Incremental Linearization: A practical approach to Satisfiability Modulo Nonlinear Arithmetic and Transcendental Functions

(Invited Paper)

Alessandro Cimatti, Alberto Griggio, Ahmed Irfan, Marco Roveri
Fondazione Bruno Kessler (FBK), Italy
Email: {cimatti, griggio, irfan, roveri}@fbk.eu

Roberto Sebastiani
University of Trento, Italy
Email: roberto.sebastiani@unitn.it

***Abstract*—Satisfiability Modulo Theories (SMT) is the problem of deciding the satisfiability of a first-order formula with respect to some theory or combination of theories. In this paper, we overview our recent approach called Incremental Linearization, which successfully tackles the problems of SMT over the theories of nonlinear arithmetic over the reals (NRA), nonlinear arithmetic over the integers (NIA) and their combination, and of NRA augmented with transcendental (exponential and trigonometric) functions (NTA). Moreover, we showcase some of the experimental results and outline interesting future directions.**

***Keywords*-Automated Reasoning; Linear and Nonlinear Arithmetic; SMT; Transcendental Functions;**

I. Introduction

Satisfiability Modulo Theories (SMT) solvers are used as automated reasoning engines in various problem areas such as formal verification. Over the years, there has been a lot of progress on the efficient techniques for the quantifier-free theories of linear arithmetic and uninterpreted functions – many powerful and effective SMT solvers are available for such theories. Yet, many application domains require more expressive theories (e.g. aerospace, cyber-physical systems, and railways). Supporting nonlinear theories and finding their efficient solving methods, that work well in practice, in SMT solvers has become a fundamental research problem. Unfortunately, dealing with nonlinearity is a hard challenge. Going from linear to nonlinear arithmetic yields a complexity gap that results in a computational barrier in practice – in the case of the reals, most available complete solvers rely on Cylindrical Algebraic Decomposition (CAD) techniques [1], which require double exponential time in worst case; the problem becomes undecidable [2] (the result of Hilbert's 10th problem) in the case of the integers. Similarly, the addition of transcendental functions to the reals has been shown to be undecidable [3].

Recently, we have proposed a conceptually-simple yet effective practical approach for dealing with the quantifier-free theory of nonlinear arithmetic over the reals, over the reals extended with transcendental functions, and over integers, called *Incremental Linearization* [4], [5], [6], [7]. Its underlying idea is that of trading the use of expensive, exact solvers for nonlinear arithmetic for an abstraction-refinement loop on top of much less expensive solvers for linear arithmetic and uninterpreted functions. In this paper, we give an overview of the approach.

The paper is organized as follows. In II we provide some background. Then, we present high-level ideas of incremental linearization for SMT in III. In IV we mention related work and in V we present some highlights of the experimental results. Finally, in VI we draw some conclusions and outline future work directions.

II. Background

We assume the standard first-order quantifier-free logical setting and standard notions of theory, satisfiability, and logical consequence. We denote formulae with φ, ψ, terms with t, s, variables with x, y, constants with a, b, c, functions with f, TF, each possibly with subscripts. If X is a set of variables, we write $\varphi(X)$ to denote the fact that the all the variables of φ are in X. If μ is a model and x is a variable, we write $\mu[x]$ to denote the value of x in μ, and we extend this notation to terms and formulae in the usual way. If Γ is a set of formulae, we write $\bigwedge \Gamma$ (or simply Γ) to denote the conjunction of all the formulae in Γ. We abuse the notation and write $t \in \varphi$ to denote that term t occurs in φ. $abs(t)$ stands for $\text{ITE}(t < 0, -t, t)$, ITE being the standard if-then-else term operator. We denote with $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$ the set of integer, rational and real numbers, respectively.

Satisfiability Modulo Theories: Satisfiability Modulo Theories (SMT) is the problem of deciding the satisfiability of a first-order formula with respect to some first-order theory ($\mathcal{T}$) or combination of first-order theories ($\mathcal{T}_1 \cup \mathcal{T}_2$). A formula is satisfiable in $\mathcal{T}$ (or $\mathcal{T}$-satisfiable) if it is satisfiable in a model of $\mathcal{T}$ (also written as $\mathcal{T}$-model). An SMT solver is a decision procedure which solves the SMT problem. There exist several theories that the modern SMT solvers support. In this work we are interested in the following theories: *Equality and Uninterpreted Functions*, *Linear Arithmetic* and *Nonlinear Arithmetic* over the reals and over the integers, and in their combinations thereof. The theory of linear real arithmetic (LRA) [resp. LIA] is the first-order theory with equality whose atoms are linear polynomial constraints interpreted over $\mathbb{R}$, whereas the theory of nonlinear real arithmetic (NRA) [resp. NIA

] is the first-order theory with equality whose atoms are nonlinear polynomial constraints interpreted over $\mathbb{R}$ [resp. $\mathbb{Z}$]. We denote with UFLRA the combined theory of UF and LRA, and UFLIA the combined theory of UF and LIA.

Functions over the reals: We assume that we have continuous and differentiable functions. If f is a univariate function, we write $\frac{d}{dx}f$ for the first-order derivative of f. We also write $f^{(i)}$ for the i-th derivative of f, and f' and f'' for $f^{(1)}$ for $f^{(2)}$, respectively. Let f be a univariate function twice differentiable at point c. The *concavity* of f at c is the sign of $f''(c)$. Let $f(x,y)$ be a bivariate function. We write $\frac{d}{dx}f(x,y)$ and $\frac{d}{dy}f(x,y)$ for the first-order partial derivatives of $f(x,y)$ w.r.t. x and y, respectively. The *tangent plane* at (a,b) to a bivariate function $f(x,y)$, denoted with $\textsc{TanPlane}_{f,a,b}(x,y)$, is defined as follows: $\textsc{TanPlane}_{f,a,b}(x,y) \doteq f(a,b) + \frac{d}{dx}f(a,b) * (x-a) + \frac{d}{dy}f(a,b) * (y-b)$.

