Microsoft Research

Automatic theorem proving and SMT

Nikolaj Bjørner
Microsoft Research
HCSS, May 8, 2013

Outline

- Symbolic methods and SMT
- Z3 – what's new?
- Validating Network Access Restrictions
- Satisfiability of Horn Clauses

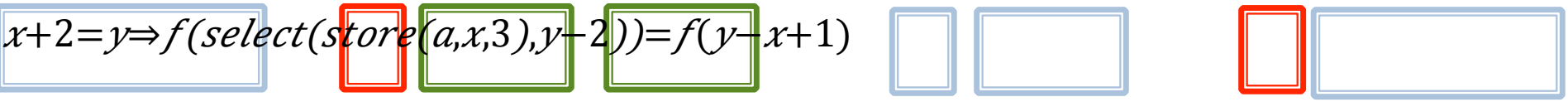
Symbolic methods and SMT

Satisfiability Modulo Theories (SMT)

Is formula φ satisfiable modulo theory T ?

SMT solvers have specialized algorithms for T

Satisfiability Modulo Theories (SMT)



Array Theory

Arithmetic

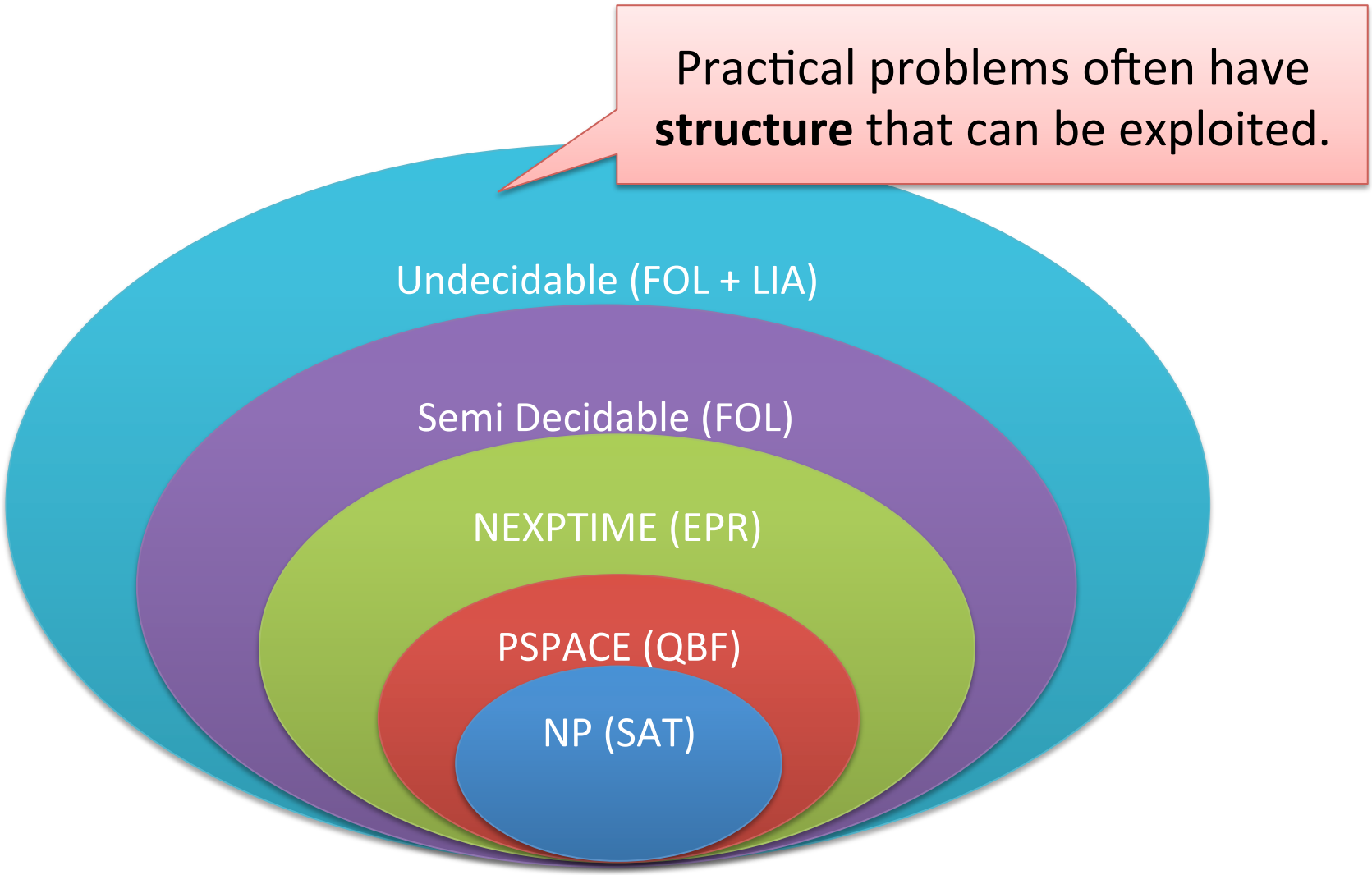
Uninterpreted Functions

$select(store(a,i,v),i)=v$

$i \neq j \Rightarrow select(store(a,i,v),j)=select(a,j)$

Symbolic Reasoning

High computational Complexity



Some Microsoft Tools using Z3

SAGE

HAVOC

BOOGIE

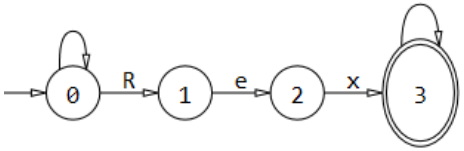
FORMULA
Modeling Foundations.

