Satisfiability Modulo Transcendental Functions via Incremental Linearization

Alberto Griggio

Fondazione Bruno Kessler, Trento, Italy

Joint work with A. Irfan, A. Cimatti, M. Roveri, R. Sebastiani

Executive Summary

- **Main idea**
  - Abstract transcendental functions with uninterpreted functions
  - Incrementally add upper- and lower- bound linear lemmas
    - Tangent and secant lines
    - Added to refine spurious models
- **Challenges**
  - Irrational values: transcendental functions give irrational outputs to (most) rational inputs
  - Linearization requires calculation of slope at arbitrary point, which is not straightforward
  - Handling periodicity (of trigonometric functions)
  - Detecting SAT

Some Math Background

- Transcendental function $f(x)$: doesn't satisfy a polynomial equation
  - We assume to be continuous and (n-times) differentiable
- Tangent line to $f(x)$ at point a : $T_{f,a}(x) \stackrel{\text{def}}{=} f(a) + \frac{d}{dx} f(a) * (x - a)$
- Secant line to $f(x)$ between a and b :

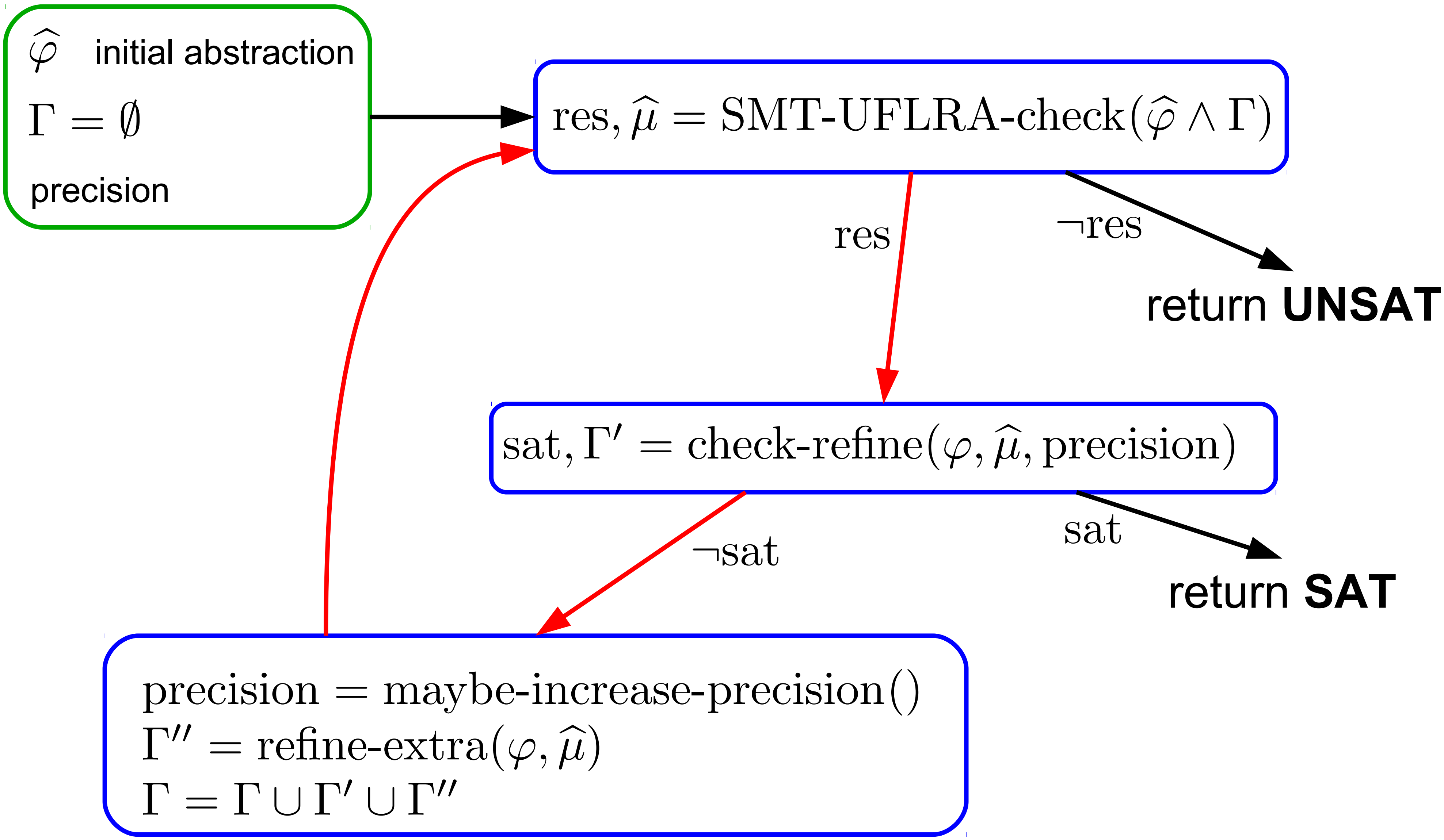
$$S_{f,a,b}(x) \stackrel{\text{def}}{=} \frac{f(a) - f(b)}{a - b} * (x - a) + f(a)$$

