Efficient Local Search for Nonlinear Real Arithmetic

Zhonghan Wang, Bohua Zhan, Bohan Li, Shaowei Cai

Institute of Software, Chinese Academy of Sciences

VMCAI 2024   15 January 2024

Outline

1. Problem - Nonlinear Real Arithmetic
   - Syntax of SMT(NRA)
   - Fragment of Local Search
2. Incremental Computation of Variable Scores
   - Scoring Boundary for Arithmetic Variable
   - Incremental Computation
3. Temporary Relaxation of Equality (Non-Strick) Constraints
   - Value Complexity in Local Search
   - Relaxation Method
4. Experiment

Outline

1. Problem - Nonlinear Real Arithmetic
   - Syntax of SMT(NRA)
   - Fragment of Local Search
2. Incremental Computation of Variable Scores
   - Scoring Boundary for Arithmetic Variable
   - Incremental Computation
3. Temporary Relaxation of Equality (Non-Strick) Constraints
   - Value Complexity in Local Search
   - Relaxation Method
4. Experiment

Syntax of SMT(NRA)

polynomial: $p ::= x \mid c \mid p + p \mid p - p \mid p \times p$
atoms: $a ::= b \mid p = 0 \mid p > 0 \mid p < 0$
formula: $f ::= a \mid \neg f \mid f \wedge f \mid f \vee f$

SMT: Determine whether the formula is satisfied by some assignment (local search focuses), or prove unsat

Example:
$x^2 + y^2 \leq 1 \wedge x + y < 1 \wedge x + z > 0$
assignment with $\{x \to 0, y \to 0, z \to 1\}$ satisfies all clauses.

Fragment of Local Search (1)

Input : A set of clauses F
Output: An assignment satisfying F, or failure
Initialize assignment to variables;
while $\top$ **do**
  if *all clauses satisfied* **then**
    return *success with assignment;*
  end
  if *time or step limit reached* **then**
    return *failure;*
  end
  Critical move procedure.
end

Algorithm 1: Basic Fragment of Local Search[a]

[a] Shaowei Cai, Bohan Li, and Xindi Zhang. "Local Search for SMT on Linear Integer Arithmetic". In: *Computer Aided Verification - 34th International Conference, CAV*. ed. by Sharon Shoham and Yakir Vizel. Springer, 2022.

Fragment of Local Search (2)

$var, new\_value, score \leftarrow$ best move according to make-break score;
if *score* $>$ *0* **then**
  Move var to new_value;
end
else
  Update clause weight;
end

repeat
  $cls \leftarrow$ random unsatisfied clause;
  $var, new\_value, score \leftarrow$ critical move making cls satisfied;
  if *score* $\neq -\infty$ **then**
    move var to new_value;
  end
until *3 times*;
if *no move performed* **then**
  Move some variables in unsatisfied clauses;
end

Local Search for SAT and SMT

Problem / LS	SAT	SMT
Operation (Move)	Flip	Critical Move
Score Definition	Weighted unsat clauses	
Score Computation	Cached score	No Caching, time costly

What LS for SAT brings us:

Maintain scoring information after each iteration.

Difficulty:

Predetermine critical move shift value.

Our Solution:

Introduce Scoring Boundaries.

Outline

Infeasible Set

Definition

infeasible set[1] of a clause c with respect to an assignment $asgn$ is the set of values that the variables in c can take under $asgn$ such that c is unsatisfied.

Example

Current assignment: $\{x \mapsto 1\}$
Calculate infeasible set for y:

- $x^2 + y^2 \leq 1$: $(-\infty, 0) \cup (0, \infty)$.
- $x + y < 1$: $[0, \infty)$.

If we choose values from infeasible set, the satisfied clause will be unsatisfied, which changes the whole score.

[1] Dejan Jovanovic and Leonardo Mendonça de Moura. "Solving Non-linear Arithmetic". In: *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June*

Make-break Intervals

Definition

make-break interval[2] is a combination of (in)feasible intervals of arithmetic varaible x with respect to **all clauses**.

