Newton Refinement as Alternative to Craig Interpolation in CPAchecker

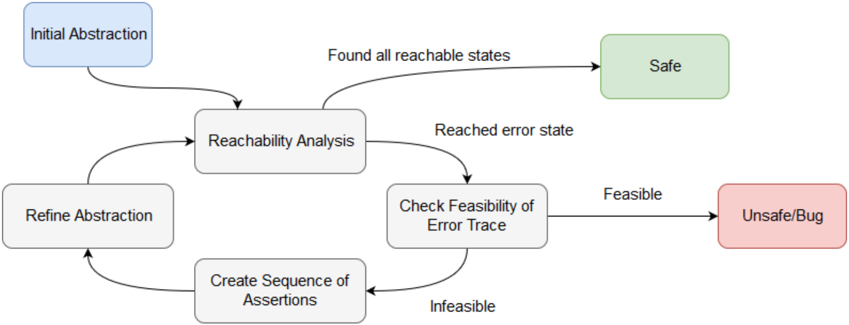
Matthias Gerlach

09.01.2019

Motivation Verification

- Growing reliance on computer systems
- Ensure specifications
- Tests only covers a subset of scenarios
- Verification can prove specification over all scenarios
- Common approach: Predicate Analysis based on CEGAR

Counterexample Guided Abstraction Refinement



- Requires a method to extract state assertions from error traces
  - Typical approach: **Craig Interpolation**
  - Alternative (previous) approach: **Newton Refinement**

Motivation and Goals of this Thesis

- Motivation for Newton Refinement
  - Alternative to Craig Interpolation
  - Recent paper "Craig vs. Newton" shows comparable results to interpolation
  - Limited number of SMT-solvers supporting interpolation
- Implementation of Newton Refinement in CPAchecker
- Evaluation of the method based on Benchmarks

Statements and Path Formulas

- Two types of statements
  - Assignment: $x := e$
  - Assume statement: `assume` φ
- Each statement can be translated to a path formula:
  - Assignment: $x_i = \text{rename}_i(e)$
  - Assume statement: $\text{rename}_i(\varphi)$
- $\text{rename}_i(\psi)$
  - Replaces all program variables with indexed version
  - Index is location of last assignment new value.

Basic Approach - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Trace and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$

Basic Approach - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Trace and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$

Basic Approach - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Trace and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$

Basic Approach - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Trace and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Sequence of Assertions

- Has to fulfill following conditions:

$$\begin{aligned} \varphi_0 &= \texttt{true} \\ \varphi_{i+1} &= \textbf{SP}(\varphi_i, st_{i+1}) \qquad \text{for } i = 0, \dots, n-1 \\ \varphi_n &= \texttt{false} \end{aligned}$$

- $\textbf{SP}(\varphi_i, st_{i+1})$ is the strongest postcondition of the statement st_{i+1}
  - Conjunction of φ_i and F_i
  - For Assignments: Existentially quantify old indexed variables

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
2	$y := 1;$	
3	$\texttt{assume } x < 1;$	
4	$x = x + 1;$	
5	$\texttt{assume } \neg(x < 1);$	
6	$\texttt{assume } x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
3	$\texttt{assume } x < 1;$	
4	$x = x + 1;$	
5	$\texttt{assume } \neg(x < 1);$	
6	$\texttt{assume } x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \wedge y_2 = 1$
3	$\texttt{assume}\ x < 1;$	
4	$x = x + 1;$	
5	$\texttt{assume}\ \neg(x < 1);$	
6	$\texttt{assume}\ x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \land y_2 = 1$
3	$\texttt{assume } x < 1;$	
		$\varphi_3 := x_1 = 0 \land y_2 = 1 \land x_1 < 1$
4	$x = x + 1;$	
5	$\texttt{assume } \neg(x < 1);$	
6	$\texttt{assume } x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \wedge y_2 = 1$
3	$\texttt{assume}\ x < 1;$	
		$\varphi_3 := x_1 = 0 \wedge y_2 = 1 \wedge x_1 < 1$
4	$x = x + 1;$	
		$\varphi_4 := \exists \mathbf{x_1}.\mathbf{x_1} = \mathbf{0} \wedge \mathbf{y_2} = \mathbf{1} \wedge \mathbf{x_1} < \mathbf{1} \wedge \mathbf{x_4} = \mathbf{x_1} + \mathbf{1}$
5	$\texttt{assume}\ \neg(x < 1);$	
6	$\texttt{assume}\ x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \land y_2 = 1$
3	$\texttt{assume } x < 1;$	
		$\varphi_3 := x_1 = 0 \land y_2 = 1 \land x_1 < 1$
4	$x = x + 1;$	
		$\varphi_4 := x_4 = 1 \land y_2 = 1$
5	$\texttt{assume } \neg(x < 1);$	
6	$\texttt{assume } x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \land y_2 = 1$
3	$\texttt{assume}\ x < 1;$	
		$\varphi_3 := x_1 = 0 \land y_2 = 1 \land x_1 < 1$
4	$x = x + 1;$	
		$\varphi_4 := x_4 = 1 \land y_2 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	
		$\varphi_5 := x_4 = 1 \land y_2 = 1 \land \neg(x_4 < 1)$
6	$\texttt{assume}\ x \neq 1;$	

