Integration of Fourier-Motzkin based Variable-Elimination into iSAT

Albert-Ludwigs-Universität Freiburg

Karsten Scheibler, Felix Neubauer, Bernd Becker

Chair of Computer Architecture

SWIM 2012

Outline

What iSAT is

How iSAT works

Variable-Elimination

- Boolean Variable-Elimination
- Variable-Elimination based on Fourier-Motzkin
- One Constraint per Clause Form

Experimental Results

Conclusion & Future Work

What iSAT is

iSAT can be used as a (bounded) model checking tool for hybrid systems

What iSAT is

iSAT can be used as a (bounded) model checking tool for hybrid systems

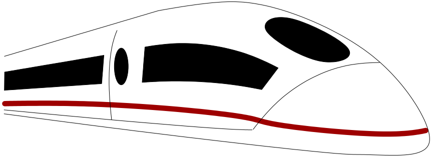
Hybrid system: continuous and discrete dynamic behaviour

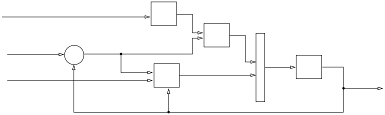
What iSAT is

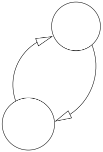
iSAT can be used as a (bounded) model checking tool for hybrid systems

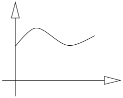
Hybrid system:	continuous and discrete dynamic behaviour
Model Checking:	check if a model of a system satisfies desired properties