Vcc

The Spec# Programming System

SLAM

TERMINATOR

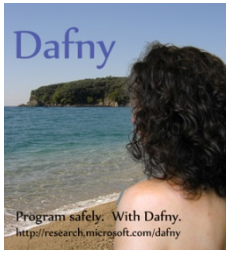


Yogi

SymDiff

Spec Explorer

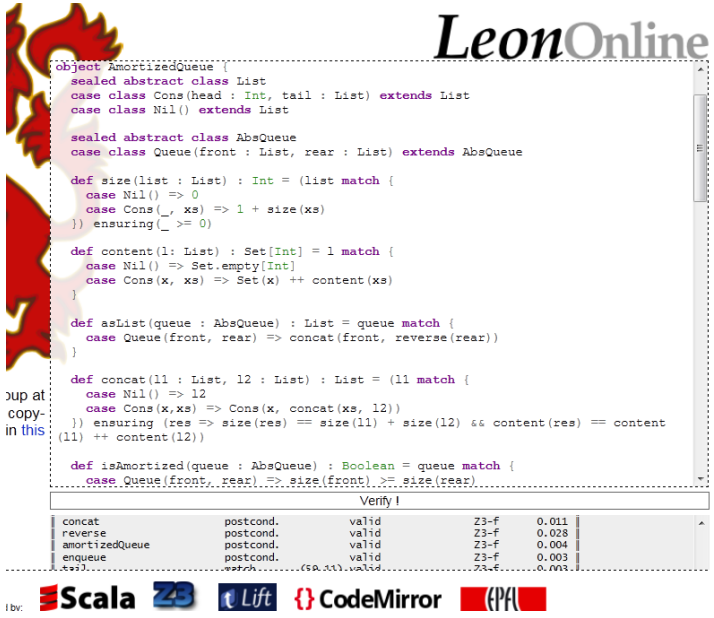
Pex



Other Cool tools using Z3

Sledge Hammer

Liquid Types



PUG

ESBMC

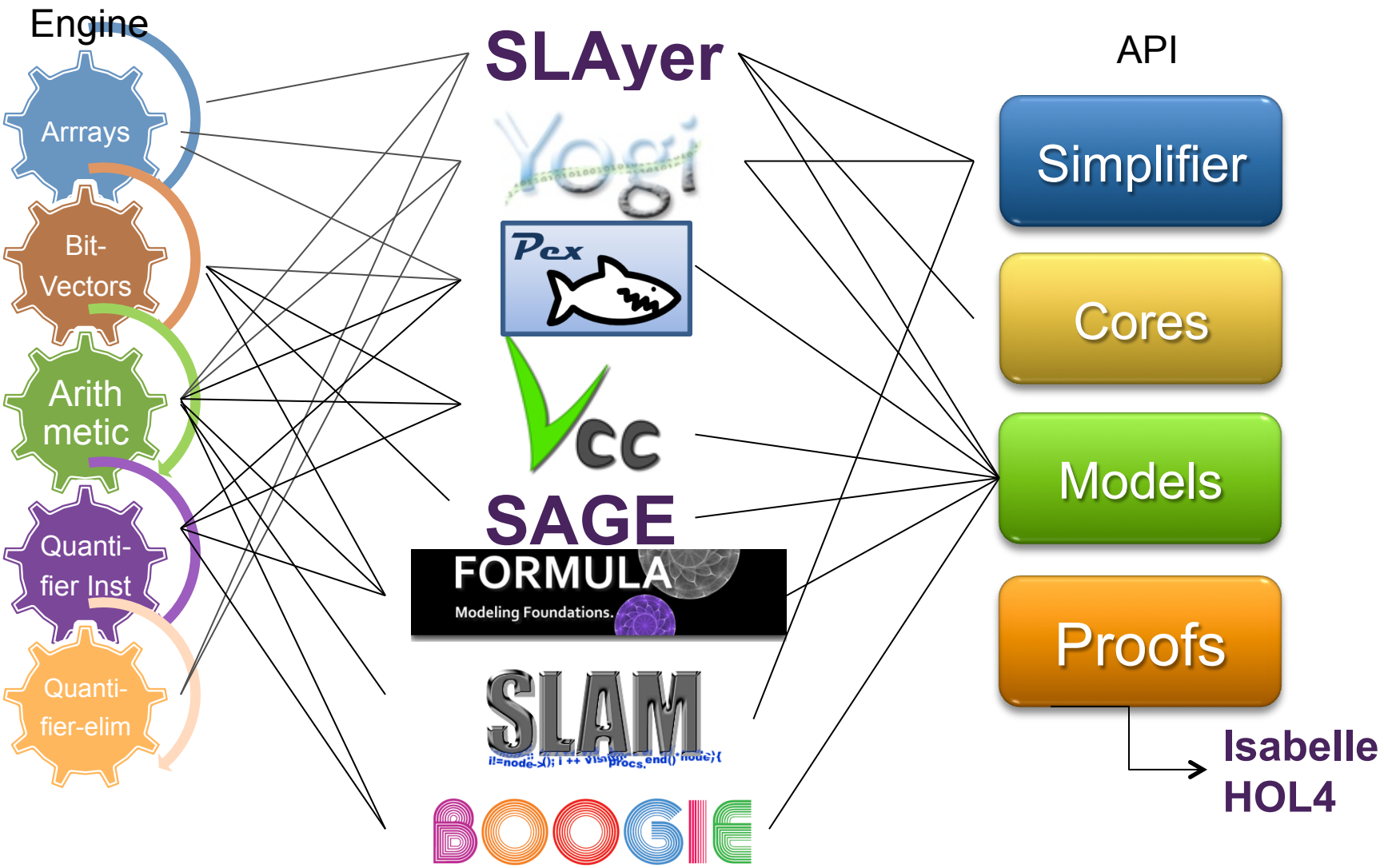
Scala^{Z3}

MetiTarski

KeYmaera

Jeves

Feature Usage



Z3: Little Engines of Proof

http://research.microsoft.com/projects/z3

What people say about Z3

