 (not (iff p22 p26))) (and (iff p45 (and p42 p14)) (and (iff p102 (and (and p13 p23) (and p66 p24))) (and (iff p117 (and (and p14 p23) (and p67 p24))) (and (iff p142 (not (iff (and p14 p66) (and p67 p67)))) (and (iff p131 (not (iff (and p13 p66) (and p66 p67)))) (and (iff p65 p63) (and (iff p110 (and p104 p108)) (and (iff p17 (not (iff (and p12 p13) (and p15 p14)))) (and (iff p96 (not (iff p89 p93))) (and (iff p53 (not (iff (not (iff p34 p48)) p50))) (and (iff p47 p45) (and (iff p86 (or (and p77 p82) (and (not (iff p77 p82)) p84))) (and (iff p79 (and p42 p67)) (and (iff p41 (and p36 p40)) (and (iff p16 (and (and p12 p13) (and p15 p14))) (and (iff p59 (and p53 p57)) (and (iff p33 p31) (and (iff p139 (not (iff (not (iff p133 p135)) p136))) (and (iff p125 (and p119 p123)) (and (iff p68 (and (and p12 p66) (and p15 p67))) (and (iff p19 (not (iff p12 p17))) (and (iff p69 (not (iff (and p12 p66) (and p15 p67)))) (and (iff p55 (not (iff (not (iff p35 p49)) p52))) (and (iff p98 (not (iff p90 p95))) (and (iff p88 (or (and p78 p83) (and (not (iff p78 p83)) p86))) (and (iff p62 (not (iff p55 p59))) (and (iff p57 (and p12 p51)) (and (iff p127 (or (and p120 p124) (and (not (iff p120 p124)) p125))) (and (iff p119 (and p118 p13)) (and (iff p151 p149) (and (iff p71 (not (iff p15 p69))) (and (iff p100 (and (and p13 p13) (and p66 p14))) (and (iff p104 (and p103 p13)) (and (iff p99 p97) (and (iff p115 (and (and p14 p13) (and p67 p14))) (and (iff p130 (and (and p13 p66) (and p66 p67))) (and (iff p27 (and p22 p26)) (and (iff p81 p79) (and (iff p141 (and (and p14 p66) (and p67 p67))) (and (iff p150 (not (iff (not (iff p144 p146)) p147))) (and (iff p52 (or (and p34 p48) (and (not (iff p34 p48)) p50))) (and (iff p64 (not (iff p56 p61))) (and (iff p143 (and p118 p66)) (and (iff p94 (not (iff p87 p91))) (and (iff p132 (and p103 p66)) (and (iff p113 (not (iff (not (iff p105 p109)) p110))) (and (iff p138 (or (and p133 p135) (and (not (iff p133 p135)) p136))) (and (iff p31 (and p28 p13)) (and (iff p54 (or (and p35 p49) (and (not (iff p35 p49)) p52))) (and (iff p140 p138) (iff p30 (not (iff p25 p27)))

Contents

$((a + b) = 3)$ . . . linear constraint

$((a > 10) \wedge ((a \times b) < 20))$ . . . non-linear constraint

MiniSmt [Zankl, Middeldorp, 2010]



$\neg\,\mathsf{SAT}(\phi_p)$

$\vee\ \neg\,\mathsf{SAT}(\phi)$

$\vee$ too few bits for arithmetic variables

Example 0

$\phi = a + b = 3$ ... guess bit-width 2 for standard binary encoding

$\rightsquigarrow [a_1, a_0] + [b_1, b_0] = [\top, \top]$ ... $a = 2^1 \cdot a_1 + 2^0 \cdot a_0$