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \wedge y_2 = 1$
3	$\texttt{assume}\ x < 1;$	
		$\varphi_3 := x_1 = 0 \wedge y_2 = 1 \wedge x_1 < 1$
4	$x = x + 1;$	
		$\varphi_4 := x_4 = 1 \wedge y_2 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	
		$\varphi_5 := x_4 = 1 \wedge y_2 = 1 \wedge \neg(x_4 < 1)$
6	$\texttt{assume}\ x \neq 1;$	
		$\boldsymbol{\varphi_6 : x_4 = 1 \wedge y_2 = 1 \wedge \neg(x_4 < 1) \wedge x_4 \neq 1}$

Sequence of Assertions - Example

i	st_i	State assertions
		$\varphi_0 := \texttt{true}$
1	$x := 0;$	
		$\varphi_1 := x_1 = 0$
2	$y := 1;$	
		$\varphi_2 := x_1 = 0 \wedge y_2 = 1$
3	$\texttt{assume } x < 1;$	
		$\varphi_3 := x_1 = 0 \wedge y_2 = 1 \wedge x_1 < 1$
4	$x = x + 1;$	
		$\varphi_4 := x_4 = 1 \wedge y_2 = 1$
5	$\texttt{assume } \neg(x < 1);$	
		$\varphi_5 := x_4 = 1 \wedge y_2 = 1 \wedge \neg(x_4 < 1)$
6	$\texttt{assume } x \neq 1;$	
		$\varphi_6 := \texttt{false}$

Motivation for Abstractions

- State assertions very specific
- State assertions become very long

⇒ Abstraction of state assertions

- Infeasible Core abstraction
- Live Variable abstraction

Infeasible Core Abstraction

- **Idea:** Only add path formulas that are relevant
- Using Unsatisfiable Core
  - Subset of formulas, such that the conjunction is still unsatisfiable
  - Supported by most SMT solvers
- Unsatisfiable core of path formulas = Infeasible Core
- Only use path formulas of the infeasible core for state assertions

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$x_1 = 0$	
2	$y := 1;$	$y_2 = 1$	
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
4	$x = x + 1;$	$x_4 = x_1 + 1$	
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
6	$\texttt{assume}\ x \neq 1;$	$x_4 \neq 1$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 :=$ `true`
1	$x := 0;$	$\mathbf{x_1 = 0}$	
2	$y := 1;$	$y_2 = 1$	
3	`assume` $x < 1;$	$x_1 < 1$	
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
5	`assume` $\neg(x < 1);$	$x_4 \geq 1$	
6	`assume` $x \neq 1;$	$\mathbf{x_4 \neq 1}$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
6	$\texttt{assume}\ x \neq 1;$	$\mathbf{x_4 \neq 1}$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
			$\varphi_3 := x_1 = 0$
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
			$\boldsymbol{\varphi_4 := \exists x_1 . x_1 = 0 \land x_4 = x_1 + 1}$
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
6	$\texttt{assume}\ x \neq 1;$	$\mathbf{x_4 \neq 1}$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
			$\varphi_3 := x_1 = 0$
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
			$\varphi_4 := x_4 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
6	$\texttt{assume}\ x \neq 1;$	$\mathbf{x_4 \neq 1}$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume } x < 1;$	$x_1 < 1$	
			$\varphi_3 := x_1 = 0$
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
			$\varphi_4 := x_4 = 1$
5	$\texttt{assume } \neg(x < 1);$	$x_4 \geq 1$	
			$\varphi_5 := x_4 = 1$
6	$\texttt{assume } x \neq 1;$	$\mathbf{x_4 \neq 1}$	