Z3 is powerful: *Thank you for your advise and your powerful Z3 !*

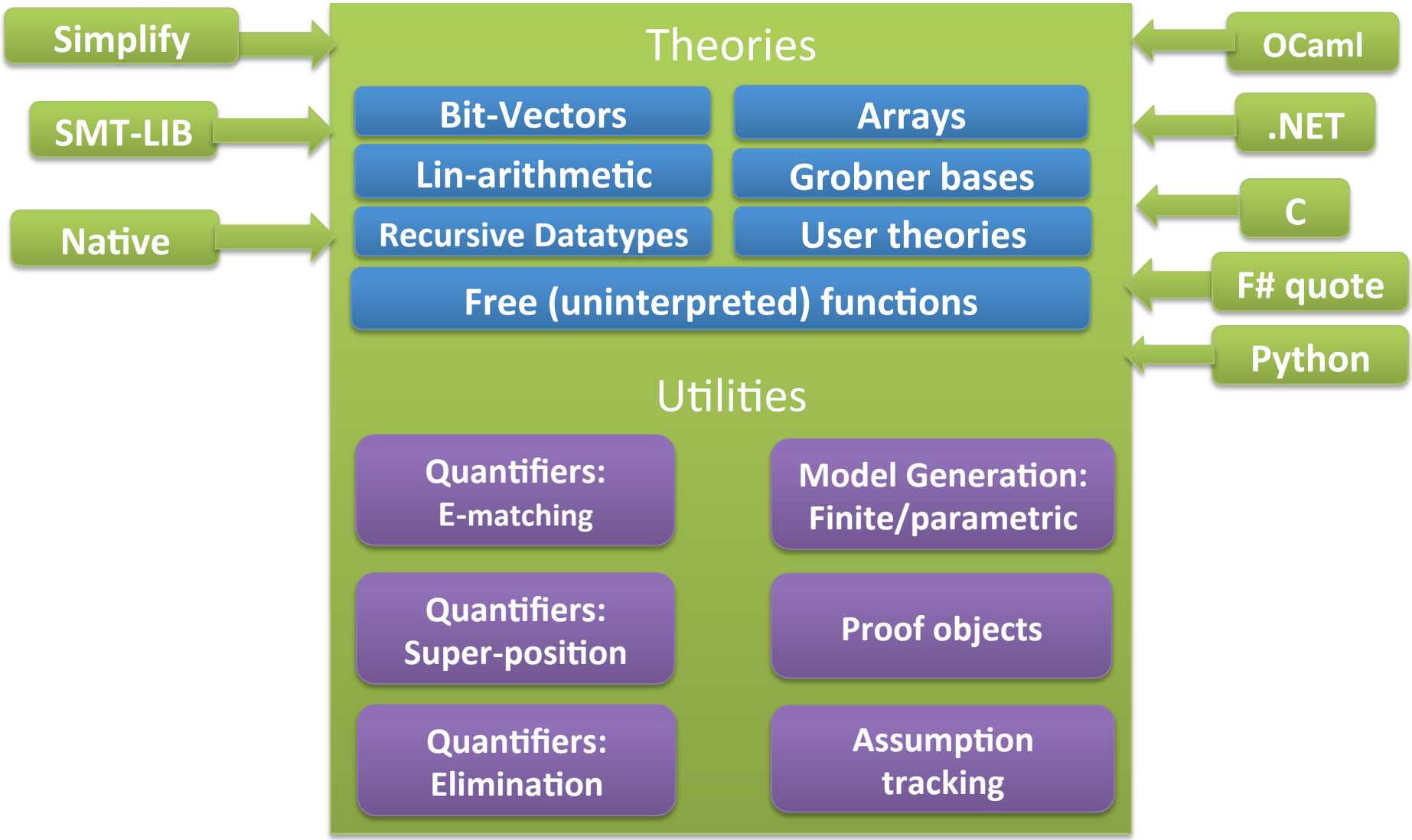
Z3 is a *crown jewel*. 

Z3 is not just a car: *Der neue Z3 is höllisch schnell (und ich meine kein Auto).*

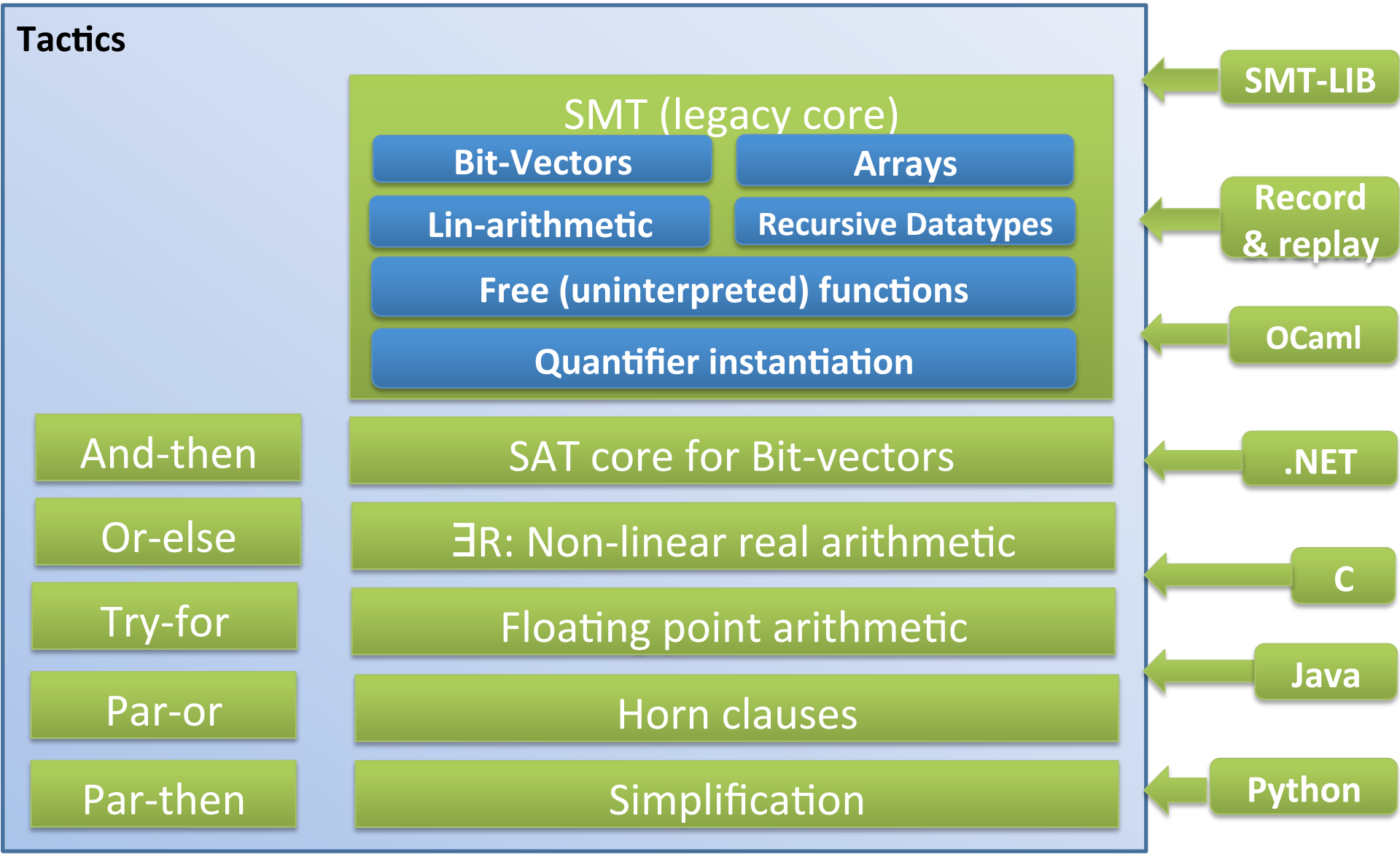
Z3 is smarter than the speaker: *I just meant that I am part of the PC to which you sent email so writing me that you sent email to the PC is ... well, redundant ☺. Even Z3 should be able to derive it.* [Andrei Voronkov]

– What’s new?