- Concavity: sign of the second derivative
- Taylor theorem: $f(x) = \underbrace{\sum_{i=0}^{n} \frac{f^{(i)}(a)}{i!} * (x-a)^i}_{P_{n,f(a)}} + R_{n+1,f(a)}(x)$

$$R_{n+1,f(a)}(x) \leq \underbrace{\max_{c \in [\min(a,x), \max(a,x)]} (|f^{(n+1)}(c)|) * \frac{|(x-a)^{n+1}|}{(n+1)!}}_{\overline{R_{n+1,f(a)}}^{u}(x)}$$

$$P_{n,f(a)}(x) - \overline{R_{n+1,f(a)}}^{u}(x) \leq f(x) \leq P_{n,f(a)}(x) + \overline{R_{n+1,f(a)}}^{u}(x)$$

Main Algorithm



Initial Abstraction

- Replace every occurrence of a transcendental function $tf(x)$ with a corresponding uninterpreted function $uf(x)$

- Add some basic lemmas about the behaviour of the function
  - E.g. for exponential

$$\exp(x) > 0$$

$$(x = 0 \leftrightarrow \exp(x) = 1) \wedge (x < 0 \leftrightarrow \exp(x) < 1) \wedge (x > 0 \leftrightarrow \exp(x) > 1)$$

$$x = 0 \vee \exp(x) > x + 1$$

Spuriousness Check

- Check that the model is consistent wrt $tf(x)$
  - Intuitively, check $\widehat{\mu}[uf(x)] = tf(\widehat{\mu}[x])$
- Problem: $tf(\widehat{\mu}[x])$ is typically irrational
  - Can't check precisely
- Solution: check whether $\widehat{\mu}[uf(x)]$ is **close enough** to $tf(\widehat{\mu}[x])$
  - use Taylor's theorem to compute polynomial bounds

$$P_l(\widehat{\mu}[x]) \leq tf(\widehat{\mu}[x]) \leq P_u(\widehat{\mu}[x])$$

    - Depend on the current precision
  - Model is definitely spurious if

$\widehat{\mu}[uf(x)] < P_l(\widehat{\mu}[x])$ or $\widehat{\mu}[uf(x)] > P_u(\widehat{\mu}[x])$

Spuriousness Check and Refinement

```
while true:
  e := 10^-precision
  L := {}
```