Taylor Series and Taylor's Theorem: Let $f(x)$ be n-differentiable at a. The *Taylor series* of f of degree n centered around a is the polynomial: $P_{n,f,a}(x) \doteq \sum_{i=0}^{n} \frac{f^{(i)}(a)}{i!} * (x-a)^i$. The Taylor series centered around zero is also called *Maclaurin series*. According to *Taylor's theorem*, any continuous function $f(x)$ that is $(n+1)$-differentiable can be written as the sum of the Taylor series and the remainder term: $f(x) = P_{n,f,a}(x) + R_{n+1,f,a}(x)$, where $R_{n+1,f,a}(x)$ is the Lagrange form of the remainder. An upper bound on the size of the remainder $R^{\mathrm{U}}_{n+1,f,a}(x)$ at a point x can be defined as: $\max_{c\in[\min(a,x),\max(a,x)]}(|f^{(n+1)}(c)|) * \frac{|(x-a)^{n+1}|}{(n+1)!}$. From this, we obtain a lower- and an upper-bound for $f(x)$, given by $P_{n,f,a}(x) - R^{\mathrm{U}}_{n+1,f,a}(x)$ and $P_{n,f,a}(x) + R^{\mathrm{U}}_{n+1,f,a}(x)$ respectively. Clearly, the closer is a to x, the tighter the approximation of $f(x)$ will be.

III. SMT via Incremental Linearization

We first give the intuition by an example.

Example: Consider the following constraints: $x*x + y*y \leq 2 \wedge (x \geq 1.1 \vee x \leq -1.1) \wedge (y \geq 1.1 \vee y \leq -1.1)$, which are graphically shown in Figure 1a. We want to check whether their intersection is nonempty or not. One way is to use some nonlinear solving method to answer that question. However, notice that the intersection can be shown empty by approximating the circle: this can be done by replacing nonlinear multiplications, i.e., $x*x$ and $y*y$ with uninterpreted functions $f_*(x,x)$ and $f_*(y,y)$, respectively, and adding the following linear constraints over the uninterpreted functions $(f_*(x,x) \geq -2.8*x - 1.96) \wedge (f_*(x,x) \geq -3*x - 2.25) \wedge (f_*(x,x) \geq 3.2*x - 2.56) \wedge (f_*(x,x) \geq 2.6*x - 1.69) \wedge (f_*(y,y) \geq 2.4*y - 1.44) \wedge (f_*(y,y) \geq -2.8*y - 1.96) \wedge (f_*(y,y) \geq 2.2*y - 1.21) \wedge (f_*(y,y) \geq -3*y - 2.25)$. Clearly, as depicted in Figure 1b the additional constraints approximate the circle. Therefore, we can answer the question by solving the linear problem (as shown in Figure 1c) using some linear method.

We now provide a high-level description of the algorithm for SMT solving on NTA (and hence NRA) based on incremental linearization, also shown in Figure 2. (We obtain the procedure for SMT(NIA) by replacing the underlying solver with SMT(UFLIA).) To simplify the presentation, and when not explicitly stated otherwise, we often implicitly assume w.l.o.g. that all multiplications in the input formula φ are either between variables (e.g. $x*y$) or between one constant and one variable (e.g. $3x$), and that all transcendental functions in φ are applied to variables (e.g., $\exp(x)$). Moreover, we consider $\exp$, $\sin$ and π transcendental functions only. Other transcendental functions such as $\log$, $\cos$, $\tan$, $\arcsin$, $\arccos$, $\arctan$ can be handled by means of rewriting.

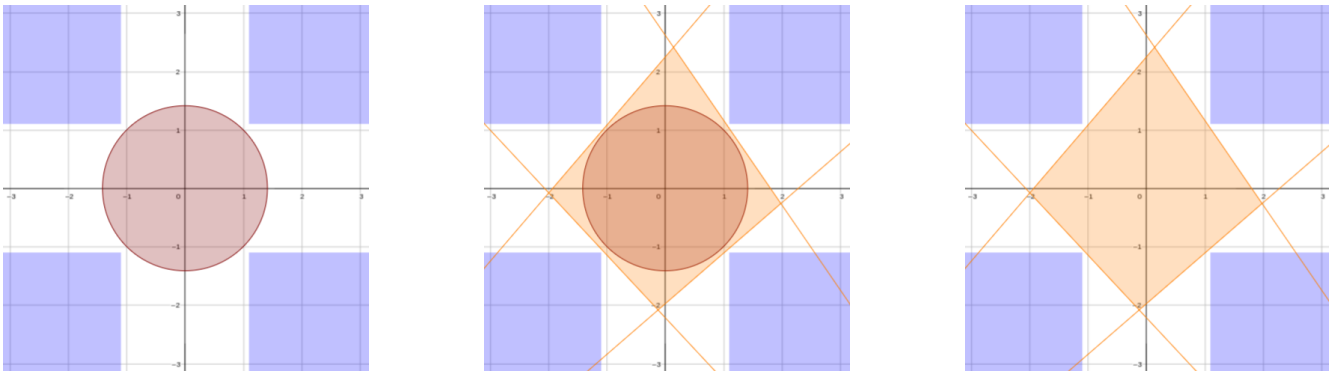
A. High-level Description

The algorithm takes as input a formula φ containing nonlinear constraints with polynomials and transcendental functions. Then the formula φ is abstracted into an over-approximating formula $\widehat{\varphi}$ over the combined theory of linear arithmetic and uninterpreted functions (UFLRA). $\widehat{\varphi}$ is the result of replacing each nonlinear term $x*y$ with $f_*(x,y)$, and each transcendental term $\mathrm{TF}(x)$ with $f_{\mathrm{TF}}(x)$, s.t. $f_*(.)$ and $f_{\mathrm{TF}}(.,.)$ are uninterpreted functions, and the symbol π with the new symbol $\widehat{\pi}$. (We remark that, linear multiplications, like e.g. $c*x$ where c is a constant, are not replaced.) Then the algorithm enters a loop. At each iteration, the approximation $\widehat{\varphi} \wedge \bigwedge \Gamma$ (Γ is empty initially) of φ is solved by an SMT(UFLRA) solver. If the solver returns SAT then the algorithm tries to lift it to a satisfiability result for the original formula φ, and in the case of an unsuccessful attempt it refines the approximation by adding new UFLRA constraints to Γ that rule out spurious solutions. The process iterates until either the formula $\widehat{\varphi} \wedge \bigwedge \Gamma$ is proved unsatisfiable in SMT(UFLRA), or the UFLRA-satisfiable result is lifted to a satisfiability result for the original formula.