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
			$\varphi_3 := x_1 = 0$
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
			$\varphi_4 := x_4 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
			$\varphi_5 := x_4 = 1$
6	$\texttt{assume}\ x \neq 1;$	$\mathbf{x_4 \neq 1}$	
			$\varphi_6 : \mathbf{x_4 = 1 \wedge x_4 \neq 1}$

With Infeasible Core Abstraction - Example

i	st_i	F_i	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\mathbf{x_1 = 0}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$y_2 = 1$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume}\ x < 1;$	$x_1 < 1$	
			$\varphi_3 := x_1 = 0$
4	$x = x + 1;$	$\mathbf{x_4 = x_1 + 1}$	
			$\varphi_4 := x_4 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	$x_4 \geq 1$	
			$\varphi_5 := x_4 = 1$
6	$\texttt{assume}\ x \neq 1;$	$\mathbf{x_4 \neq 1}$	
			$\varphi_6 := \texttt{false}$

Live Variable Abstraction

- **Idea:** Remove variables that are not used by following statements
- Identify sets of variables that are ...
  - ... Future live FL
  - ... Not future live $\overline{FL}$
- Existentially quantify variables of $\overline{FL}$

With Live Variable Abstraction - Example

i	st_i	$\overline{FL}$	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\{\}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$\{y_2\}$	
			$\varphi_2 := x_1 = 0 \land y_2 = 1$
3	$\texttt{assume}\ x < 1;$	$\{y_2\}$	
			$\varphi_3 := x_1 = 0 \land y_2 = 1 \land x_1 < 1$
4	$x = x + 1;$	$\{y_2, x_1\}$	
			$\varphi_4 := x_4 = 1 \land y_2 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	$\{y_2, x_1\}$	
			$\varphi_5 := x_4 = 1 \land y_2 = 1$
6	$\texttt{assume}\ x \neq 1;$	$\{y_2, x_1, x_4\}$	
			$\varphi_6 := \texttt{false}$

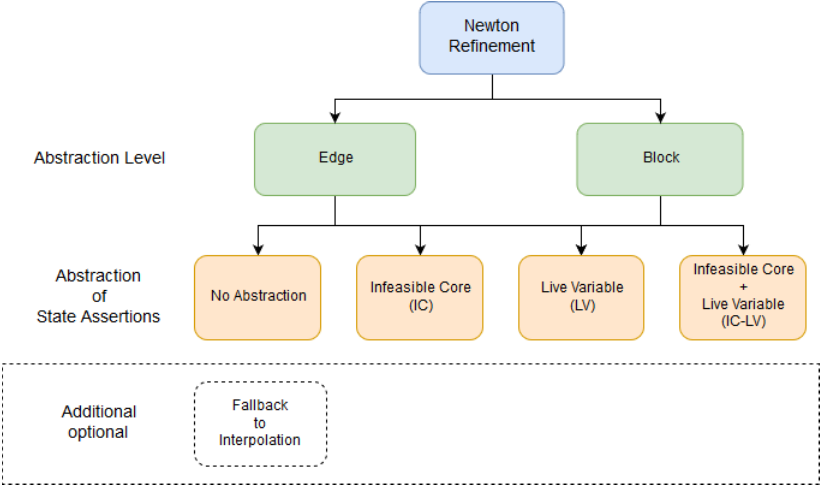
With Live Variable Abstraction - Example

i	st_i	$\overline{FL}$	State assertions
			$\varphi_0 := \texttt{true}$
1	$x := 0;$	$\{\}$	
			$\varphi_1 := x_1 = 0$
2	$y := 1;$	$\{y_2\}$	
			$\varphi_2 := x_1 = 0$
3	$\texttt{assume}\ x < 1;$	$\{y_2\}$	
			$\varphi_3 := x_1 = 0 \land x_1 < 1$
4	$x = x + 1;$	$\{y_2, x_1\}$	
			$\varphi_4 := x_4 = 1$
5	$\texttt{assume}\ \neg(x < 1);$	$\{y_2, x_1\}$	
			$\varphi_5 := x_4 = 1$
6	$\texttt{assume}\ x \neq 1;$	$\{y_2, x_1, x_4\}$	
			$\varphi_6 := \texttt{false}$