Spuriousness Check and Refinement

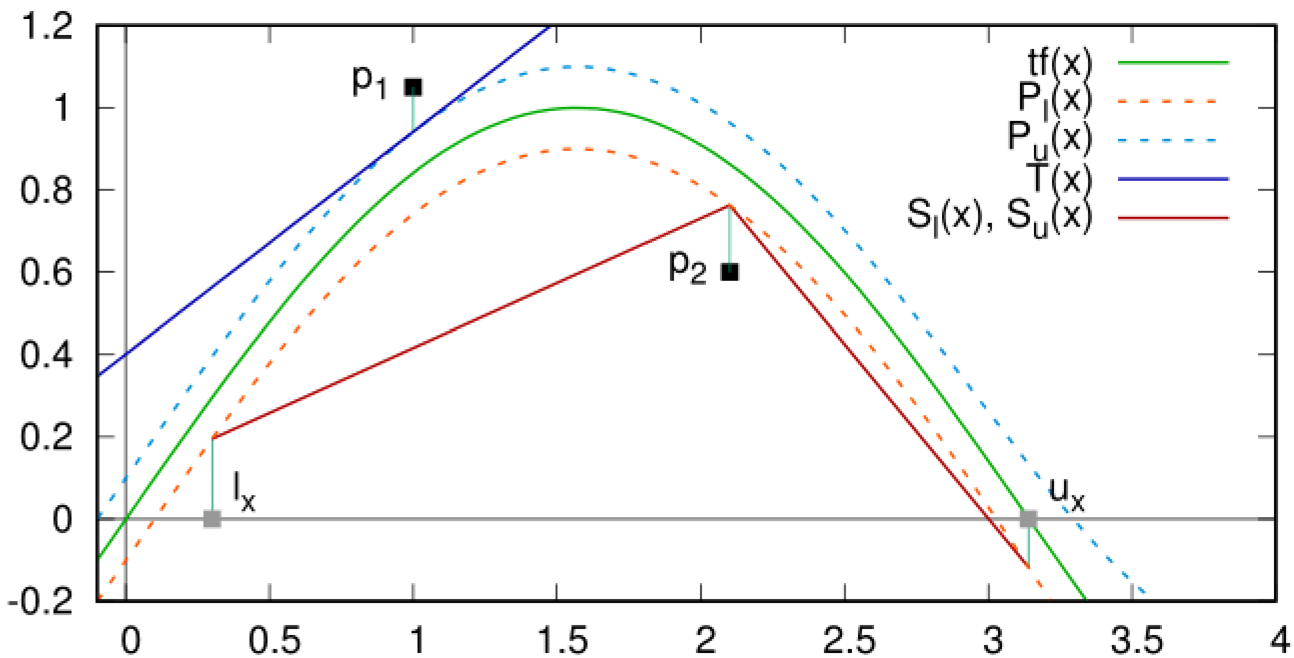
```
while true:
  e := 10^-precision
  L := {}
  for all tf(x) in φ:
    c := μ̂[x]
    P_l(x), P_u(x) := poly-approx(tf(x),c,e)
    if μ̂[uf(x)] < P_l(μ̂[x]) or μ̂[uf(x)] > P_u(μ̂[x]):
      L := L + get-lemmas-point(tf(x),μ̂,P_l(x),P_u(x))
```

Spuriousness Check and Refinement

```
while true:
  e := 10^-precision
  L := {}
  for all tf(x) in φ:
    c := μ̂[x]
    P_l(x), P_u(x) := poly-approx(tf(x),c,e)
    if μ̂[uf(x)] < P_l(μ̂[x]) or μ̂[uf(x)] > P_u(μ̂[x]):
      L := L + get-lemmas-point(tf(x),μ̂,P_l(x),P_u(x))
  if L is empty:
    if check-sat(φ, μ̂ ):
      return true
    else:
      precision += 1
  else:
    return false, L
```

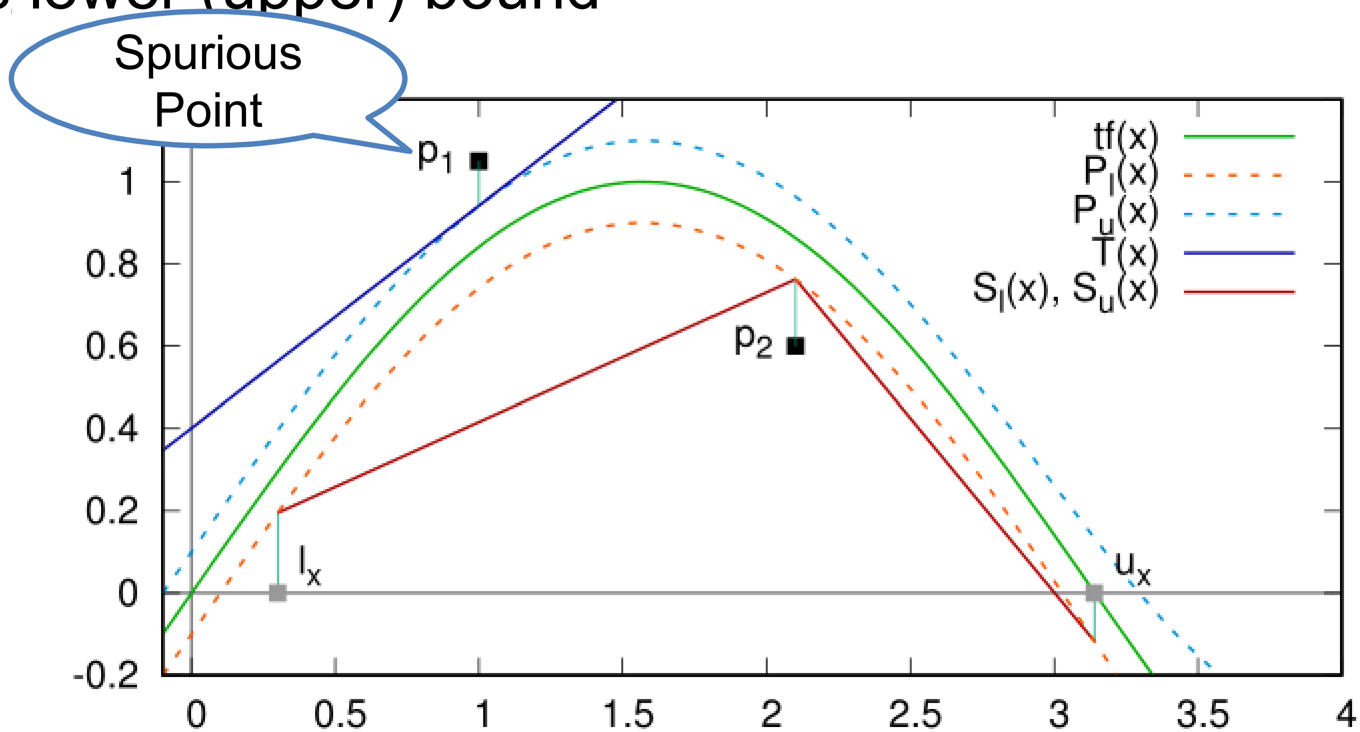
Refinement via Linearization – Basic Idea