B. Spuriousness Check and Abstraction Refinement

The lifting of UFLRA-satisfiability result is formulated as a SMT(UFLRA) search problem over a constrained version of φ, guided by the current abstract model, either by yielding a sufficient criterion for concluding the existence of a model for φ, returning that $\widehat{\mu}$ is not spurious, or by concluding $\widehat{\mu}$ is spurious. In the later case, $\widehat{\mu}$ violates some multiplications, or some transcendental functions, or both, and the refinement procedure tries to refine the spurious model $\widehat{\mu}$ by adding UFLRA refinement constraints that rule out $\widehat{\mu}$ (and other spurious solutions). This is performed in two steps: NRA refinement and NTA refinement.

1) NRA Spuriousness Check and Refinement: To determine whether $\widehat{\mu}$ implies the existence of a NRA-model for φ, one could simply check if $\widehat{\mu}[x] * \widehat{\mu}[y] = \widehat{\mu}[f_*(x,y)]$ for every multiplication term $f_*(x,y) \in \widehat{\varphi}$. However, it is easy to see that this very simple check succeeds only if the



(a) Is the intersection of the circle and squares nonempty?

(b) Linear approximation of circle.

(c) Linear constraints for solving the problem.

Figure 1. Linearization Example

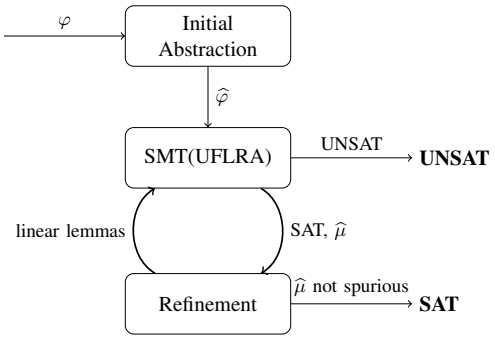


Figure 2. SMT(NTA) via Incremental Linearization

UFLRA solver "guesses" a model that is consistent with all the nonlinear multiplications. In an infinite and dense domain like the rationals or the reals, the chances that this will happen are close to zero in general. In order to detect satisfiable cases more effectively in the very-likely case in which $\widehat{\mu}$ is spurious, we also want to search for the existence of an actual model for φ "in the surroundings" of $\widehat{\mu}$. Our idea is to extract the truth assignment $\widehat{\psi}$ induced by $\widehat{\mu}$ on the atoms of $\widehat{\varphi}$:

$$\widehat{\psi} \doteq \bigwedge_{[\widehat{A} \in \text{atoms}(\widehat{\varphi}) \ s.t. \widehat{\mu} \models \widehat{A}]} \widehat{A} \ \wedge \bigwedge_{[\widehat{A} \in \text{atoms}(\widehat{\varphi}) \ s.t. \widehat{\mu} \not\models \widehat{A}]} \neg \widehat{A},$$

and then to look for another model $\widehat{\eta}$ for $\widehat{\psi}$. Notice that, any such model $\widehat{\eta}$ shares with $\widehat{\mu}$ the truth assignment on the atoms $\widehat{\psi}$, but with different values of the real variables. Then we conjoin to the truth assignment the *multiplication-line constraints*:

$$\widehat{\psi}^* \doteq \widehat{\psi} \wedge \bigwedge_{f_*(x,y) \in \widehat{\psi}} \left(\begin{array}{l} (x = \widehat{\mu}[x] \wedge f_*(x,y) = \widehat{\mu}[x] * y) \vee \\ (y = \widehat{\mu}[y] \wedge f_*(x,y) = \widehat{\mu}[y] * x) \end{array} \right).$$

The main idea is to build an UFLRA underapproximation $\widehat{\psi}^*$ of the NRA formula ψ, in which all multiplications are forced to be linear. This idea is demonstrated by the following example.

Example: Consider the following formula $\varphi \doteq x * y = 10 \wedge (2 \leq x \leq 4) \wedge (2 \leq y \leq 4)$. Its abstraction is given by $\widehat{\varphi} \doteq f_*(x,y) = 10 \wedge (2 \leq x \leq 4) \wedge (2 \leq y \leq 4)$. Suppose the following model $\widehat{\mu}$ is returned by the SMT(UFLRA) solver: $\widehat{\mu}[x] = 2$, $\widehat{\mu}[y] = 4$, $\widehat{\mu}[f_*(x,y)] = 10$. $\widehat{\mu}$ is a spurious interpretation because $2 * 4 \neq 10$ in NRA. However, using the search for NRA-model described earlier, we can still find an NRA-compliant model from the "guesses", which solves the following UFLRA-satisfiable formula:

$$\begin{aligned} \widehat{\psi}^* \doteq \ & f_*(x,y) = 10 \wedge (2 \leq x \leq 4) \wedge (2 \leq y \leq 4) \wedge \\ & ((x = 2 \wedge f_*(x,y) = 2 * y) \vee \\ & (y = 4 \wedge f_*(x,y) = 4 * x)). \end{aligned}$$

A possible UFLRA-model $\widehat{\mu}^*$ for $\widehat{\varphi}$ is $\widehat{\mu}^*[x] = \frac{5}{2}$, $\widehat{\mu}^*[y] = 4$, $\widehat{\mu}^*[f_*(x,y)] = 10$, that is also compliant with NRA.

If the search for NRA-model fails, then we looks for UFLRA constraints on multiplication terms in the form $f_*(x,y)$ occurring in $\widehat{\varphi}$ which are violated by $\widehat{\mu}$. We refine the abstraction by selecting suitable instantiations of given constraint schemata which prevent spurious assignments to multiplication terms. We consider the refinement constraint schemata in Figure 3 (more details can be found in [7]), where x, x_i, y, y_i are variables and a, b are generic rational values.

Example: Consider the case where φ contains the multiplications $u_1 * w_1$ and $u_2 * w_2$, so that $\widehat{\varphi}$ contains the multiplication terms $f_*(u_1, w_1)$ and $f_*(u_2, w_2)$. Let $\widehat{\mu}$ be a spurious assignment s.t. $\widehat{\mu}[u_1] = 2$, $\widehat{\mu}[w_1] = 3$, $\widehat{\mu}[f_*(u_1, w_1)] = 7$, $\widehat{\mu}[u_2] = 3$, $\widehat{\mu}[w_2] = -4$, $\widehat{\mu}[f_*(u_2, w_2)] = 5$. $\widehat{\mu}$ violates the third Zero constraint on $f_*(u_2, w_2)$, and it violates the three Monotonicity constraints on $\langle f_*(u_1, w_1), f_*(u_2, w_2) \rangle$. Overall, this leads to the addition of the Zero and Mono-