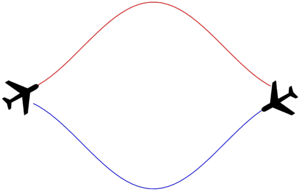
Examples



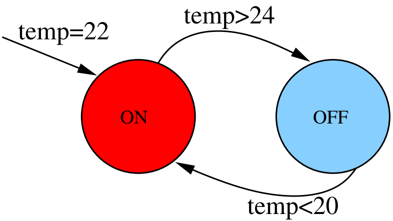


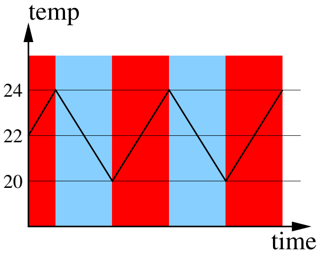




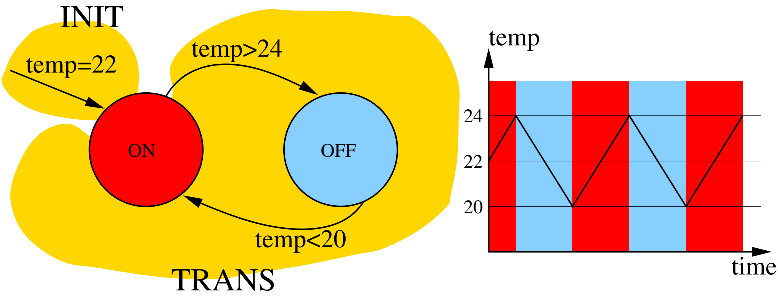


Examples

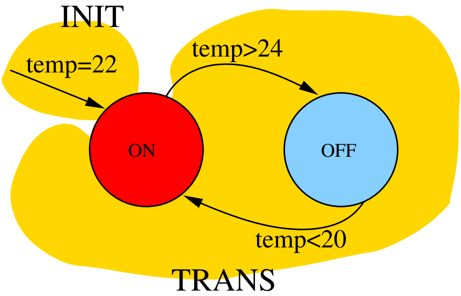


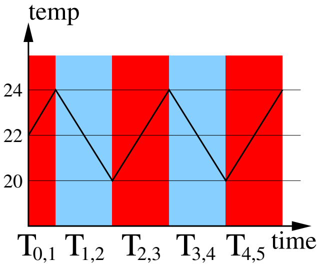


Examples



Examples


INIT
temp=22
temp>24
ON
OFF
temp<20
TRANS


temp
24
22
20
$T_{0,1}$ $T_{1,2}$ $T_{2,3}$ $T_{3,4}$ $T_{4,5}$ time

Bounded Model Checking

- Given by the model: INIT, TRANS

Bounded Model Checking

- Given by the model: INIT, TRANS
- Desired property: PROP

Bounded Model Checking

- Given by the model: INIT, TRANS
- Desired property: PROP
- iSAT solves the following formulas:
  - INIT $\wedge \neg$PROP
  - INIT $\wedge$ TRANS$_{0,1}$ $\wedge \neg$ PROP
  - INIT $\wedge$ TRANS$_{0,1} \wedge$ TRANS$_{1,2}$ $\wedge \neg$ PROP
  - INIT $\wedge$ TRANS$_{0,1} \wedge$ TRANS$_{1,2} \wedge$ TRANS$_{2,3}$ $\wedge \neg$ PROP
  - ...
- If one formula is satisfiable the desired property was violated

How iSAT works

- iSAT uses internally a conjunction of clauses

How iSAT works

- iSAT uses internally a conjunction of clauses
- every clause is a disjunction of atoms

How iSAT works

- iSAT uses internally a conjunction of clauses
- every clause is a disjunction of atoms
- every atom is either:
  - a boolean variable: a
  - a negated boolean variable: $\neg a$
  - a simple bound: $x < 5$
  - an arithmetic constraint: $u = v^3,\ y = sin(x),\ z = x + y,\ \ldots$

How iSAT works

- iSAT uses internally a conjunction of clauses
- every clause is a disjunction of atoms
- every atom is either:
  - a boolean variable: a
  - a negated boolean variable: $\neg a$
  - a simple bound: $x < 5$
  - an arithmetic constraint: $u = v^3,\ y = sin(x),\ z = x + y,\ \ldots$
- every integer and real variable is bounded

Small example

- $a, b \in \{\texttt{true}, \texttt{false}\}, \quad x \in [3,7], y \in [-2,49]$

  $(a \vee \neg b) \wedge (a \vee (y \leq 25)) \wedge (b \vee (y = x^2))$

Small example

- $a, b \in \{\texttt{true}, \texttt{false}\}, \quad x \in [3,7], y \in [-2,49]$

  $(a \vee \neg b) \wedge (a \vee (y \leq 25)) \wedge (b \vee (y = x^2))$

- Decision: $a = \mathit{false}$

Small example

- $a, b \in \{\texttt{true}, \texttt{false}\}, \quad x \in [3,7], y \in [-2,49]$

  $(a \vee \neg b) \wedge (a \vee (y \leq 25)) \wedge (b \vee (y = x^2))$

- Decision: $a = \mathit{false}$

- Deduction: $b = \mathit{false}$, $y \leq 25$, $y = x^2$

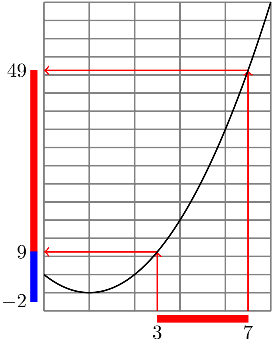
Small example

- $a, b \in \{\texttt{true}, \texttt{false}\}, \quad x \in [3,7], y \in [-2,49]$

  $(a \vee \neg b) \wedge (a \vee (y \leq 25)) \wedge (b \vee (y = x^2))$
- Decision: $a = \mathit{false}$
- Deduction: $b = \mathit{false}$, $y \leq 25$, $y = x^2$
- learn from conflicts already found (not shown in this example)

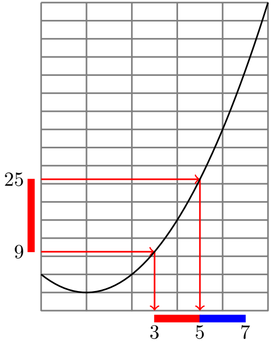
Small example (ICP)