- Use upper and lower **polynomials** for linearization
- If the **concavity** of the function is negative (positive):
  - **Tangent Refinement:** Tangent Line to the upper (lower) polynomial gives upper (lower) bound
  - **Secant Refinement:** Secant Line to the lower (upper) polynomial gives lower (upper) bound



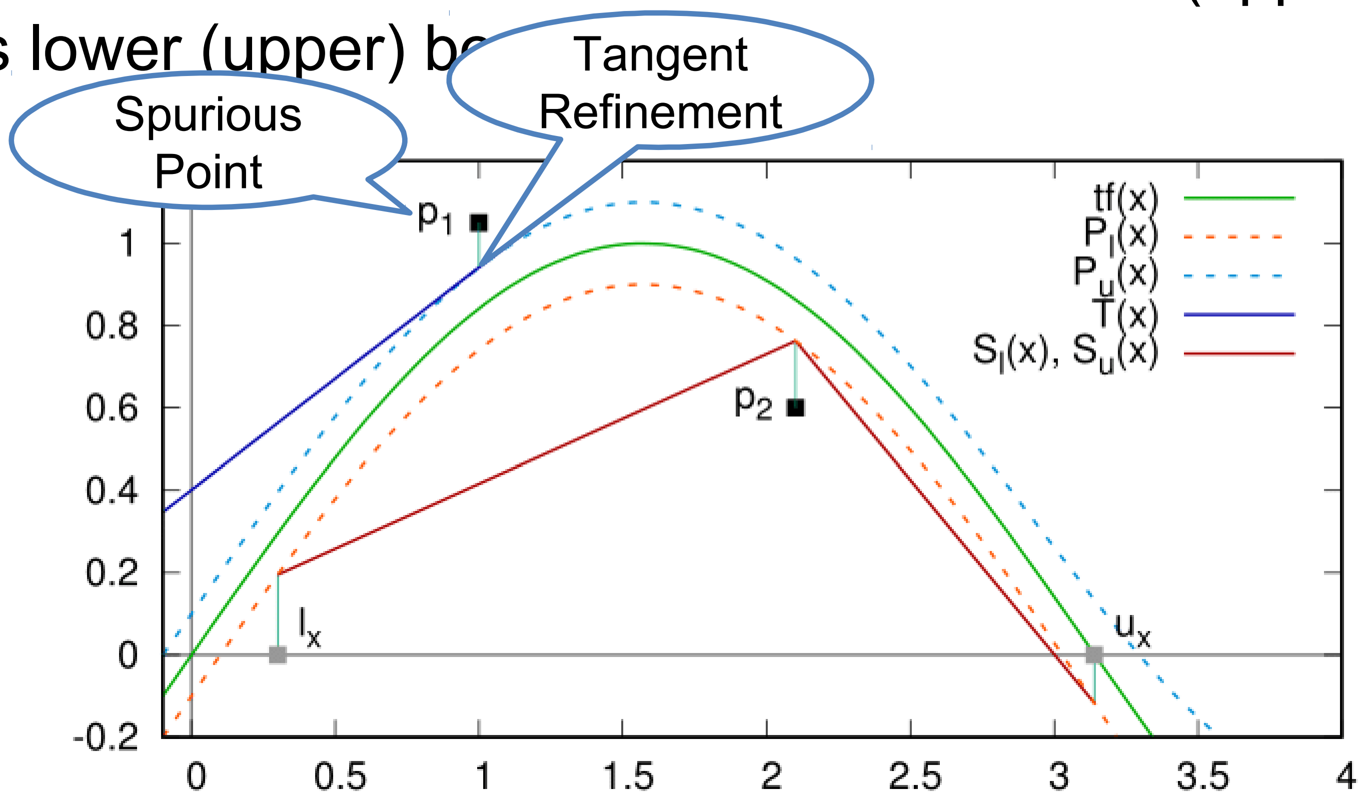
Refinement via Linearization – Basic Idea