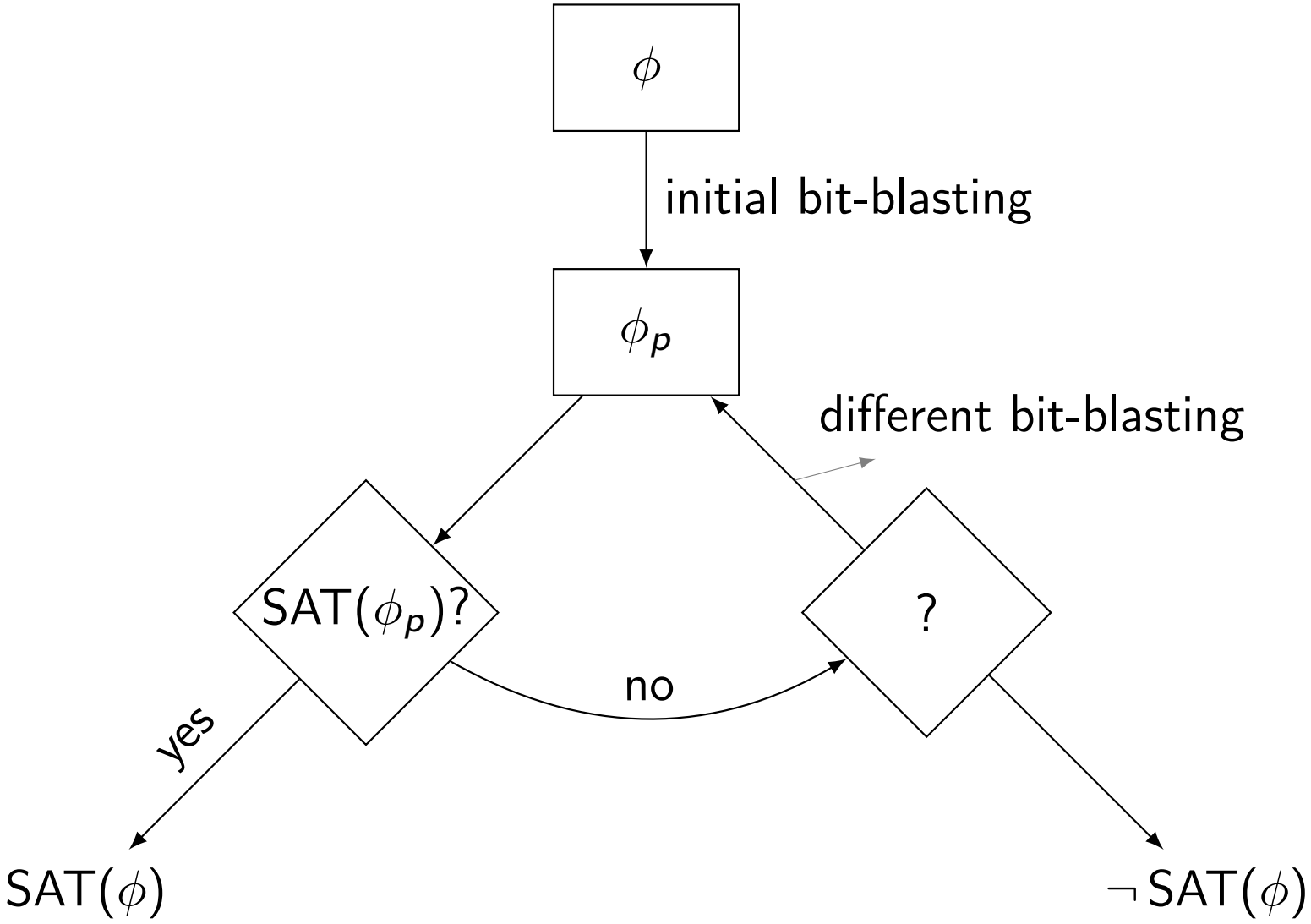
$\rightsquigarrow [a_1, a_0] + [b_1, b_0] = [s_2, s_1, s_0] \wedge [s_2, s_1, s_0] = [\bot, \top, \top]$

$$\begin{aligned}\phi_p = {} & (s_0 \leftrightarrow a_0 \oplus b_0) \wedge (c_1 \leftrightarrow a_0 \wedge b_0) \wedge \\ & (s_1 \leftrightarrow a_1 \oplus b_1 \oplus c_1) \wedge (c_2 \leftrightarrow (a_1 \wedge b_1) \vee (a_1 \wedge c_1) \vee (b_1 \wedge c_1)) \wedge \\ & (s_2 \leftrightarrow c_2) \wedge \\ & (s_0 \leftrightarrow \top) \wedge (s_1 \leftrightarrow \top) \wedge (s_2 \leftrightarrow \bot)\end{aligned}$$

$\rightsquigarrow$ CNF

$\rightsquigarrow \mathrm{SAT}(\phi_p)$

$\rightsquigarrow \mathrm{SAT}(\phi),\ \{a \mapsto 3, b \mapsto 0\}$


ϕ
initial bit-blasting
ϕ_p
different bit-blasting
SAT(ϕ_p)?
?
no
yes
SAT(ϕ)
$\neg$ SAT(ϕ)

Unsatisfiability Proof $\mathcal{C}_p$

some SAT solvers extract a set of propositional variables $\mathcal{C}_p$ (unsatisfiable core) if the formula is unsatisfiable

Candidate Variables $\mathcal{C}_\phi$

compute candidate variables $\mathcal{C}_\phi$ from $\mathcal{C}_p$ for incremental approach (to compute different bit-blasting, or determine unsatisfiability)