$$
\begin{array}{rl}
\textit{Zero:} & \forall x,y.((x=0 \vee y=0) \leftrightarrow f_*(x,y)=0) \\
& \forall x,y.(((x>0 \wedge y>0) \vee (x<0 \wedge y<0)) \leftrightarrow f_*(x,y)>0) \\
& \forall x,y.(((x<0 \wedge y>0) \vee (x>0 \wedge y<0)) \leftrightarrow f_*(x,y)<0) \\
\textit{Monotonicity:} & \forall x_1,y_1,x_2,y_2.((abs(x_1)\leq abs(x_2) \wedge abs(y_1)\leq abs(y_2)) \rightarrow abs(f_*(x_1,y_1)) \leq abs(f_*(x_2,y_2))) \\
& \forall x_1,y_1,x_2,y_2.((abs(x_1)< abs(x_2) \wedge abs(y_1)\leq abs(y_2) \wedge y_2 \neq 0) \rightarrow abs(f_*(x_1,y_1)) < abs(f_*(x_2,y_2))) \\
& \forall x_1,y_1,x_2,y_2.((abs(x_1)\leq abs(x_2) \wedge abs(y_1)< abs(y_2) \wedge x_2 \neq 0) \rightarrow abs(f_*(x_1,y_1)) < abs(f_*(x_2,y_2))) \\
\textit{Tangent plane:} & \forall x,y.(f_*(a,y)=a*y \wedge f_*(x,b)=b*x \wedge \\
& \quad (((x>a \wedge y<b) \vee (x<a \wedge y>b)) \rightarrow f_*(x,y) < \textsc{TanPlane}_{*,a,b}(x,y)) \wedge \\
& \quad (((x<a \wedge y<b) \vee (x>a \wedge y>b)) \rightarrow f_*(x,y) > \textsc{TanPlane}_{*,a,b}(x,y)))
\end{array}
$$

Figure 3. The refinement UFLRA constraint schemata for multiplication.

$$
\begin{array}{rl}
\textit{Zero:} & ((u_2<0 \wedge w_2>0) \vee (u_2>0 \wedge w_2<0)) \leftrightarrow f_*(u_2,w_2)<0 \\
\textit{Monotonicity:} & (abs(u_1)\leq abs(u_2) \wedge abs(w_1)\leq abs(w_2)) \rightarrow abs(f_*(u_1,w_1)) \leq abs(f_*(u_2,w_2)) \\
& (abs(u_1)< abs(u_2) \wedge abs(w_1)\leq abs(w_2) \wedge w_2 \neq 0) \rightarrow abs(f_*(u_1,w_1)) < abs(f_*(u_2,w_2)) \\
& (abs(u_1)\leq abs(u_2) \wedge abs(w_1)< abs(w_2) \wedge u_2 \neq 0) \rightarrow abs(f_*(u_1,w_1)) < abs(f_*(u_2,w_2)) \\
\textit{Tangent plane:} & f_*(2,w_1)=2*w_1 \\
& f_*(u_1,3)=3*u_1 \\
& ((u_1>2 \wedge w_1<3) \vee (u_1<2 \wedge w_1>3)) \rightarrow f_*(u_1,w_1) < 3*u_1+2*w_1-6 \\
& ((u_1<2 \wedge w_1<3) \vee (u_1>2 \wedge w_1>3)) \rightarrow f_*(u_1,w_1) > 3*u_1+2*w_1-6 \\
& f_*(3,w_2)=3*w_2 \\
& f_*(u_2,-4)=-4*u_2 \\
& ((u_2>3 \wedge w_2<-4) \vee (u_2<3 \wedge w_2>-4)) \rightarrow f_*(u_2,w_2) < -4*u_2+3*w_2+12 \\
& ((u_2<3 \wedge w_2<-4) \vee (u_2>3 \wedge w_2>-4)) \rightarrow f_*(u_2,w_2) > -4*u_2+3*w_2+12
\end{array}
$$

Figure 4. Top: example of instantiation of constraint schemata for the multiplication terms $f_*(u_1,w_1)$ and $f_*(u_2,w_2)$, where $\widehat{\mu}[u_1]=2,\ \widehat{\mu}[w_1]=3,\ \widehat{\mu}[f_*(u_1,w_1)]=7,\ \widehat{\mu}[u_2]=3,\ \widehat{\mu}[w_2]=-4,\ \widehat{\mu}[f_*(u_2,w_2)]=5$.

tonicity constraints, plus the Tangent-plane ones in the points $(2,3)$ and $(3,-4)$, which are reported at the top of Figure 4.

2) NTA Spuriousness Check and Refinement: The process is performed for progressively-improving precision. At each iteration, first we looks for UFLRA constraints on transcendental terms in the form $f_{\mathrm{TF}}(x)$ occurring in $\widehat{\varphi}$ which are violated by $\widehat{\mu}$. These constraints (if any) are added to Γ. If Γ contains at least one refinement constraint ruling out $\widehat{\mu}$, then the refinement process is completed. If not so, then no result in either direction was obtained with the current precision. Then, the current precision is increased and spuriousness check is invoked again with the improved precision and search for linear lemmas is performed. The whole process is iterated until either φ is found satisfiable, or some refinement constraint is produced (or the process is terminated due to resource-budget exhaustion). The NTA-consistency check of the SMT(UFLRA)-model $\widehat{\mu}$ amounts to checking if $\widehat{\mu}[f_{\mathrm{TF}}(x)] = \mathrm{TF}(\widehat{\mu}[x])$. In practice, the check cannot be implemented, since transcendental functions at rational points most often have irrational values (see e.g. [8]), which cannot be represented in SMT(UFLRA). Therefore, for each term $\mathrm{TF}(x)$ in φ, we instead compute two rational values, namely QL and QU, with the property that $\mathrm{QL} \leq \mathrm{TF}(\widehat{\mu}[x]) \leq \mathrm{QU}$. The computation of QL and QU is based on polynomials computed using Taylor series, according to the given current precision. This is done by expanding the Maclaurin series of TF, generating polynomials for the upper and lower bounds according to Taylor's theorem, until the requested precision is met. The two values QL and QU are simply the results of evaluating the two computed polynomials $P_l(x)$ and $P_u(x)$ at $\widehat{\mu}[x]$. As an additional requirement that will be explained below, the function also ensures that the concavity of the Taylor polynomials is the same as that of TF at $\widehat{\mu}[x]$.