Z3 architecture - original

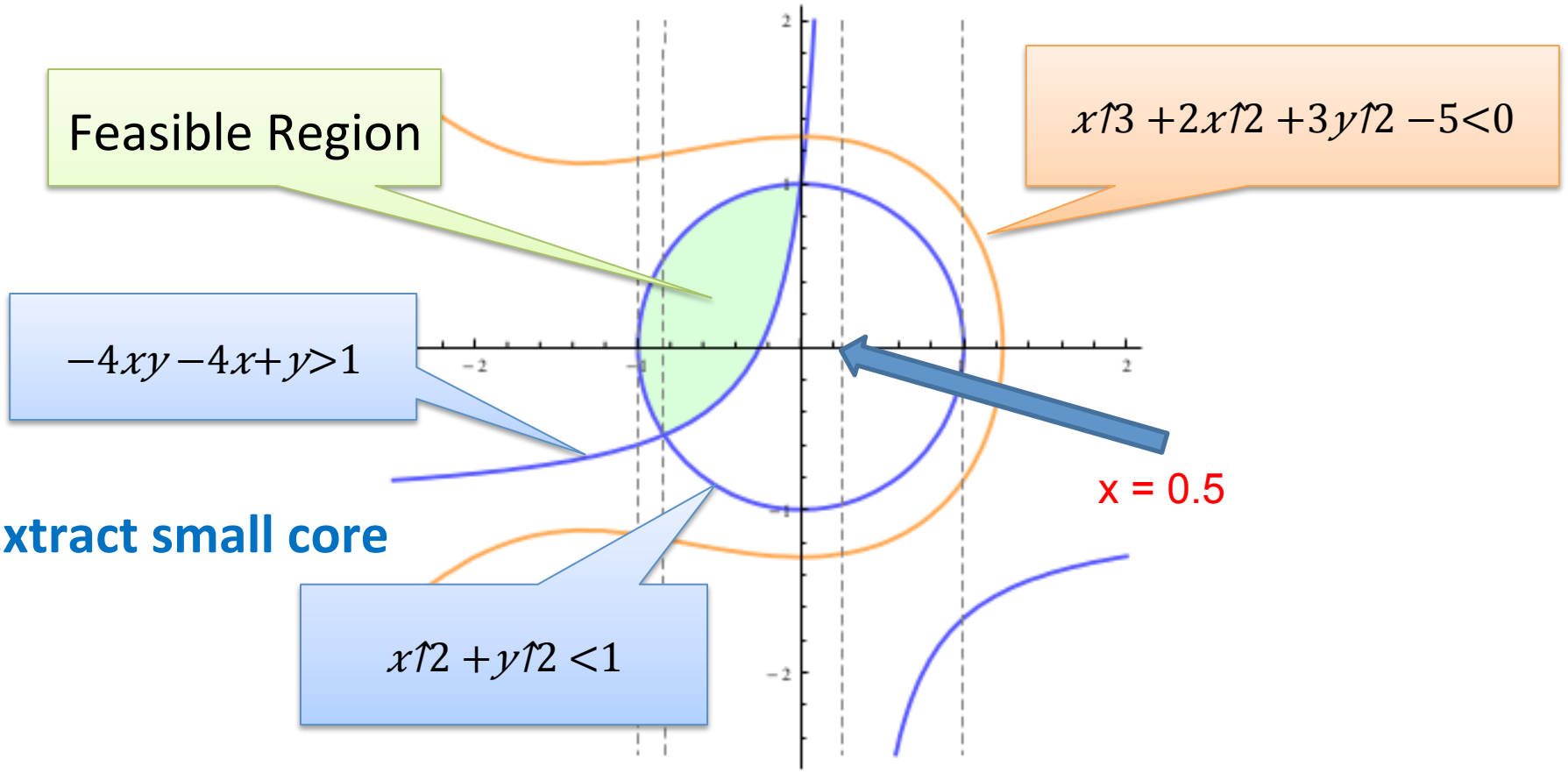


Z3 architecture - new



Solving ∃R Efficiently

A key idea: Use partial solution to guide the search



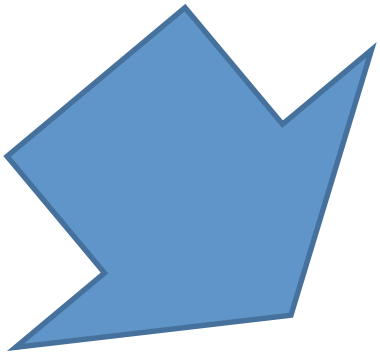
Dejan Jojanovich & Leonardo de Moura, IJCAR 2012

Horn Clause Satisfiability

mc(x) = x-10 **if x > 100**

mc(x) = mc(mc(x+11)) **if x ≤ 100**

assert (x ≤ 101 ⇒ mc(x) = 91)