Example

Current assignment: $\{x \mapsto 1, y \mapsto 1, z \mapsto 1\}$
Calculate infeasible set for each clause.

- $x^2 + y^2 \leq 1$ (unsat): $(-\infty, 0) \cup (0, \infty)$.
- $x + y < 1$ (unsat): $[0, \infty)$.
- $x + z > 0$ (sat): $(-\infty, -1]$.

Combined information: x: $(-\infty, -1] \mapsto 0$, $(-1, 0) \mapsto 1$, $\underline{[0, 0] \mapsto 1,\ (0, \infty) \mapsto 0}$.

[2] Bohan Li and Shaowei Cai. "Local Search For SMT On Linear and Multilinear Real Arithmetic". In: *CoRR* abs/2303.06676 (2023). arXiv: 2303.06676.

Traditional Computation

Input : unsat clauses F
Output: Best critical move (variable, value)
foreach *variable* v *in unsat clauses* **do**
  foreach *unsat clause* c *with* v **do**
    Compute interval-score info of v in c.
  end
  Combine interval-score information.
  Update best var-value move.
end
return *best critical move*

Repeated computation:

- variable's (in)feasible set
- clause's sat staus

Boundary

Definition. A quadruple $\langle val, is\_open, is\_make, cid \rangle$, where val is a real number, is_open and is_make are boolean values, and cid is a clause identifier.

Meaning

- val : make-break value.
- is_open : active or not at val point.
- is_make : make or break, increase or decrease score.
- cid : causing clause.

Sorting: First ordered by val, then by is_open $(\perp < \top)$.

Boundary

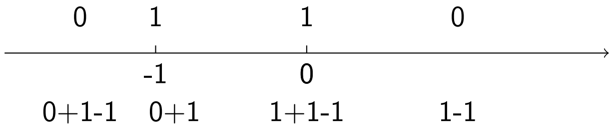
Current assignment: $\{x \mapsto 1, y \mapsto 1, z \mapsto 1\}$

- $x^2 + y^2 \leq 1$: starting score 0, boundary set $\{(0, \perp, \top, 1), (0, \top, \perp, 1)\}$, indicating no change for large negative values, *make* at boundary $[0, \cdots,$ followed by *break* at boundary $(0, \cdots.$
- $x + y < 1$: starting score 1, boundary set $\{(0, \perp, \perp, 1)\}$, indicating *make* at large negative values, and *break* at boundary $[0, \ldots.$
- $x + z > 0$: starting score -1, boundary set $\{(-1, \top, \top, 1)\}$, indicating *break* at large negative values, and *make* at boundary $(-1, \ldots.$

sorted boundary set:
$\{(-1, \top, \top, 1), (0, \perp, \top, 1), (0, \perp, \perp, 1), (0, \top, \perp, 1)\}$

Boundary Example

boundary set:
$\{(-1,\top,\top,1),(0,\bot,\top,1),(0,\bot,\bot,1),(0,\top,\bot,1)\}$



Starting score: Score when x moves to -∞.
Maintain and Change: We maintain the boundary info for all arithmetic varaibles, unless the neighbour does a critical move.

Algorithm for computing boundary

Input : Variable v that is modified
Output: Make-break score for all variables

$S \leftarrow \{\}$; `// set of updated variables`

for *clause* cls *that contains* v **do**
  for *variable* v' *appearing in* cls **do**
    add v' to S;
    recompute starting score and boundary of v' with respect to cls;
  end
end
for *variable* v' *in S* **do**
  recompute best critical move and score in terms of boundary information;
end

Outline

Complexity of Values

Definition

We define a preorder $\prec_c$ on algebraic numbers as follows. $x \prec_c y$ if x is rational and y is irrational, or if both x and y are rational numbers, and the denominator of x is less than that of y. We write $x \sim_c y$ if neither $x \prec_c y$ nor $y \prec_c x$.