If the value of $\mathrm{TF}(x)$ in $\widehat{\mu}$ is not included in the interval $[\mathrm{QL}, \mathrm{QU}]$, we generate (piecewise) linear constraints that remove the point $(\widehat{\mu}[x], \widehat{\mu}[\mathrm{TF}(x)])$ (and possibly many others) from the graph of f_{TF}, thus refining the abstraction. First, we attempts to exclude the bad point by invoking basic lemmas, which instantiates some *basic constraint schemata* describing very general properties of the transcendental function TF under consideration (see [6] for details). These constraints encode some simple properties of transcendental functions (such as sign and monotonicity conditions, or bounds at noteworthy values) via linear relations. If none of the basic constraints is violated, then two situations are possible, as illustrated in Figure 5. Let the green line be

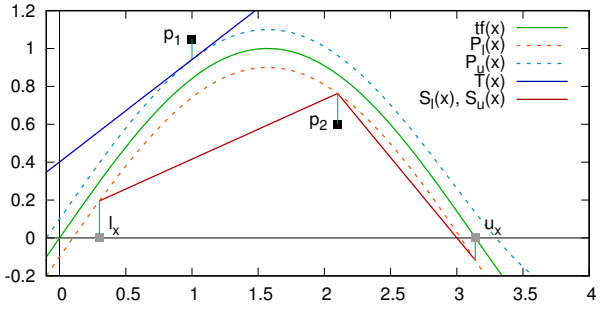


Figure 5. Piecewise-linear refinement illustration.

the graph of some transcendental function TF. The points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$, say (1.0,1.0) and (2.2, 0.5), represent transcendental terms that are spurious in the current assignment $\widehat{\mu}$ —that is, $p_1 = (\widehat{\mu}[x_1], \widehat{\mu}[f_{\text{TF}}(x_1)])$ and $p_2 = (\widehat{\mu}[x_2], \widehat{\mu}[f_{\text{TF}}(x_2)])$, for some x_1, x_2. In order to eliminate them, we need to discover linear constraints that are guaranteed to safely approximate TF. Clearly, a major role is played by the position of the spurious value $\widehat{\mu}[f_{\text{TF}}(x)]$ relative to the correct value $\text{TF}(\widehat{\mu}[x])$, and by the concavity of TF around the point $\widehat{\mu}[x]$. If the concavity is negative or equal to zero, and the point lies above the function, then the tangent to the function would be adequate to block the spurious assignment –and tighten the approximation of TF. (This is the case for p_1.) However, if the concavity is negative but the point lies below the function, then a tangent would be not adequate. (This is the case for p_2.) For this reason, secants are required, which are unfortunately not unique.

The Exponential Function: Since $\frac{d}{dx}\exp(x) = \exp(x)$, all the derivatives of exp are positive. The polynomial $P_{n,\exp,0}(x)$ is given by the Maclaurin series $P_{n,\exp,0}(x) = \sum_{i=0}^{n} \frac{x^i}{i!}$ and behaves differently depending on the sign of x. Thus, we distinguish three cases for finding the polynomials $P_l(x)$ and $P_u(x)$:

- Case $x = 0$: since $\exp(0) = 1$, we have $P_l(0) = P_u(0) = 1$;
- Case $x < 0$: we have that $P_{n,\exp,0}(x) < \exp(x)$ if n is odd, and $P_{n,\exp,0}(x) > \exp(x)$ if n is even; we therefore set $P_l(x) = P_{n,\exp,0}(x)$ and $P_u(x) = P_{n+1,\exp,0}(x)$ for a suitable n so that the required precision ϵ is met;
- Case $x > 0$: we have that $P_{n,\exp,0}(x) < \exp(x)$ and $P_{n,\exp,0}(x) * (1 - \frac{x^{n+1}}{(n+1)!})^{-1} > \exp(x)$ when $(1 - \frac{x^{n+1}}{(n+1)!}) > 0$, therefore we set $P_l(x) = P_{n,\exp,0}(x)$ and $P_u(x) = P_{n,\exp,0}(x) * (1 - \frac{x^{n+1}}{(n+1)!})^{-1}$ for a suitable n.

Since the concavity of exp is always positive, the tangent refinement will always give lower bounds for $\exp(x)$, and the secant refinement will give upper bounds.

The Sine Function: The correctness of our refinement procedure relies crucially on being able to compute the concavity of the transcendental function TF at a given point c. This is needed in order to know whether a computed tangent or secant line constitutes a valid upper or lower bound for TF around c (see Figure 5). In the case of the sin function, computing the concavity at an arbitrary point c is problematic, since this essentially amounts to computing the value $c' \in [-\pi, \pi)$ s.t. $c = 2\pi n + c'$ for some integer n, because in $[-\pi, \pi)$ the concavity of $\sin(c')$ is the opposite of the sign of c'. This is not easy to compute because π is a transcendental number. In order to solve this problem, we exploit another property of sin, namely its periodicity (with period 2π). More precisely, we split the reasoning about sin depending on two kinds of periods: base period, with argument from $-\pi$ to π, and extended period. For each $\sin(x)$ term we introduce an "artificial" sin term $\sin(\omega_x)$, where ω_x is a fresh variable called *base variable*. Base variables are constrained to be interpreted over the base period, where the sin value for the corresponding variable in the extended period is computed. This is done by adding the following constraint during formula preprocessing:

$$\bigwedge_{\sin(x)\in\varphi} \left(\begin{array}{l} -\pi \leq \omega_x < \pi \ \wedge \ \sin(x) = \sin(\omega_x)) \wedge \\ ((-\pi \leq x < \pi) \rightarrow x = \omega_x) \end{array} \right).$$