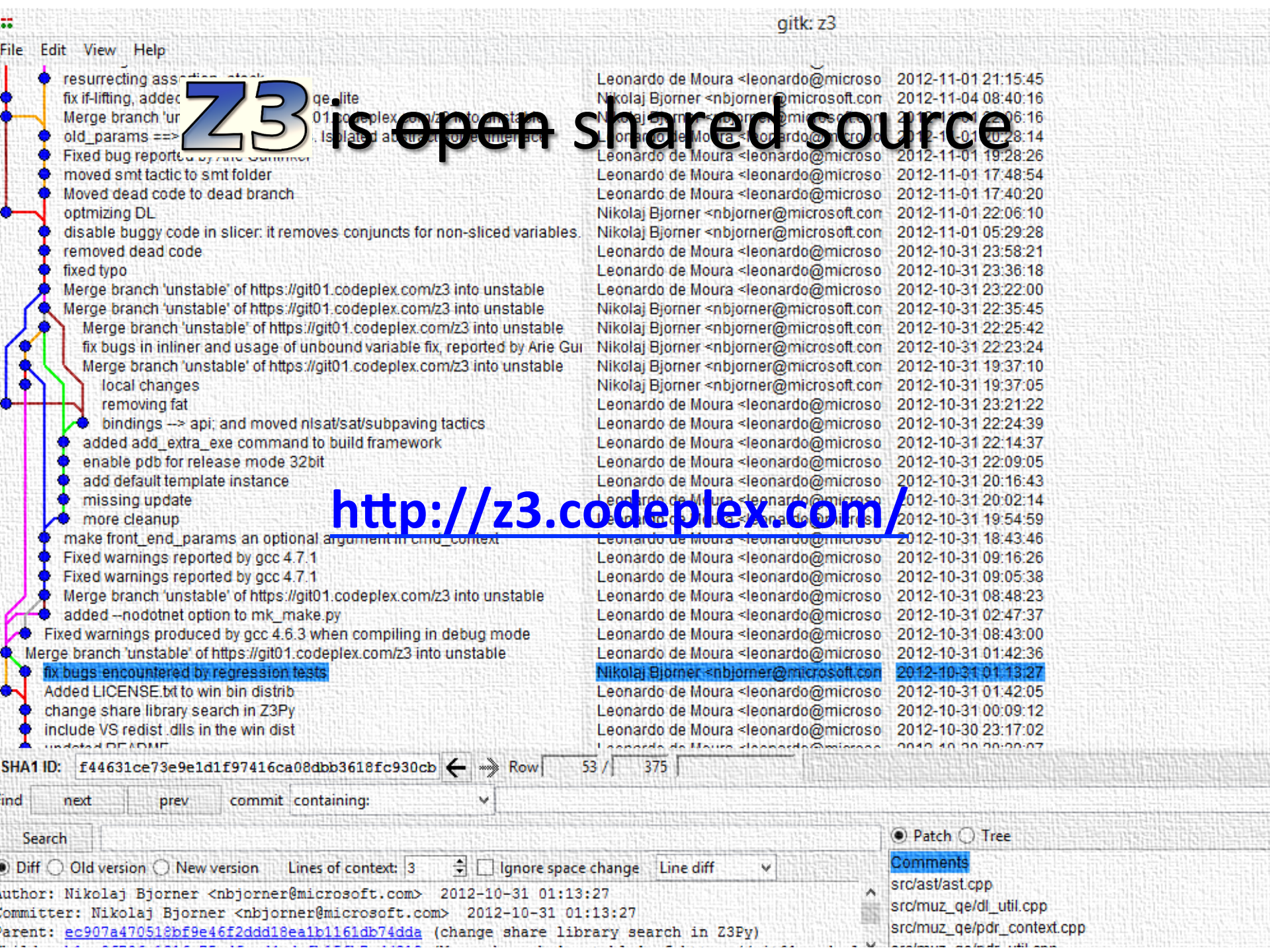
$$\forall X.\ X > 100 \rightarrow \mathrm{mc}(X, X-10)$$

$$\forall X, Y, R.\ X \leq 100 \wedge \mathrm{mc}(X+11, Y) \wedge \mathrm{mc}(Y, R) \rightarrow \mathrm{mc}(X, R)$$

$$\forall X, R.\ \mathrm{mc}(X, R) \wedge X \leq 101 \rightarrow R = 91$$

Krystof Hoder & Nikolaj Bjorner, SAT 2012

Bjorner, McMillan, Rybalchenko, SMT 2012


gitk: z3
File Edit View Help
Z3 is open shared source
http://z3.codeplex.com/
SHA1 ID: f44631ce73e9e1d1f97416ca08dbb3618fc930cb
Row 53 / 375
Find next prev commit containing:
Search
Diff Old version New version Lines of context: 3 Ignore space change Line diff
Patch Tree
Comments
src/ast/ast.cpp
src/muz_qe/dl_util.cpp
src/muz_qe/pdr_context.cpp
Author: Nikolaj Bjorner <nbjorner@microsoft.com> 2012-10-31 01:13:27
Committer: Nikolaj Bjorner <nbjorner@microsoft.com> 2012-10-31 01:13:27
Parent: ec907a470518bf9e46f2ddd18ea1b1161db74dda (change share library search in Z3Py)

Z3 source is (evil) subverting 8/

Munich, Germany -גר ב

הצג את הפרופיל המלא

Jan Wildeboer

Jan Wildeboer 3 באוק 2012 - גלוי לכולם

Microsoft trying to subvert Open Source. Not the first time: http://research.microsoft.com/en-us/um/people/leonardo/blog/2012/10/02/open-z3.html

"Z3 is now open source. The source code is available under the MSR-LA license. This is the same license used to distribute the Z3 binaries. The source code can be used for non-commercial purposes. "

Note the "non-commercial". This is not open source with such a limitation. It fits the political agenda of Microsoft however, where they have told politicians again and again the they think Open Source means non-commercial. But it simply isn't.

Open Source means Open Source. And that means no limitations on use and reuse.

« Open Sourcing Z3

Leonardo de Moura' Homepage

Validation

SecGuru: Automatic Validation of Network Connectivity Restrictions

Karthick Jayaraman, Charlie Kaufman,
and Ramanathan Venkatapathy

Nikolaj Bjørner

Network Policies:
Complexity, Challenge and Opportunity

Several devices, vendors, formats

- Net filters
- Firewalls
- Routers

Challenge in the field

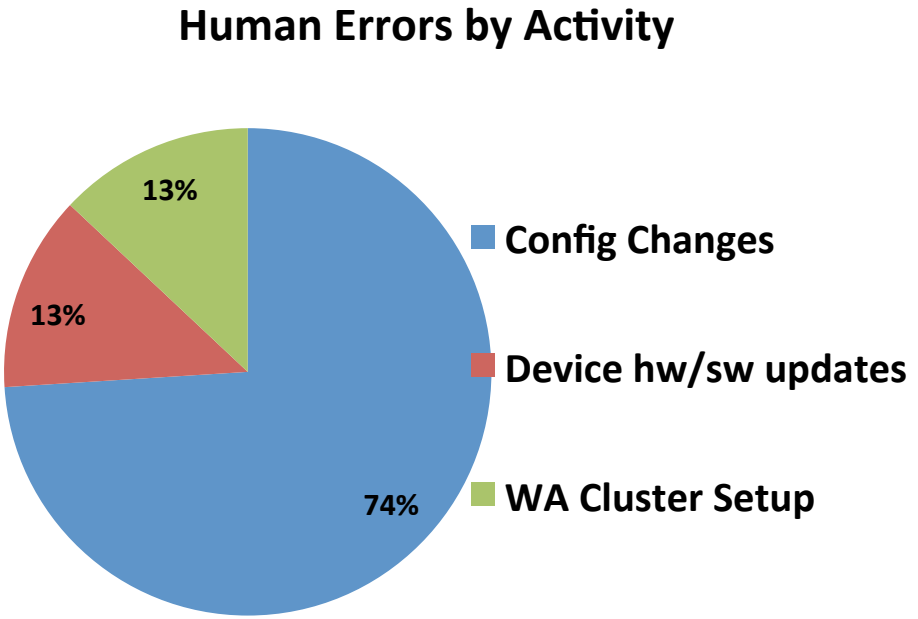
- Do devices enforce policy?
- Ripple effect of policy changes