- Use upper and lower **polynomials** for linearization
- If the **concavity** of the function is negative (positive):
  - **Tangent Refinement:** Tangent Line to the upper (lower) polynomial gives upper (lower) bound
  - **Secant Refinement:** Secant Line to the lower (upper) polynomial gives lower (upper) bound



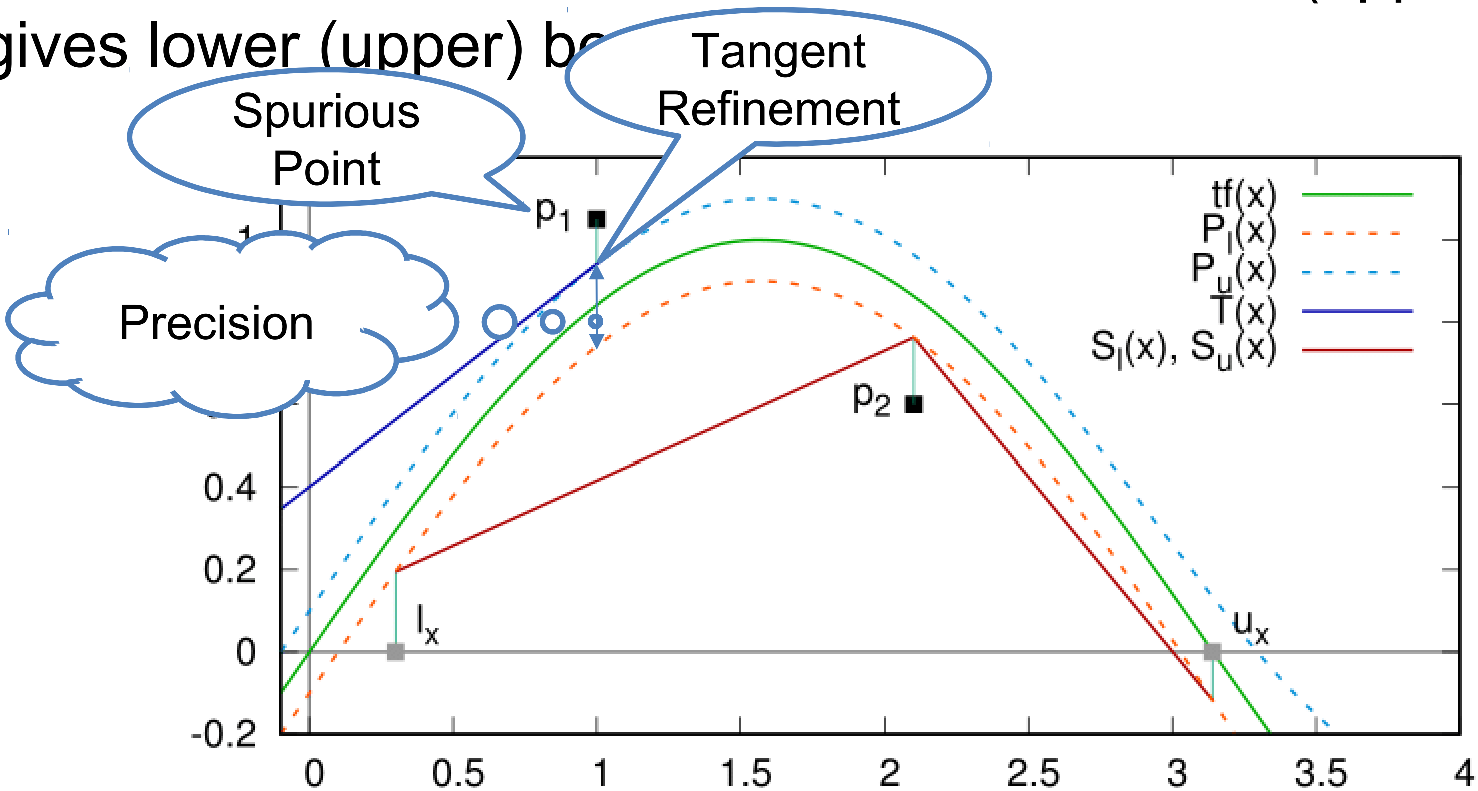
Refinement via Linearization – Basic Idea