$y = x^2$:



$x \in [3,7] \wedge y \in [-2,49]$

$\rightsquigarrow \quad y \geq 9$



$x \in [3,7] \wedge y \in [9,25]$

$\rightsquigarrow \quad x \leq 5$

How iSAT works (2)

- generalization of conflict-driven clause-learning (CDCL) framework as used in propositional SAT-Solvers

How iSAT works (2)

- generalization of conflict-driven clause-learning (CDCL) framework as used in propositional SAT-Solvers
- tight integration of interval constraint propagation (ICP) for arithmetic constraints into the solver core

How iSAT works (2)

- generalization of conflict-driven clause-learning (CDCL) framework as used in propositional SAT-Solvers
- tight integration of interval constraint propagation (ICP) for arithmetic constraints into the solver core
- use optimizations from propositional SAT-Solvers (non-chronological backtracking, two-watched literal scheme, restarts, . . .)

How iSAT works (3)

- possible results:
  - an unresolvable conflict is found (UNSATISFIABLE)
  - all variables are point intervals (SATISFIABLE)
  - intervals are small enough (CANDIDATE SOLUTION)

How iSAT works (3)

- possible results:
  - an unresolvable conflict is found (UNSATISFIABLE)
  - all variables are point intervals (SATISFIABLE)
  - intervals are small enough (CANDIDATE SOLUTION)

- iSAT will return a CANDIDATE SOLUTION for problems like $((x < y) \wedge (y < z) \wedge (z < x))$, because of interval arithmetic

How iSAT works (3)

- possible results:
  - an unresolvable conflict is found (UNSATISFIABLE)
  - all variables are point intervals (SATISFIABLE)
  - intervals are small enough (CANDIDATE SOLUTION)

- iSAT will return a CANDIDATE SOLUTION for problems like $((x < y) \wedge (y < z) \wedge (z < x))$, because of interval arithmetic

- improve perfomance: solve more benchmarks and/or get more conclusive answers

Variable-Elimination

- used to reduce number of variables and clauses
- because smaller problems are solved faster usually
- today boolean variable elimination is a standard preprocessing technique in propositional SAT-Solvers

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a
- do a full resolution (every a-clause with every $\neg a$-clause):
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee b)$) = $(b \vee \neg c \vee b)$ = $(b \vee \neg c)$
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee c)$) = $(b \vee \neg c \vee c)$ = `true`

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a
- do a full resolution (every a-clause with every $\neg a$-clause):
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee b)$) = $(b \vee \neg c \vee b)$ = $(b \vee \neg c)$
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee c)$) = $(b \vee \neg c \vee c)$ = `true`
- ignore clauses containing a tautology, in this example:
  $(b \vee \neg c \vee c)$

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a
- do a full resolution (every a-clause with every $\neg a$-clause):
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee b)$) = $(b \vee \neg c \vee b)$ = $(b \vee \neg c)$
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee c)$) = $(b \vee \neg c \vee c)$ = `true`
- ignore clauses containing a tautology, in this example:
  $(b \vee \neg c \vee c)$
- all other clauses are kept unchanged

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a
- do a full resolution (every a-clause with every $\neg a$-clause):
  resolve$((a \vee b \vee \neg c), (\neg a \vee b))$ = $(b \vee \neg c \vee b)$ = $(b \vee \neg c)$
  resolve$((a \vee b \vee \neg c), (\neg a \vee c))$ = $(b \vee \neg c \vee c)$ = `true`
- ignore clauses containing a tautology, in this example:
  $(b \vee \neg c \vee c)$
- all other clauses are kept unchanged