The first conjunct constrains ω_x to the base period. The second conjunct constrains $\sin(x)$ to have the same value as $\sin(\omega_x)$. The third conjunct states that if x is interpreted in the base period then it has the same value as its base variable. In order to reason about the irrational π, we introduce a variable $\widehat{\pi}$, and add the constraint $l_\pi < \widehat{\pi} < u_\pi$ to φ. l_π and u_π are valid rational lower and upper bounds for the actual value of π that can be computed with various methods. Then, for each term $f_{\sin}(\omega_x)$ that needs to be refined, we first check whether $\widehat{\mu}[\omega_x] \in [-l_\pi, l_\pi]$, where l_π is the current lower bound for $\widehat{\pi}$. If this is the case, then we derive the concavity of sin at $\widehat{\mu}[\omega_x]$ by just looking at the sign of $\widehat{\mu}[\omega_x]$. We can therefore perform tangent or secant refinement. More precisely, we finds the lower and upper polynomials using Taylor's theorem, which ensures that $P_{n,\sin,0}(\omega_x) - R^{\text{U}}_{n+1,\sin,0}(\omega_x) \leq \sin(\omega_x) \leq P_{n,\sin,0}(\omega_x) + R^{\text{U}}_{n+1,\sin,0}(\omega_x)$ where $P_{n,\sin,0}(\omega_x) = \sum_{k=0}^{n} \frac{(-1)^k * \omega_x^{2k+1}}{(2k+1)!}$ and $R^{\text{U}}_{n+1,\sin,0}(\omega_x) = \frac{\omega_x^{2(n+1)}}{(2(n+1))!}$. We set $P_l(x) = P_{n,\sin,0}(x) - R^{\text{U}}_{n+1,\sin,0}(x)$ and $P_u(x) = P_{n,\sin,0}(x) + R^{\text{U}}_{n+1,\sin,0}(x)$. Under the above hypothesis that $\widehat{\mu}[\omega_x] \in [-l_\pi, l_\pi]$, if $\widehat{\mu}[\omega_x] \geq 0$, then the validity interval is $[0, \widehat{\pi})$, otherwise, it is $[-\widehat{\pi}, 0]$.

The remaining case to discuss is when the value of ω_x in $\widehat{\mu}$ is not within the interval $[-l_\pi, l_\pi]$. In this case, we cannot reliably compute the concavity of sin at $\widehat{\mu}[\omega_x]$. Therefore, instead of performing a tangent/secant refinement, we refine the precision of $\widehat{\pi}$ by computing a tighter interval (l'_π, u'_π) for it, using Machin's formula [9].

Example: In order to rule out a spurious interpretation $\widehat{\mu}[x] = 2.0, \widehat{\mu}[f_{\exp}(x)] = 3.0$ (where $f_{\exp}(x)$ is the abstraction of the exponential function) we may exploit the positive concavity of $\exp(x)$ and obtain a linear lower-bound constraint, e.g. $f_{\exp}(x) > \frac{155}{21} + \frac{331}{45} * (x - 2)$. Notice that, $\exp(2.0) \approxeq 7.389$, $\frac{155}{21} \approxeq 7.381 \lessapprox \exp(2.0)$, and $\frac{331}{45} \approxeq 7,356 \lessapprox \frac{d}{dx} exp(2.0)$. These values are such that the above linear constraint "approximates" the tangent of $\exp(x)$ in $x = 2$, since it always lower-bounds $\exp(x)$ and its value and derivative are very near to those of $\exp(x)$ for $x = 2.0$.

Remark: It is important to notice that we describe the refinement strategy which is currently implemented, which is only one of the many alternative strategies by which refinement can be performed. Moreover, we skipped the method to conclude the existence of an NTA-model, the details can be found in [6].

IV. Related Work

Various techniques have been explored for nonlinear SMT solving, that include: quantifier elimination methods like *cylindrical algebraic decomposition* (CAD) [1] and *virtual substitution* (VS) [10] for NRA (adopted by the solvers: SMT-RAT [11], Z3 [12] and YICES [13]); methods based on *interval constraint propagation* (ICP) [14] (implemented in the RASAT [15] solver for NRA, and ISAT3 [16] and DREAL [17] for NTA); linearization techniques [18], [19], [20], [21] for NRA and NIA (implemented in CVC4 [22]); and *bit-blasting* approaches [23], [24] (Z3 and SMT-RAT) and a combination of SMT(NRA) solving techniques with branch and bound [25] and [26] (YICES and SMT-RAT) for NIA. Moreover, in the context of theorem proving, two remarkable deductive methods are described in [27] (METITARSKI [27]) and in [28] for NTA.

V. Experimental Evaluation

The incremental linearization procedure has been implemented within the MATHSAT SMT solver [29]. The GMP arbitrary-precision library is used to represent rational numbers. For the evaluation, we compared MATHSAT with various SMT solvers. The experiments were run on a cluster of identical machines equipped with 2.6GHz Intel Xeon X5650 processors. The memory limit was set to 6 GB and 1000 seconds for the time limit. We present the results using survival plots and briefly discuss them. (For more detailed results and discussion, we refer to [30].) In the plots, by VIRTUALBEST we mean the results of a virtual portfolio solver that performs on each benchmark as the best of the solvers in the portfolio.

A. Other Approaches

For SMT(NRA), we considered Z3, YICES, and SMT-RAT, which implement expensive techniques based on variants of CAD, and ISAT3 and DREAL, based on ICP. For SMT(NIA), we considered Z3 and SMT-RAT which are based on the bit-blasting approach, YICES which uses CAD together with the branch-and-bound technique. We also considered the recent version of CVC4 for SMT(NRA) and SMT(NIA) that, as discussed in [20], is an independent implementation of our incremental linearization approach. For SMT(NTA), we considered ISAT3, DREAL and METITARSKI, that implements a deductive approach.

B. Benchmarks

For the experimental evaluation on SMT we selected the following benchmarks. For NRA and NIA, we used all the SMT-LIB [31] benchmarks from the QF-NRA and QF-NIA categories, respectively. The QF-NRA class contains 11354 benchmarks the QF-NIA category contains 23876 benchmarks. Since SMT(NTA) is not standardized in SMT-LIB, for NTA we adopted an extended version of SMT-LIB including special function symbols for $\sin$, $\exp$ and π. We collected and encoded SMT(NTA) benchmarks from various sources, for a total of 2512 benchmarks. ISAT3 requires that each variable is constrained to some interval. Therefore, we used a very large bound (allowed by ISAT3) for each variable in the NRA benchmarks. However, it also requires the intervals to be small when there are transcendental functions. In this case we generated scaled-down versions of the NTA benchmarks, by adding bound constraints that force all the real variables in the problem to assume values in the $[-300, 300]$ interval.

C. Results

1) SMT(NRA): The results are shown in Figure 6. MATHSAT (and also CVC4, that basically implements the same technique) demonstrates very good performance. Overall, MATHSAT solves 8852 benchmarks (behind Z3 with 9922 and YICES 9874). Interestingly, incremental linearization is highly complementary with respect to the more expensive techniques implemented in Z3, YICES and SMT-RAT. MATHSAT is able to solve 317 benchmarks that cannot be solved by other solvers (with the exception of CVC4).