Example I: $a + 2 > 6$

$$
\begin{aligned}
\phi &= a + 2 > 6 \\
&\rightsquigarrow [a_1, a_0] + [\top, \bot] > [\top, \top, \bot] \\
&\rightsquigarrow [a_1, a_0] + [\top, \bot] = [s_2, s_1, s_0] \wedge [s_2, s_1, s_0] > [\top, \top, \bot] \\
&\rightsquigarrow \phi_p \\
&\rightsquigarrow \neg \mathsf{SAT}(\phi_p),\ \mathcal{C}_p \\
&\rightsquigarrow \mathcal{C}_\phi = \{a\}
\end{aligned}
$$

$$
\begin{aligned}
\phi &\rightsquigarrow [a_2, a_1, a_0] + [\top, \bot] > [\top, \top, \bot] \\
&\rightsquigarrow \phi'_p \\
&\rightsquigarrow \mathsf{SAT}(\phi'_p) \\
&\rightsquigarrow \mathsf{SAT}(\phi),\ \{a \mapsto 6\}
\end{aligned}
$$

Example II: $a > b > c \geqslant 0 \wedge d = 1$

$$\phi = (a > b) \wedge (b > c) \wedge (c \geqslant 0) \wedge (d = 1)$$

$$\rightsquigarrow ([a_0] > [b_0]) \wedge ([b_0] > [c_0]) \wedge ([c_0] \geqslant [\bot]) \wedge ([d_0] = [\top])$$

$$\rightsquigarrow \phi_p$$

$$\rightsquigarrow \neg\,\mathsf{SAT}(\phi_p),\ \mathcal{C}_p$$

$$\rightsquigarrow \mathcal{C}'_\phi = \{b\}$$

$$\rightsquigarrow \mathcal{C}_\phi = \{a, b, c\}$$ . . . take all variables from parent boolean connectives

$$\phi \rightsquigarrow ([a_1, a_0] > [b_1, b_0]) \wedge ([b_1, b_0] > [c_1, c_0]) \wedge ([c_1, c_0] \geqslant [\bot]) \wedge [d_0] = [\top]$$

$$\rightsquigarrow \phi'_p$$

$$\rightsquigarrow \mathsf{SAT}(\phi'_p)$$

$$\rightsquigarrow \mathsf{SAT}(\phi),\ \{a \mapsto 3, b \mapsto 2, c \mapsto 0\}$$

Example III: $a > b > c \wedge 2 > a, b, c$

$$\phi = (a > b) \wedge (b > c) \wedge (2 > a) \wedge (2 > b) \wedge (2 > c)$$

$$\rightsquigarrow ([a_1, a_0] > [b_1, b_0]) \wedge ([b_1, b_0] > [c_1, c_0]) \wedge$$
$$([\top, \bot] > [a_1, a_0]) \wedge ([\top, \bot] > [b_1, b_0]) \wedge ([\top, \bot] > [c_1, c_0])$$

$$\rightsquigarrow \phi_p$$

$$\rightsquigarrow \neg \mathsf{SAT}(\phi_p),\ \mathcal{C}_p$$

$$\rightsquigarrow \mathcal{C}_\phi = \{a, b, c\}$$

SAT encoding covers range of a, b, c $(0 \leqslant a, b, c \leqslant 1)$

$$\rightsquigarrow \neg \mathsf{SAT}(\phi)$$

Unsatisfiability Proof $\mathcal{C}_p$

obtain $\mathcal{C}_p$ from SAT solver obtain ~~$\mathcal{C}_p$~~ $\mathcal{C}_{\text{CNF}}$ from SAT solver, compute $\mathcal{C}_p$
MiniSat ✗, PicoSat ✓