Previous work ignores equalities constraints[3], or only consider multi-linear (one-degree)examples[4].
Our Solution: Introducing relaxation, temporary enlarge the point irrational interval

[3] Haokun Li, Bican Xia, and Tianqi Zhao. "Local Search for Solving Satisfiability of Polynomial Formulas". In: *Computer Aided Verification - 35th International Conference, CAV, 2023*. Ed. by Constantin Enea and Akash Lal. Vol. 13965. Lecture Notes in Computer Science. Springer, 2023, pp. 87–109.

[4] Bohan Li and Shaowei Cai. "Local Search For SMT On Linear and Multilinear Real Arithmetic". In: *CoRR* abs/2303.06676 (2023). arXiv: 2303.06676.

Relaxation

Example

Given assignment $\{x \mapsto 1, y \mapsto 1\}$

$$z^2 = x^2 + y^3 \qquad z^3 \geq 5x^2 + y \vee z^3 \leq 3x + 3y$$

Both situations force z to an irrational number.

Relaxation

- If the constraint is of the form $p = 0$, it is relaxed into the pair of inequalities $p < \epsilon_p$ and $p > -\epsilon_p$.
- If the constraint is of the form $p \geq 0$, it is relaxed into $p > -\epsilon_p$. Likewise, if the constraint is of the form $p \leq 0$, it is relaxed into $p < \epsilon_p$.
- **Slacked var:** the var that is being assigned.

Restore

Input : slacked clauses
Output: succeed or not
for *each slacked clause* cls **do**
  $v \leftarrow$ slacked variable in cls;
  $accu\_val \leftarrow inf\_set(cls)$;
  move v to $accu\_val$;
end
for *variable* v' *in slacked clauses* **do**
  recompute best critical move and score in terms of boundary information;
end
return number of unsat clauses $== 0$

Local Search with Relaxation

```
Input   : A set of clauses F
Output: An assignment of variables that satisfy F, or failure
Initialize assignment to variables;
while ⊤ do
    if all clauses satisfied then
        success ← find exact solution;
        if success then
            return success with assignment;
        end
        else
            Restore relaxed constraints to original form;
            success ← find exact solution by limited local search;
            if success then
                return success with assignment;
            end
        end
    end
    if time or step limit reached then
        return failure;
    end
    Proceed traditional local search (slack).
end
```

Outline

1. Problem - Nonlinear Real Arithmetic
   - Syntax of SMT(NRA)
   - Fragment of Local Search
2. Incremental Computation of Variable Scores
   - Scoring Boundary for Arithmetic Variable
   - Incremental Computation
3. Temporary Relaxation of Equality (Non-Strick) Constraints
   - Value Complexity in Local Search
   - Relaxation Method
4. Experiment

Implementation Detail

code available at:
https://github.com/yogurt-shadow/LS_NRA

Preprocessing

- Combine constraints $p \geq 0$ and $p \leq 0$ into equality $p = 0$.
- Eliminate variable x in an equation of the form $c \cdot x + q = 0$, where c is a constant and q is a polynomial with degree at most 1 and containing at most 2 variables.

Restart mechanism Two-level restart mechanism with two parameters $T_1 = 100$ and $T_2 = 100$.

- **Minor restart:** randomly change one of the variables in one of the unsatisfied clauses.
- **Major restart:** reset the value of all variables.

Overall Result

Category	#inst	Z3	cvc5	Yices	Ours	Unique
20161105-Sturm-MBO	120	0	0	0	**88**	88
20161105-Sturm-MGC	2	**2**	0	0	0	0
20170501-Heizmann	60	3	1	0	**8**	6
20180501-Economics-Mulligan	93	**93**	89	91	90	0
2019-ezsmt	61	**54**	51	52	19	0
20200911-Pine	237	**235**	201	**235**	224	0
20211101-Geogebra	112	**109**	91	99	101	0
20220314-Uncu	74	73	66	**74**	70	0
LassoRanker	351	155	**304**	122	272	13
UltimateAtomizer	48	**41**	34	39	27	2
hycomp	492	**311**	216	227	304	11
kissing	42	**33**	17	10	**33**	1
meti-tarski	4391	**4391**	4345	4369	4351	0
zankl	133	70	61	58	**100**	27
Total	6216	5570	5476	5376	**5687**	148