- Use upper and lower **polynomials** for linearization
- If the **concavity** of the function is negative (positive):
  - **Tangent Refinement:** Tangent Line to the upper (lower) polynomial gives upper (lower) bound
  - **Secant Refinement:** Secant Line to the lower (upper) polynomial gives lower (upper) bo



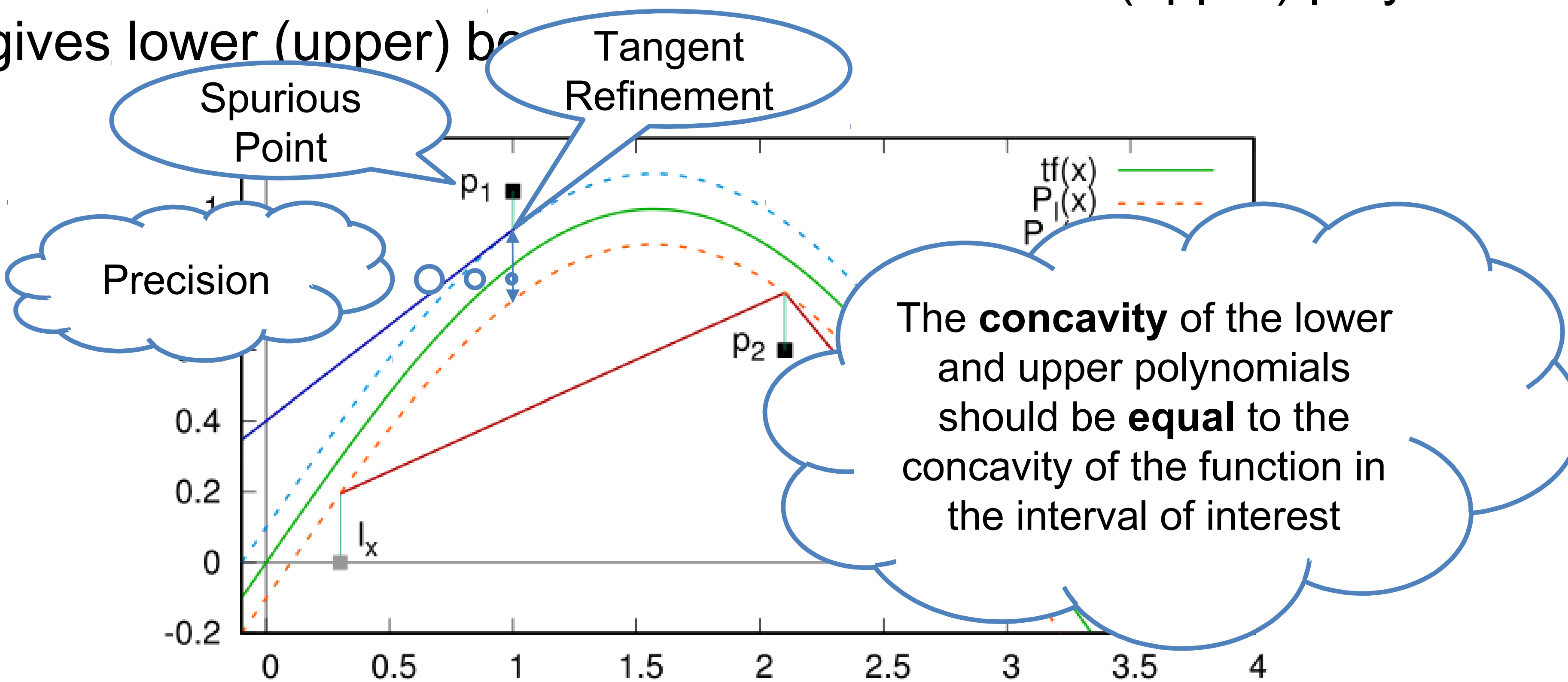
Refinement via Linearization – Basic Idea