Arcane

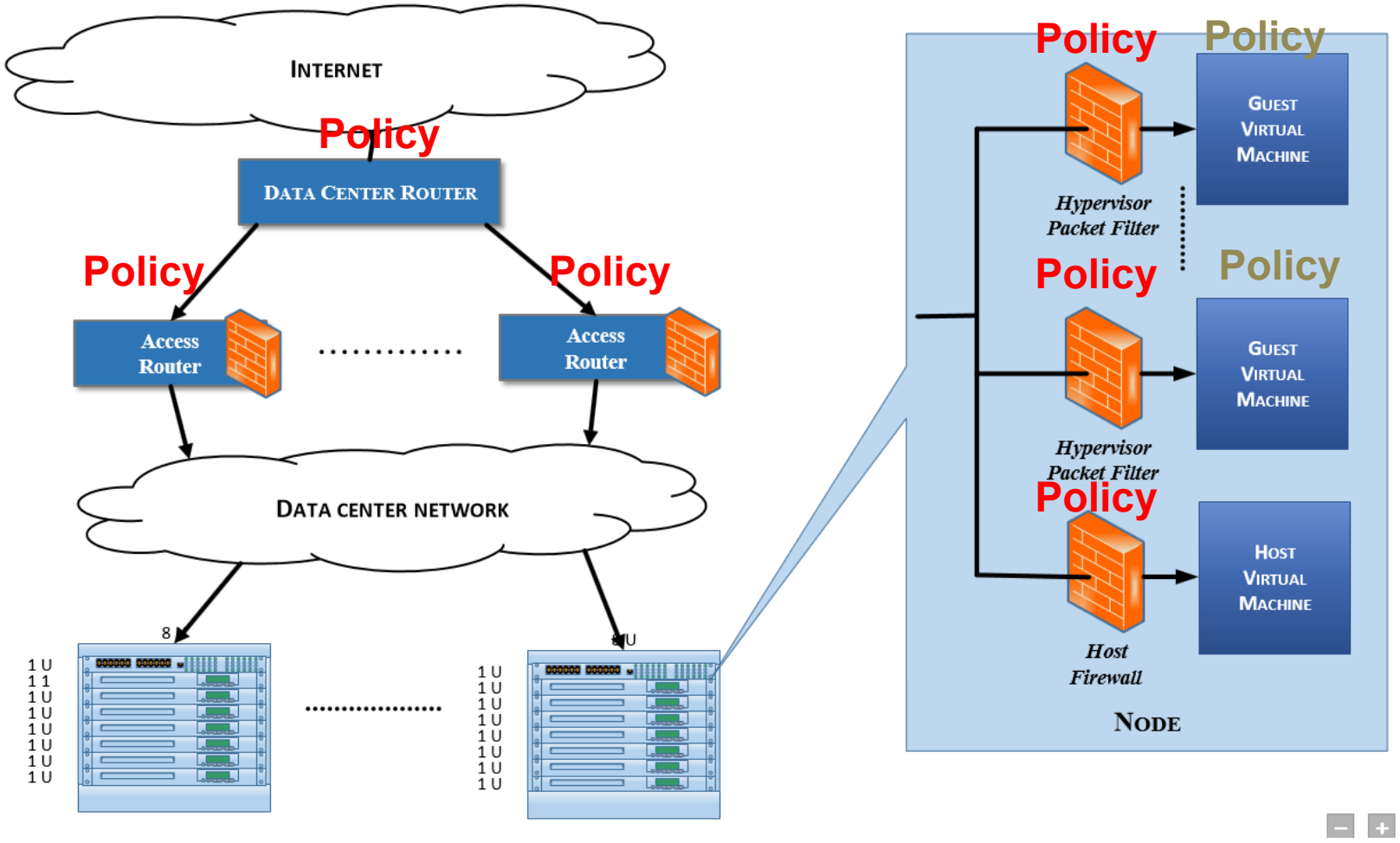
- Low-level configuration files
- Mostly manual effort
- Kept working by

"Masters of Complexity"