Scatter Plot

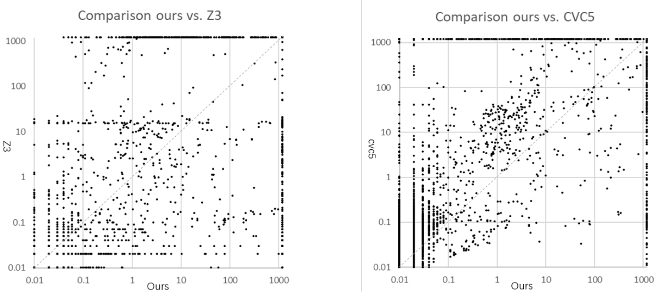


Figure: Scatter plots of running time vs. Z3 and cvc5.

Category	#inst	Incremental	Naive	Limit-45
20161105-Sturm-MBO	120	88	85	85
20161105-Sturm-MGC	2	0	0	0
20170501-Heizmann	60	8	5	5
20180501-Economics-Mulligan	93	90	89	89
2019-ezsmt	61	19	19	15
20200911-Pine	237	224	222	222
20211101-Geogebra	112	101	101	101
20220314-Uncu	74	70	70	70
LassoRanker	351	272	264	269
UltimateAtomizer	48	27	26	26
hycomp	492	304	298	298
kissing	42	33	32	33
meti-tarski	4391	4351	4352	4352
zankl	133	100	100	100
Total	6216	5687	5663	5665

Table: Comparison of incremental computation

Category	#inst	Relaxation	Threshold	NoOrder
20161105-Sturm-MBO	120	88	100	99
20161105-Sturm-MGC	2	0	0	0
20170501-Heizmann	60	8	9	3
20180501-Economics-Mulligan	93	90	89	86
2019-ezsmt	61	19	19	19
20200911-Pine	237	224	223	222
20211101-Geogebra	112	101	98	92
20220314-Uncu	74	70	70	70
LassoRanker	351	272	277	278
UltimateAtomizer	48	27	26	20
hycomp	492	304	211	164
kissing	42	33	31	27
meti-tarski	4391	4351	4353	4360
zankl	133	100	100	100
Total	6216	5687	5606	5540

Table: Comparison of temporary relaxation of constraints

Future Work

- Integrate into z3++ solver
  https://z3-plus-plus.github.io/
- Cacheing about cylindrical cells by CAD (we enter the same cell multiple times, how can we find that?)
- incorporate with other algorithms, like MCSAT or varaible substitution.
- used for nonlinear optimization

References I

[JM12] Dejan Jovanovic and Leonardo Mendonça de Moura. "Solving Non-linear Arithmetic". In: *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*. Ed. by Bernhard Gramlich, Dale Miller, and Uli Sattler. Vol. 7364. Lecture Notes in Computer Science. Springer, 2012, pp. 339–354. DOI: 10.1007/978-3-642-31365-3_27. URL: https://doi.org/10.1007/978-3-642-31365-3_27.

References II

[CLZ22] Shaowei Cai, Bohan Li, and Xindi Zhang. "Local Search for SMT on Linear Integer Arithmetic". In: *Computer Aided Verification - 34th International Conference, CAV*. Ed. by Sharon Shoham and Yakir Vizel. Springer, 2022.

[LC23] Bohan Li and Shaowei Cai. "Local Search For SMT On Linear and Multilinear Real Arithmetic". In: *CoRR* abs/2303.06676 (2023). arXiv: 2303.06676.

References III

[LXZ23] Haokun Li, Bican Xia, and Tianqi Zhao. "Local Search for Solving Satisfiability of Polynomial Formulas". In: *Computer Aided Verification - 35th International Conference, CAV, 2023*. Ed. by Constantin Enea and Akash Lal. Vol. 13965. Lecture Notes in Computer Science. Springer, 2023, pp. 87–109.