- Use upper and lower **polynomials** for linearization
- If the **concavity** of the function is negative (positive):
  - **Tangent Refinement:** Tangent Line to the upper (lower) polynomial gives upper (lower) bound
  - **Secant Refinement:** Secant Line to the lower (upper) polynomial gives lower (upper) bo



Refinement via Linearization – Basic Idea

- Use upper and lower **polynomials** for linearization
- If the **concavity** of the function is negative (positive):
  - **Tangent Refinement:** Tangent Line to the upper (lower) polynomial gives upper (lower) bound
  - **Secant Refinement:** Secant Line to the lower (upper) polynomial gives lower (upper) bo[und]



Refinement: Exponential Function

- Using **Taylor's theorem**:

- **Case x = 0:**
  - Lower Polynomial: $P_l(0) = 1$
  - Upper Polynomial: $P_u(0) = 1$

- **Case x < 0:**
  - Lower Polynomial: $P_l(x) = \sum_{i=0}^{n} \frac{x^i}{i!}$ when n is odd
  - Upper Polynomial: $P_u(x) = \sum_{i=0}^{n} \frac{x^i}{i!}$ when n is even

- **Case x > 0:**
  - Lower Polynomial: $P_l(x) = \sum_{i=0}^{n} \frac{x^i}{i!}$
  - Upper Polynomial: $P_u(x) = \sum_{i=0}^{n} \frac{x^i}{i!} * (1 - \frac{x^{n+1}}{(n+1)!})^{-1}$

Sin Function

- We introduce a symbolic $\hat{\pi}$ variable with initial rational bounds

$$\frac{333}{106} < \hat{\pi} < \frac{355}{113}$$

- Reasoning is split depending on two periods:
  - **Base Period:** $-\hat{\pi}$ to $\hat{\pi}$
  - **Extended Period:** when not in the base period

- For each sin(x), new application sin(y_x)
  - y_x fresh (called a base variable)
  - The domain of y_x is in the base period
  - sin(x) and sin(y_x) are equal in the base period

$$(-\hat{\pi} \leq y_x \leq \hat{\pi}) \wedge ((-\hat{\pi} \leq x \leq \pi) \rightarrow y_x = x) \wedge fsin(x) = fsin(y_x)$$

Sin Function

- Tangent and secant refinement only with base variables

- Concavity check in the base period is easy
  - **Case x > 0:** concavity is negative
  - **Case x < 0:** concavity is positive

- Using Taylor's theorem and current precision
  - Lower Polynomial:

$$P_l(y_x) = \sum_{k=0}^{n} \frac{(-1)^k * y_x^{2k+1}}{(2k+1)!} - \frac{y_x^{2(n+1)}}{(2(n+1))!}$$

  - Upper Polynomial:

$$P_u(y_x) = \sum_{k=0}^{n} \frac{(-1)^k * y_x^{2k+1}}{(2k+1)!} + \frac{y_x^{2(n+1)}}{(2(n+1))!}$$

Sin Function – Extended periods

- Shift $\widehat{\mu}[x]$ to the base period, and compare with $\widehat{\mu}[y_x]$
  - Shift calculation $s = (\hat{\mu}[x] + \hat{\mu}[\hat{\pi}])/(2 \cdot \hat{\mu}[\hat{\pi}])$
- If the values differ, we perform shift refinement
  - Relate the extended period with the base period (after appropriate shift)

$$(\hat{\pi} * (2s - 1) \leq x \leq \hat{\pi} * (2s + 1)) \rightarrow (y_x = x - 2s * \hat{\pi})$$