Implementation in CPAchecker

- New class `NewtonRefinementManager`
- Two approaches to get path formulas
  - **Edge-Level:** Edges in Control Flow Automaton
  - **Block-Level:** Block formulas of Abstract Reachability Graph
- Solver independent quantifier elimination in `PseudoExistQeManager`
  - Destructive Equality Resolution
  - Unconnected Parameter Drop

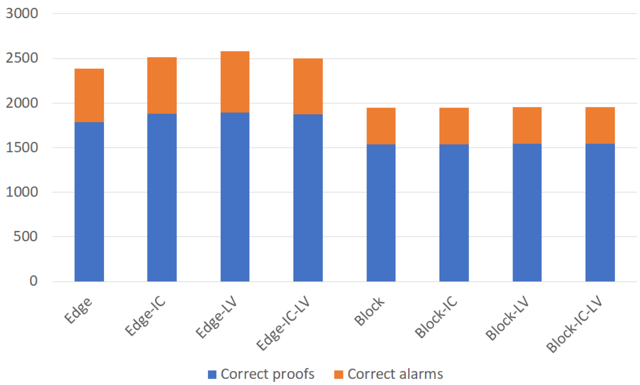
Configurations



Setting and Questions

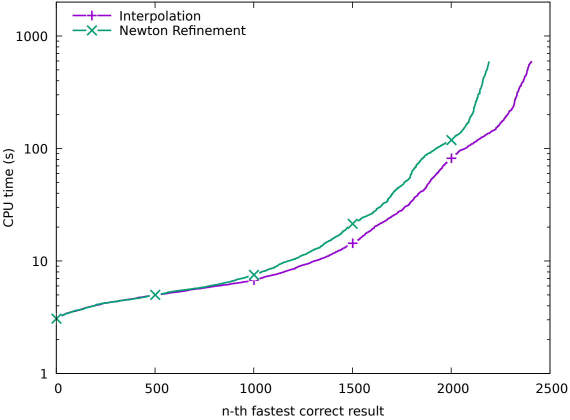
- Benchmark setting
  - SV-COMP 2018
  - Verifier Cloud
  - Linux 2CPUs(Intel Xeon), 15GB RAM
  - Timeout 600s
  - Solver: MathSAT5
- Interesting Questions
  - Best configuration of Newton Refinement?
  - Effectivity compared to Craig Interpolation?
  - Correct results where interpolation fails?

Comparing Newton Refinement Configurations

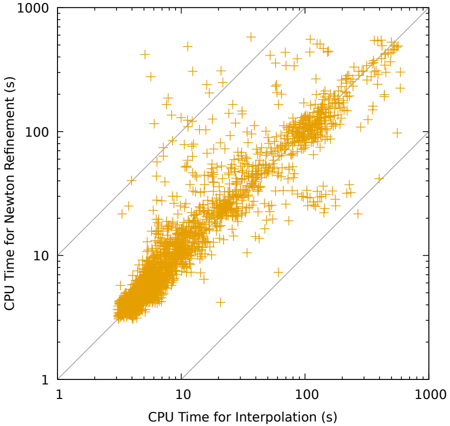


$\Rightarrow$ Best configuration: *Edge-LV*

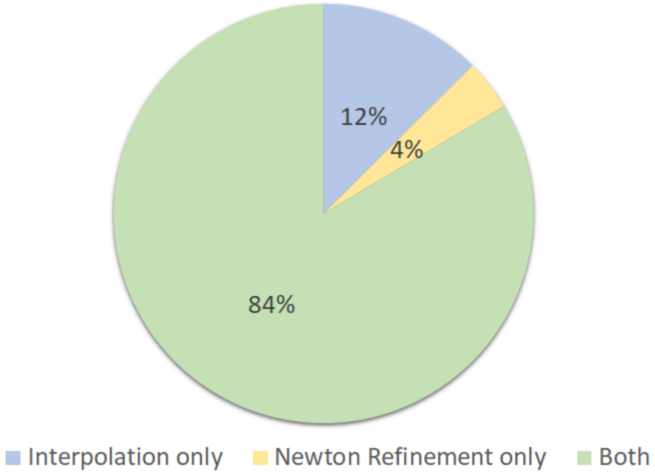
Comparison to Craig Interpolation Quantile Plot