2) SMT(NIA): The results are shown in Figure 7. MATHSAT solves the highest number of satisfiable and unsatisfiable instances. It solves a total of 16716 problems; the closest to MATHSAT is YICES which solves 15785 problems. Z3, SMT-RAT, and CVC4 are able to solve less than 10000 benchmarks. Notice that YICES implements a combination of expensive NRA quantifier elimination (CAD) and branch-and-bound technique, while Z3 and SMT-RAT rely on bit-blasting, and CVC4 is using a variant of incremental linearization. The difference between MATHSAT and VIRTUALBEST is around 2000 instances; that suggests some complementarity among the approaches.

3) SMT(NTA): The results are shown in Figure 8 and Figure 9. We can see that MATHSAT is able to solve more benchmarks than DREAL and ISAT3. METITARSKI

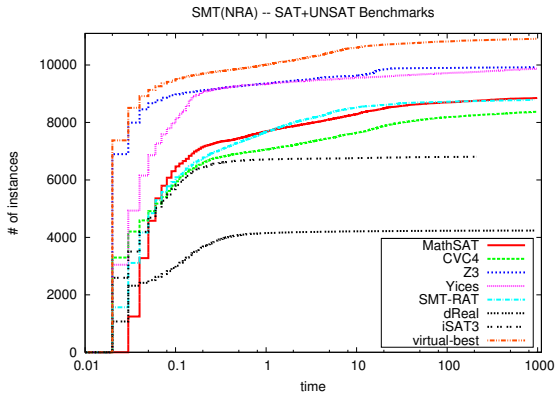


Figure 6. Survival plots for SMT(NRA) benchmarks

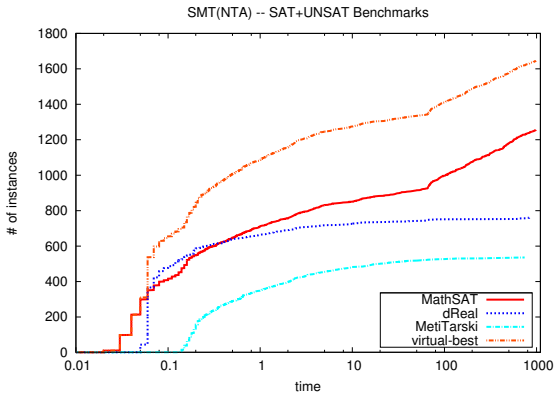


Figure 8. Survival plots for SMT(NTA) – unbounded benchmarks

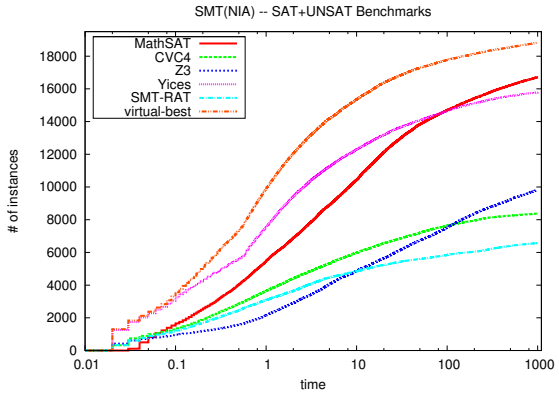


Figure 7. Survival plots for SMT(NIA) benchmarks

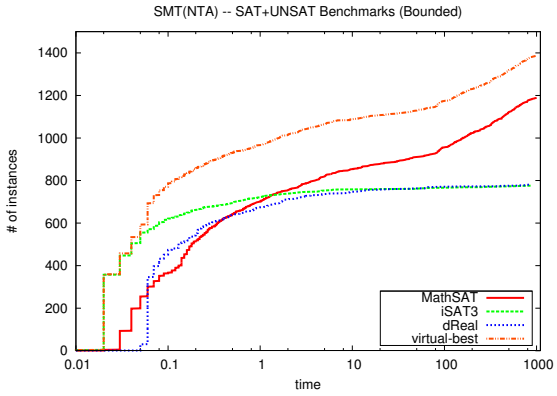


Figure 9. Survival plots for SMT(NTA) – bounded benchmarks

is unable to deal with benchmarks involving Boolean combinations, and thus could not be over all the benchmarks. Similarly to the case of SMT(NRA), DREAL is able to solve only unsatisfiable benchmarks, and returns MAYBESAT in many situations that are in fact unsatisfiable. If we consider the scaled-down case, ISAT3 demonstrates better performance than DREAL, being able to prove more satisfiable benchmarks. Overall, however, it is still behind MATHSAT.

VI. CONCLUSION

We have overviewed *incremental linearization*, which we have proposed as a general framework for automated reasoning about nonlinear polynomials and transcendental functions such as exponentiation and trigonometric functions. The approach has been implemented inside the MATHSAT SMT solver. The experimental results show the merits of incremental linearization. The technique is surprisingly effective, even compared to other complete (when available) and more mature approaches. Moreover, in [4], [6], we have successfully applied it to the verification of invariant properties of infinite transition systems with nonlinear constraints (the approach has been implemented within the NUXMV model checker [32]).

This work opens a number of research directions. Incremental linearization is clearly an incomplete technique in general for nonlinear problems, nevertheless, it would be interesting to identify subclasses of nonlinear problems for which incremental linearization is (theoretically) complete. To further improve efficiency, we would like to investigate the integration of incremental linearization with complementary techniques, such as ICP or CAD. Since most state-of-the-art SMT solvers for UF and LA can compute interpolants [33], extending incremental linearization to compute interpolants would be an exciting direction. Another application of incremental linearization would be in the case of Optimization Modulo Theories (OMT) [34] problems w.r.t. NRA, NTA, and NIA.

REFERENCES

[1] G. E. Collins, "Quantifier elimination for real closed fields by cylindrical algebraic decomposition-preliminary report," *ACM SIGSAM Bulletin*, vol. 8, no. 3, pp. 80–90, 1974.

[2] Y. V. Matiyasevich, *Hilbert's Tenth Problem*, ser. Foundations of computing. MIT Press, 1993.

[3] D. Richardson, "Some undecidable problems involving elementary functions of a real variable," *J. Symb. Log.*, vol. 33, no. 4, pp. 514–520, 1968.

[4] A. Cimatti, A. Griggio, A. Irfan, M. Roveri, and R. Sebastiani, "Invariant checking of NRA transition systems via incremental reduction to LRA with EUF," in *TACAS*, ser. LNCS, vol. 10205, 2017, pp. 58–75.