Candidate Variables $\mathcal{C}_\phi$

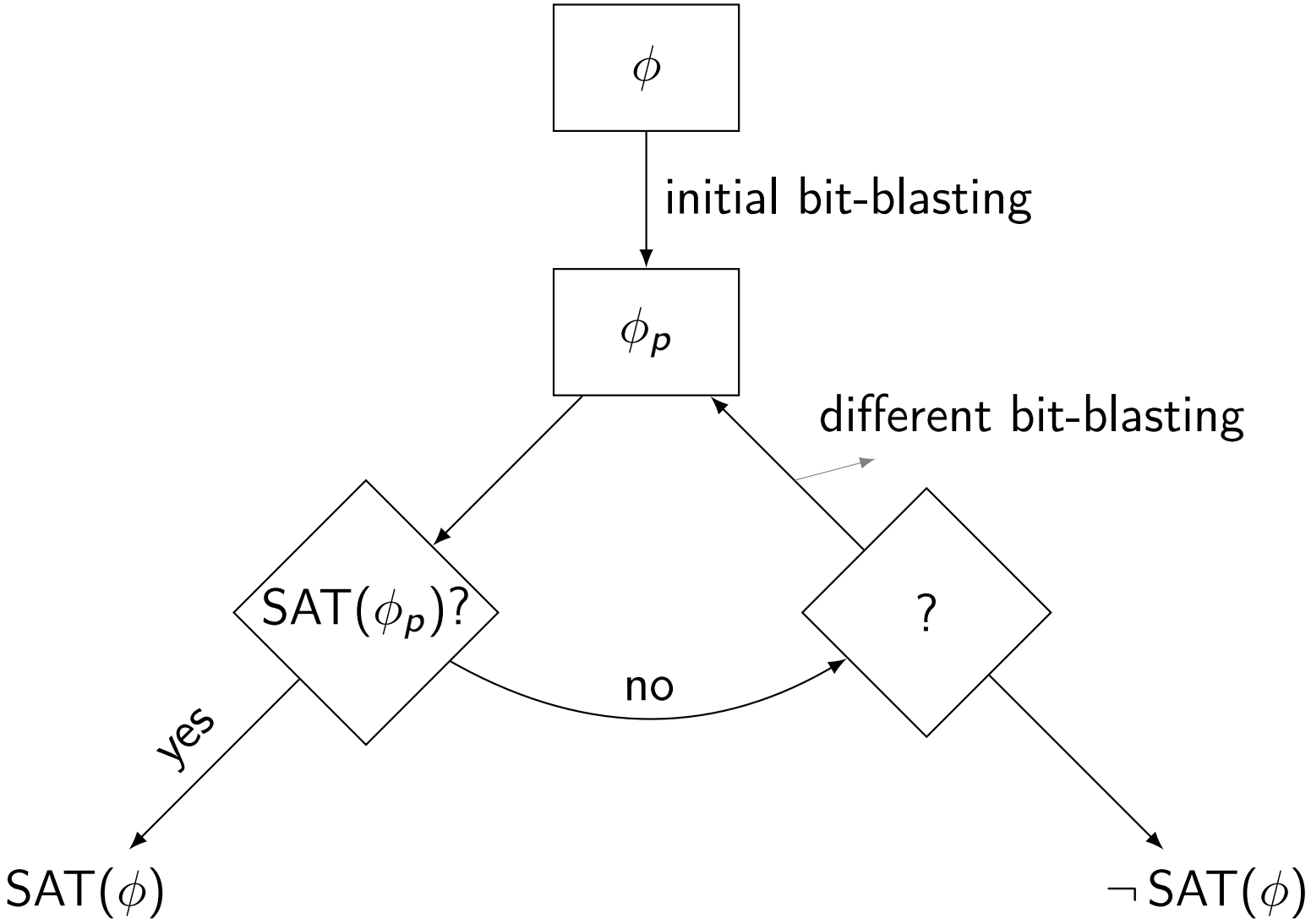
$\forall p \in \mathcal{C}_p$: determine corresponding arithmetic variable a,
take all variables from parent boolean connective

Incremental Approach

- determine unsatisfiability if bit-blasting covers range of $\mathcal{C}_\phi$
- use $\mathcal{C}_\phi$ to increase the bit-width of some variables (different strategies)

Main Results

- bit-blasting with incremental approach is still
  - sound
  - incomplete
- promising evaluation results


ϕ
initial bit-blasting
ϕ_p
different bit-blasting
SAT(ϕ_p)?
?
no
yes
SAT(ϕ)
$\neg$ SAT(ϕ)

Evaluation (SAT/¬SAT)

Statistics for leipzig

	--	+2	*4
ib 1	96	155	148
ib 2	129	151	150
ib 3	153	151	153
ib 4	147	147	146

Statistics for calypto

	--	+2	*4
ib 1	27/0	41/1	41/1
ib 2	38/0	46/1	42/1
ib 3	40/0	42/1	47/0
ib 4	38/0	42/1	40/1

- `ib n` initial bit-width
- `--` no refinement of variables
- `+2` refine variables by adding 2 bits
- `*4` refine variables by multiplying the number with 4

Related Work

Bryant et al.

bit-vector arithmetic $\longrightarrow$ SAT

Borralleras et al.

non-linear arithmetic $\longrightarrow$ linear arithmetic

Summary

extended MiniSmt by iterative procedure and criterion for unsatisfiability

□

Thank you!