Human errors > 4 x DOS attacks



A Data-center Architecture



A Data-center Architecture

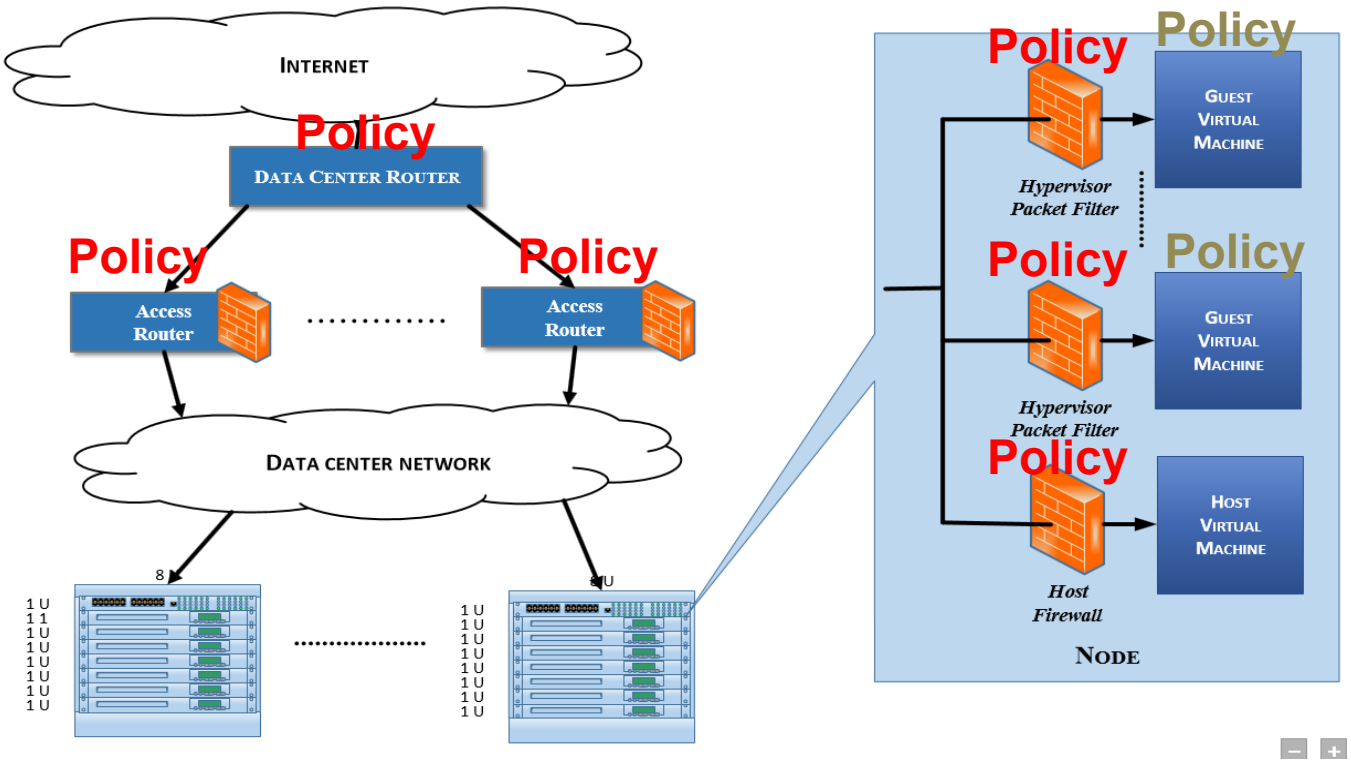
Defense in Depth = Crypto + Policies

Efficient and Flexible Defense by Policy only

Policies (Access Control Lists) matter

- outside IP can be spoofed

- inside IP cannot be spoofed



Network Policies
What they look like in routers

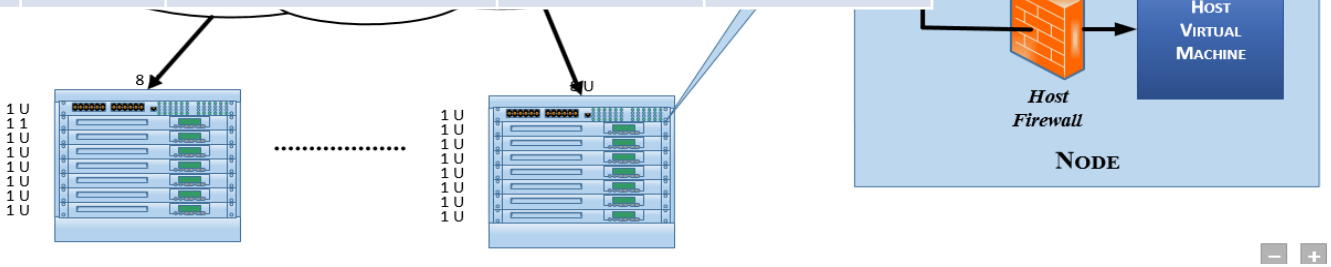
Defense in Depth = Crypto + Policies

Efficient and Flexible Defense by Policy only

- outside IP can be spoofed

- inside IP cannot be spoofed

Type	Src Address	Src Port	Remote Address	Remote Port	Protocol
Allow	10.20.0.0/19	*	157.55.252.0/22	*	6
Allow	10.20.0.0/19	*	157.55.252.0/22	*	17
Allow	10.20.0.0/19	*	157.55.252.0/22	*	4
Allow	10.20.0.0/19	*	157.55.252.0/22	*	1
Deny	*	*	65.52.244.0/22	*	4



A Need for Automating Policy Configuration

Constantly growing

New clusters – *"same"* policy, but different addresses

Constantly changing

Hardware – capacity, semantics
New Services – selectively exposed
Patches – to *live site incidents*

Bare metal low-level format of routers are also used for policy configurations

Towards automation:
Checking Policy Configuration with **SecGuru**

Constantly growing

New clusters – *enforce and check that new policies are instances of a master template (the intent)*