Scatter Plot Computation Time



Percentage of exclusively proved Programs



Conclusion

- Alternative error trace refinement
- Best configuration: Edge-LV
- Proofs some programs, where interpolation fails
- Similar results for Z3
- Solver based quantifier elimination only slightly more successful

- Possible extension: Newton Refinement as fallback for interpolation

Backup - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Backup - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return −1;
}
```

Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Backup - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Backup - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Backup - Example

```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return -1;
}
```

Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

Backup - Example

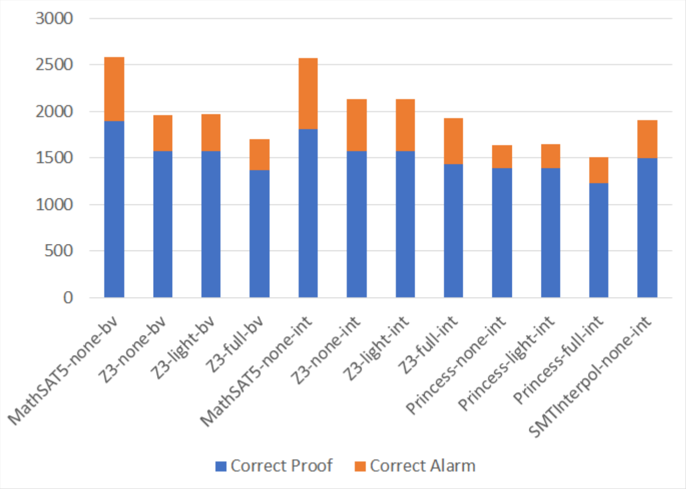
```
int main(){
  int x = 0;
  int y = 1;

  while(x < 1){
    x = x + 1;
  }
  if (x != 1){
    goto ERROR;
  }
  return 0;
ERROR:
  return −1;
}
```

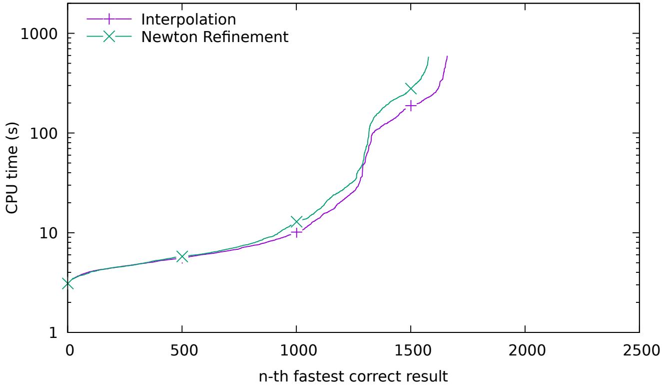
Statements and Path formula:

i	Statement st_i	Path formula F_i
1	$x := 0;$	$x_1 = 0$
2	$y := 1;$	$y_2 = 1$
3	assume $x < 1;$	$x_1 < 1$
4	$x = x + 1;$	$x_4 = x_1 + 1$
5	assume $\neg(x < 1);$	$\neg(x_4 < 1)$
6	assume $x \neq 1;$	$x_4 \neq 1$

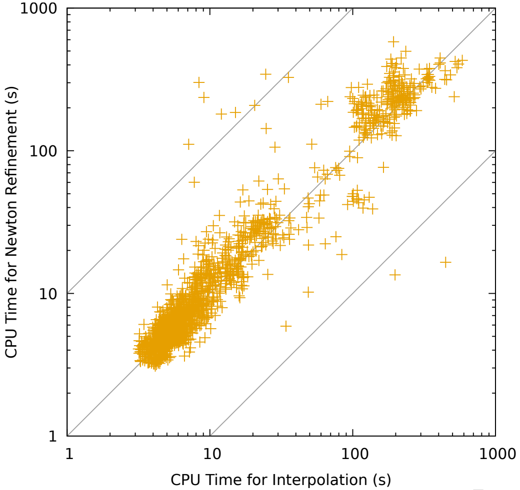
Backup Solver Benchmark



Backup Z3 Quantile



Backup Z3 Quantile



Backup Quantifier Elimination