[5] A. Cimatti, A. Griggio, A. Irfan, M. Roveri, and R. Sebastiani, "Satisfiability modulo transcendental functions via incremental linearization," in *CADE*, ser. Lecture Notes in Computer Science, vol. 10395. Springer, 2017, pp. 95–113.

[6] A. Cimatti, A. Griggio, A. Irfan, M. Roveri, and R. Sebastiani, "Incremental linearization for satisfiability and verification modulo nonlinear arithmetic and transcendental functions," *ACM TOCL*, vol. 19, pp. 19:1–19:52, 2018.

[7] A. Cimatti, A. Griggio, A. Irfan, M. Roveri, and R. Sebastiani, "Experimenting on solving nonlinear integer arithmetic with incremental linearization," in *SAT*, ser. LNCS. Springer, 2018, pp. 383–398.

[8] I. Niven, *Numbers: Rational and Irrational*. Mathematical Association of America, 1961.

[9] J. Berggren, J. Borwein, and P. Borwein, *Pi: A Source Book*. Springer New York, 2013.

[10] V. Weispfenning, "Quantifier elimination for real algebra - the quadratic case and beyond," *Appl. Algebra Eng. Commun. Comput.*, vol. 8, no. 2, pp. 85–101, 1997.

[11] F. Corzilius, G. Kremer, S. Junges, S. Schupp, and E. Ábrahám, "SMT-RAT: an open source C++ toolbox for strategic and parallel SMT solving," in *SAT*, ser. LNCS, vol. 9340. Springer, 2015, pp. 360–368.

[12] L. M. de Moura and N. Bjørner, "Z3: an efficient SMT solver," in *TACAS*, ser. LNCS, vol. 4963. Springer, 2008, pp. 337–340.

[13] B. Dutertre, "Yices 2.2," in *CAV*, ser. LNCS, vol. 8559. Springer, 2014, pp. 737–744.

[14] F. Benhamou and L. Granvilliers, "Continuous and interval constraints," in *Handbook of Constraint Programming*. Elsevier, 2006, vol. 2, pp. 571–603.

[15] V. X. Tung, T. V. Khanh, and M. Ogawa, "raSAT: An SMT solver for polynomial constraints," in *IJCAR*, ser. LNCS, vol. 9706. Springer, 2016, pp. 228–237.

[16] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert, "Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure," *JSAT*, vol. 1, no. 3-4, pp. 209–236, 2007.

[17] S. Gao, S. Kong, and E. M. Clarke, "dReal: An SMT solver for nonlinear theories over the reals," in *CADE-24*, ser. LNCS, vol. 7898. Springer, 2013, pp. 208–214.

[18] P. Fontaine, M. Ogawa, T. Sturm, and X. Vu, "Subtropical satisfiability," in *FroCoS*, ser. LNCS, vol. 10483. Springer, 2017, pp. 189–206.

[19] A. Maréchal, A. Fouilhé, T. King, D. Monniaux, and M. Périn, "Polyhedral approximation of multivariate polynomials using Handelman's Theorem," in *VMCAI*, ser. LNCS, vol. 9583. Springer, 2016, pp. 166–184.

[20] A. Reynolds, C. Tinelli, D. Jovanovic, and C. Barrett, "Designing theory solvers with extensions," in *FroCoS*, ser. LNCS, vol. 10483. Springer, 2017.

[21] C. Borralleras, S. Lucas, A. Oliveras, E. Rodríguez-Carbonell, and A. Rubio, "SAT modulo linear arithmetic for solving polynomial constraints," *JAR*, vol. 48, pp. 107–131, 2012.

[22] C. Barrett, C. L. Conway, M. Deters, L. Hadarean, D. Jovanovic, T. King, A. Reynolds, and C. Tinelli, "CVC4," in *CAV*, ser. LNCS, vol. 6806. Springer, 2011, pp. 171–177.

[23] C. Fuhs, J. Giesl, A. Middeldorp, P. Schneider-Kamp, R. Thiemann, and H. Zankl, "SAT solving for termination analysis with polynomial interpretations," in *SAT*, ser. LNCS. Springer, 2007, pp. 340–354.

[24] H. Zankl and A. Middeldorp, "Satisfiability of non-linear (ir)rational arithmetic," in *LPAR-16, 2010, Revised Selected Papers*, 2010, pp. 481–500.

[25] D. Jovanovic, "Solving nonlinear integer arithmetic with MCSAT," in *VMCAI 2017, Proceedings*, ser. LNCS, vol. 10145. Springer, 2017, pp. 330–346.

[26] G. Kremer, F. Corzilius, and E. Ábrahám, "A generalised branch-and-bound approach and its application in SAT modulo nonlinear integer arithmetic," in *CASC 2016, Proceedings*, ser. LNCS, vol. 9890. Springer, 2016, pp. 315–335.

[27] B. Akbarpour and L. C. Paulson, "MetiTarski: An automatic theorem prover for real-valued special functions," *J. Autom. Reasoning*, vol. 44, no. 3, pp. 175–205, 2010.

[28] A. Eggers, E. Kruglov, S. Kupferschmid, K. Scheibler, T. Teige, and C. Weidenbach, "Superposition modulo nonlinear arithmetic," in *FroCoS*, ser. LNCS, vol. 6989. Springer, 2011, pp. 119–134.

[29] A. Cimatti, A. Griggio, B. J. Schaafsma, and R. Sebastiani, "The MathSAT5 SMT Solver," in *TACAS*, ser. LNCS, vol. 7795. Springer, 2013, pp. 93–107.

[30] A. Irfan, "Incremental linearization for satisfiability and verification modulo nonlinear arithmetic and transcendental functions," Ph.D. dissertation, University of Trento, 2018. [Online]. Available: http://es-static.fbk.eu/people/irfan/papers/thesis.pdf

[31] C. Barrett, P. Fontaine, and C. Tinelli, "The Satisfiability Modulo Theories Library (SMT-LIB)," `www.SMT-LIB.org`, 2016.

[32] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv symbolic model checker," in *CAV*, ser. LNCS, vol. 8559. Springer, 2014, pp. 334–342.

[33] A. Cimatti, A. Griggio, and R. Sebastiani, "Efficient generation of Craig interpolants in satisfiability modulo theories," *ACM TOCL*, vol. 12, pp. 7:1–7:54, 2010.

[34] R. Sebastiani and S. Tomasi, "Optimization modulo theories with linear rational costs," *ACM TOCL*, vol. 16, no. 2, pp. 12:1–12:43, 2015.