$\rightsquigarrow$ $F_2 = (b \vee \neg c) \wedge (b \vee e)$

Boolean Variable-Elimination

- boolean formula in conjunctive normal form (CNF):
  $F_1 = (a \vee b \vee \neg c) \wedge (\neg a \vee b) \wedge (\neg a \vee c) \wedge (b \vee e)$
- want to eliminate boolean variable a
- do a full resolution (every a-clause with every $\neg a$-clause):
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee b)$) = $(b \vee \neg c \vee b)$ = $(b \vee \neg c)$
  resolve($(a \vee b \vee \neg c)$, $(\neg a \vee c)$) = $(b \vee \neg c \vee c)$ = `true`
- ignore clauses containing a tautology, in this example: $(b \vee \neg c \vee c)$
- all other clauses are kept unchanged

$\rightsquigarrow$ $F_2 = (b \vee \neg c) \wedge (b \vee e)$

- F_2 not equivalent to F_1, but: F_2 satisfiable $\Leftrightarrow$ F_1 satisfiable

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$
- want to eliminate real variable x

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$
- want to eliminate real variable x
- do a full resolution (every lower bound of x with every upper bound of x):
  resolve($(y < x)$, $(x < v + w)$) = $(y < v + w)$
  resolve($(y < x)$, $(x < z)$) = $(y < z)$

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$
- want to eliminate real variable x
- do a full resolution (every lower bound of x with every upper bound of x):
  resolve($(y < x)$, $(x < v + w)$) = $(y < v + w)$
  resolve($(y < x)$, $(x < z)$) = $(y < z)$
- all other inequalities are kept unchanged

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$
- want to eliminate real variable x
- do a full resolution (every lower bound of x with every upper bound of x):
  resolve($(y < x)$, $(x < v + w)$) = $(y < v + w)$
  resolve($(y < x)$, $(x < z)$) = $(y < z)$
- all other inequalities are kept unchanged

$\rightsquigarrow$ $F_2 = (y < v + w) \wedge (y < z) \wedge (v < u)$

Variable-Elimination based on Fourier-Motzkin

- rewrite equations and inequalities if needed to get $<$ or $\leq$ as relational operator
- conjunction of inequalities:
  $F_1 = (y < x) \wedge (x < v + w) \wedge (x < z) \wedge (v < u)$
- want to eliminate real variable x
- do a full resolution (every lower bound of x with every upper bound of x):
  resolve($(y < x)$, $(x < v + w)$) = $(y < v + w)$
  resolve($(y < x)$, $(x < z)$) = $(y < z)$
- all other inequalities are kept unchanged

$\rightsquigarrow$ $F_2 = (y < v + w) \wedge (y < z) \wedge (v < u)$

- F_2 not equivalent to F_1, but: F_2 satisfiable $\Leftrightarrow$ F_1 satisfiable

Variable-Elimination based on Fourier-Motzkin

Handling of $<$ and $\leq$:

- $y < x < z \quad \rightsquigarrow \quad y < z$
- $y < x \leq z \quad \rightsquigarrow \quad y < z$
- $y \leq x < z \quad \rightsquigarrow \quad y < z$
- $y \leq x \leq z \quad \rightsquigarrow \quad y \leq z$

One Constraint per Clause Form

But iSAT processes arbitrary boolean combinations of arithmetic constraints:

$$F = (a \vee (y < x) \vee \neg b) \wedge ((x < v + w) \vee \neg c \vee (x < s \cdot z)) \wedge (v < sin(u^2)) \wedge (\neg d \vee (x < 8) \vee e)$$

How to handle that ?

One Constraint per Clause Form

But iSAT processes arbitrary boolean combinations of arithmetic constraints:

$$F = (a \vee (y < x) \vee \neg b) \ \wedge \ ((x < v + w) \vee \neg c \vee (x < s \cdot z)) \ \wedge$$
$$(v < sin(u^2)) \ \wedge \ (\neg d \vee (x < 8) \vee e)$$

How to handle that ?