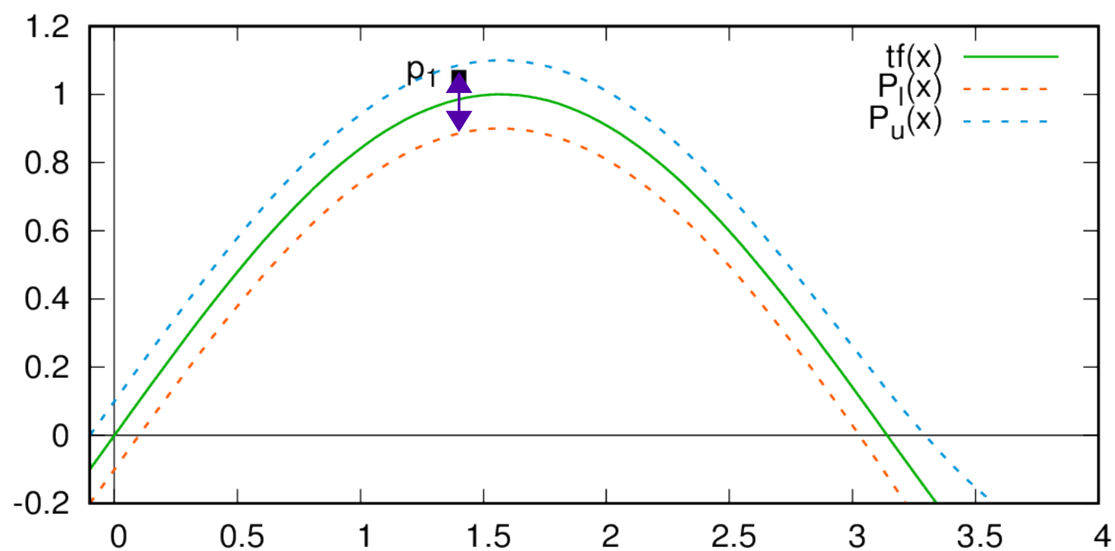
- Note that the shift is symbolic in $\hat{\pi}$
  - Ensure soundness

Check for SAT

- We know $\widehat{\mu} \models \widehat{\varphi}$

  $$P_l(\widehat{\mu}[x_i]) \leq \widehat{\mu}[uf(x_i)] \leq P_u(\widehat{\mu}[x_i])$$

  so, $\widehat{\mu}$ is a candidate solution



- Sufficient condition for sat: validity of

$$\forall uf \in \widehat{\varphi}. (\bigwedge_i P_l(\widehat{\mu}[x_i]) \leq uf(\widehat{\mu}[x]) \leq P_u(\widehat{\mu}[x])) \rightarrow \widehat{\varphi}\{X \mapsto \widehat{\mu}[X]\}$$

Replace $uf(\widehat{\mu}[x_i])$'s with fresh vars y_i and check validity of the first-order formula

$$\forall Y. ((\bigwedge_i P_l(\widehat{\mu}[x_i]) \leq y_i \leq P_u(\widehat{\mu}[x])) \rightarrow \widehat{\varphi}\{X \mapsto \widehat{\mu}[X]\}) \{Y \mapsto uf(\widehat{\mu}[X])\}$$

Implementation and Experiments

- Prototype Implementation in MathSAT + PySMT

- 887 BMC Benchmarks (also scaled)
  - Hand-crafted benchmarks
  - Discretized Hybrid System benchmarks
  - HyComp benchmarks
  - iSAT benchmarks
  - HyST benchmarks
  - HARE benchmarks
- 681 MetiTarski Benchmarks (also scaled)
- 944 dReal Benchmarks

- Tools
  - **MathSAT**
  - MetiTarski
  - iSAT3
  - dReal

Results

Benchmarks	Bounded Model Checking (887)			Mathematical (681)		
Result	SAT	UNSAT	MaybeSAT	SAT	UNSAT	MaybeSAT
METITARSKI	N.A.	N.A.	N.A.	N.A.	**530**	N.A.
MATHSAT	**72**	553	N.A.	0	210	N.A.
MATHSAT-NOUNISAT	44	**554**	N.A.	0	221	N.A.
iSAT3	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
DREAL	N.A.	392	281 (67/23)	N.A.	285	316 (0/253)
Benchmarks	**Scaled Bounded Model Checking (887)**			**Scaled Mathematical (681)**		
Result	**SAT**	**UNSAT**	**MaybeSAT**	**SAT**	**UNSAT**	**MaybeSAT**
MATHSAT	**84**	**556**	N.A.	0	215	N.A.
MATHSAT-NOUNISAT	48	**556**	N.A.	0	229	N.A.
iSAT3	35	470	87 (32/7)	0	212	137 (0/115)
DREAL	N.A.	403	251 (77/23)	N.A.	**302**	245 (0/195)

Table 1. Results on the BMC and Metitarski benchmarks.

Results

Benchmarks	dREAL (all) (944)		
Status	SAT	UNSAT	MaybeSAT
dREAL (orig.)	N.A.	**102**	524(3/4)
MATHSAT	**3**	68	N.A.
dREAL	N.A.	44	57(3/4)

Benchmarks	dREAL (exp/sin only) (96)		
Status	SAT	UNSAT	MaybeSAT
dREAL (orig.)	N.A.	17	37 (3/3)
MATHSAT	**3**	39	N.A.

Table 2. Results on the Dreal benchmarks.

Thank You