$\rightsquigarrow$ One Constraint per Clause Form (OCCF)

One Constraint per Clause Form

- every clause may contain up to one arithmetic constraint

One Constraint per Clause Form

- every clause may contain up to one arithmetic constraint
- use boolean helper variables if needed:

One Constraint per Clause Form

- every clause may contain up to one arithmetic constraint
- use boolean helper variables if needed:

$$F_1 = (a \vee (y < x) \vee \neg b) \wedge ((x < v + w) \vee \neg c \vee (x < s \cdot z)) \wedge \\ (v < sin(u^2)) \wedge (\neg d \vee (x < 8) \vee e)$$

$$F_2 = (a \vee (y < x) \vee \neg b) \wedge ((x < v + w) \vee \neg c \vee h) \wedge \\ (\neg h \vee (x < s \cdot z)) \wedge (v < sin(u^2)) \wedge (\neg d \vee (x < 8) \vee e)$$

FM with OCCF

to eliminate an integer or real variable x in an OCCF:

- do full resolution

FM with OCCF

to eliminate an integer or real variable x in an OCCF:

- do full resolution
- resolve every clause containing a lower bound of x with every clause containing an upper bound of x

FM with OCCF

to eliminate an integer or real variable x in an OCCF:

- do full resolution
- resolve every clause containing a lower bound of x with every clause containing an upper bound of x
- resolved clause contains
  - all boolean variables of both origin clauses
  - the result of resolve(x_{lower}, x_{upper})

$$(a \vee (y < x) \vee \neg b) \ \wedge \ ((x < v + w) \vee \neg c \vee h)$$

$$\rightsquigarrow \quad (a \vee \neg b \vee \neg c \vee h \vee (y < v + w))$$

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$
2. select a “good” variable and try to eliminate it

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$
2. select a “good” variable and try to eliminate it
3. if this variable occurs in too many clauses, skip it

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \dots$
2. select a “good” variable and try to eliminate it
3. if this variable occurs in too many clauses, skip it
4. do full resolution (ignore clauses containing a tautology)

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$
2. select a “good” variable and try to eliminate it
3. if this variable occurs in too many clauses, skip it
4. do full resolution (ignore clauses containing a tautology)
5. count resulting clauses, discard if too many clauses

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$
2. select a “good” variable and try to eliminate it
3. if this variable occurs in too many clauses, skip it
4. do full resolution (ignore clauses containing a tautology)
5. count resulting clauses, discard if too many clauses
6. otherwise replace original clauses with the resolved clauses

Implementation in iSAT

1. mark variables as “bad” if they occur in non-linear subexpressions like $sin(x), y \cdot z, \ldots$
2. select a “good” variable and try to eliminate it
3. if this variable occurs in too many clauses, skip it
4. do full resolution (ignore clauses containing a tautology)
5. count resulting clauses, discard if too many clauses
6. otherwise replace original clauses with the resolved clauses
7. goto 2

Experimental Results

We wanted:

- to solve more benchmarks
- to have more conclusive answers (SATISFIABLE or UNSATISFIABLE)

Experimental Results

We got:

more solved benchmarks and more conclusive answers!

	without FM-VE	with FM-VE	-/+
SATISFIABLE	6	8	+2
UNSATISFIABLE	136	146	+10
CANDIDATE SOLUTION	256	253	-3
$\sum$	398	407	+9

(overall 543 Benchmarks, Timeout 900 seconds)

Conclusion & Future Work

- solved more benchmarks
- got more conclusive answers
- currently variables are eliminated in order of appearence, examine heuristics for a better selection
- include subsumption checks as used in propositional SAT-Solvers
- compare Fourier-Motzkin VE to Loos-Weispfenning VE

References

(1) Neubauer, F.: Anwendung der Variablenelimination nach Fourier-Motzkin in einem Intervall-Constraint-Solver, bachelor thesis - Universität Freiburg (2012)

(2) Bruttomesso, R.: An Extension of the Davis-Putnam Procedure and its Application to Preprocessing in SMT - Universita della Svizzera Italiana (2009)