Constantly changing

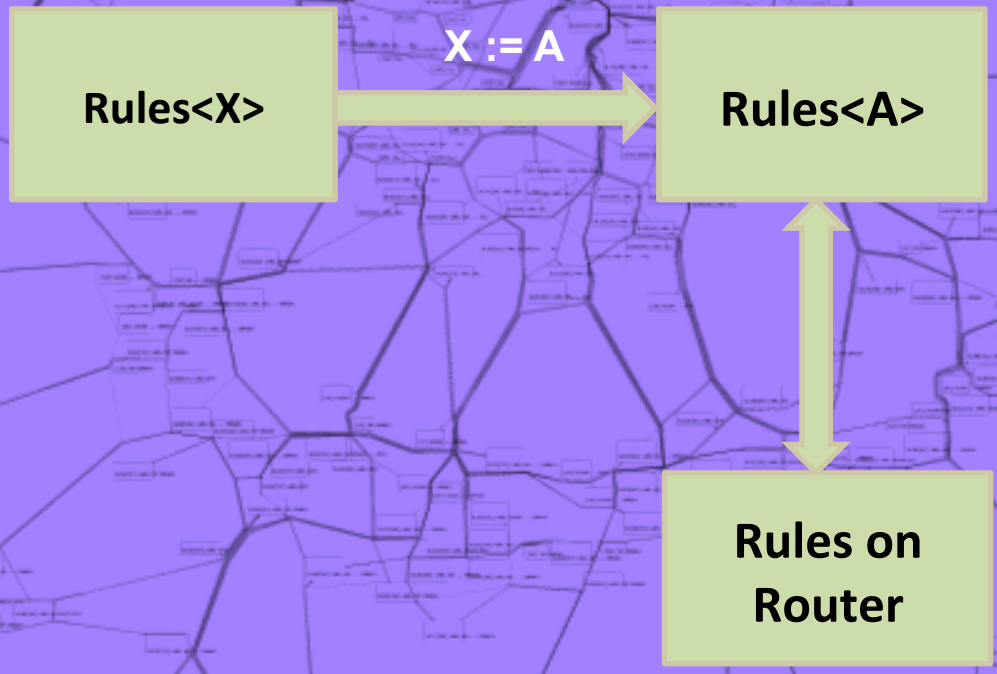
Hardware – capacity, semantics

New Services – *check effect of new rules*

Patches – *check for regressions*

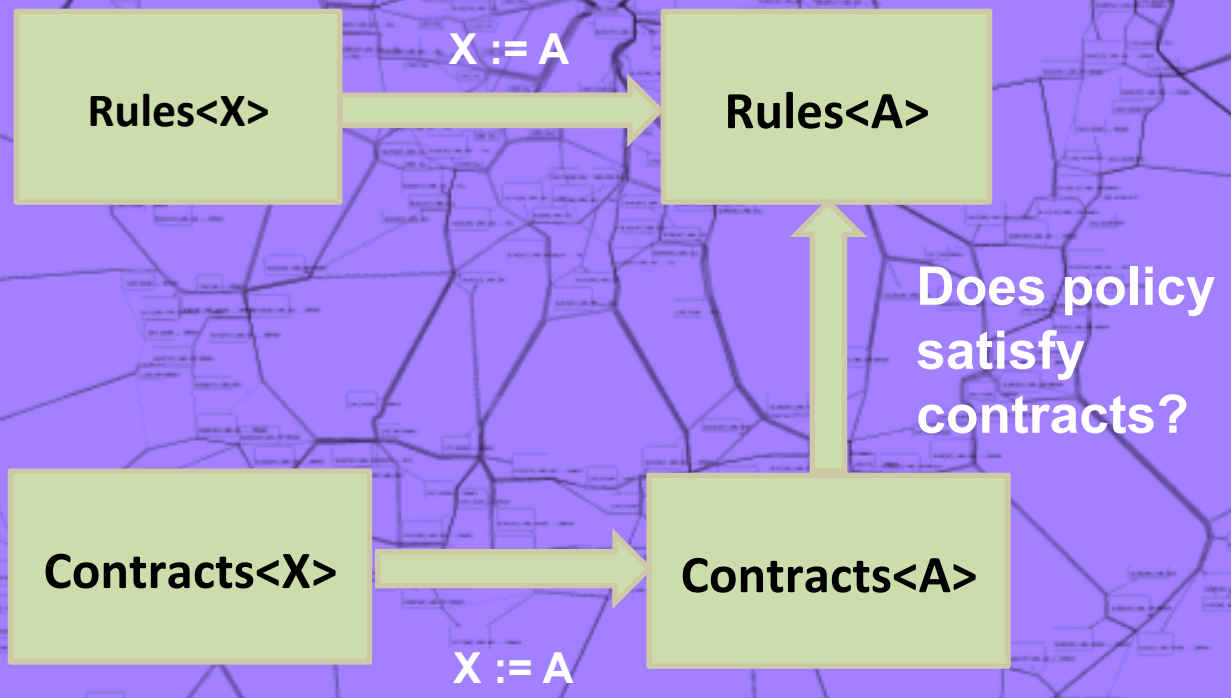
Towards automation:
Checking Policy Configuration with **SecGuru**

Enforce and check that policies are instances of a master template (the intent)



Towards automation:
Checking Policy Configuration with **SecGuru**

Enforce and check that policies satisfy contracts



Policies as Bit-Vector Formulas

Type	Src Address	Src Port	Remote Address	Remote Port	Protocol
Allow	10.20.0.0/19	*	157.55.252.0/22	*	6
Allow	10.20.0.0/19	*	157.55.252.0/22	*	17
Allow	10.20.0.0/19	*	157.55.252.0/22	*	4
Allow	10.20.0.0/19	*	157.55.252.0/22	*	1
Deny	*	*	65.52.244.0/22	*	4

IP, Port, and Protocol: bit vectors
Policy: Bit-vector logic

$$\blacksquare Allow:\& \blacksquare (10.20.0.0 \leq srcIp\ 10.20.31.255) \wedge (157.55.252.0 \leq dstIp \leq 157.55.252.255) \wedge$$

$$Policy : (Vi \uparrow \blacksquare Allow \downarrow i) \wedge (\wedge j \uparrow \blacksquare \neg Den$$

Policies as Bit-Vector Formulas

Type	Src Address	Src Port	Remote Address	Remote Port	Protocol
Allow	10.20.0.0/19	*	157.55.252.0/22	*	6
Allow	10.20.0.0/19	*	157.55.252.0/22	*	17
Allow	10.20.0.0/19	*	157.55.252.0/22	*	4
Allow	10.20.0.0/19	*	157.55.252.0/22	*	1
Deny	*	*	65.52.244.0/22	*	4

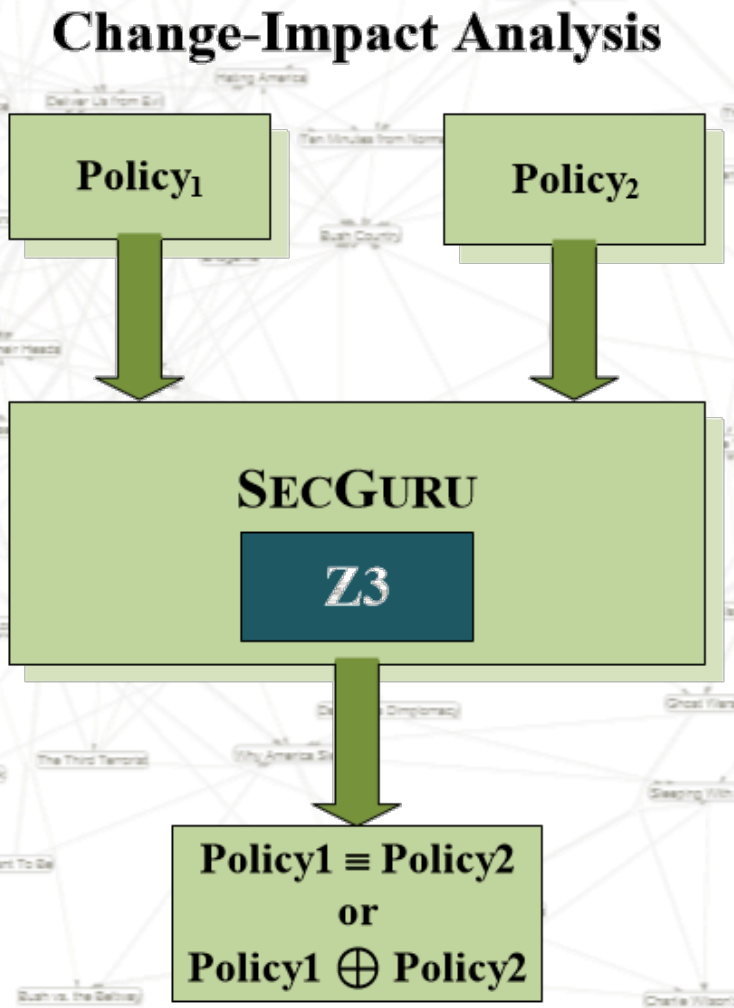
IP, Port, and Protocol: bit vectors
Policy: Bit-vector logic

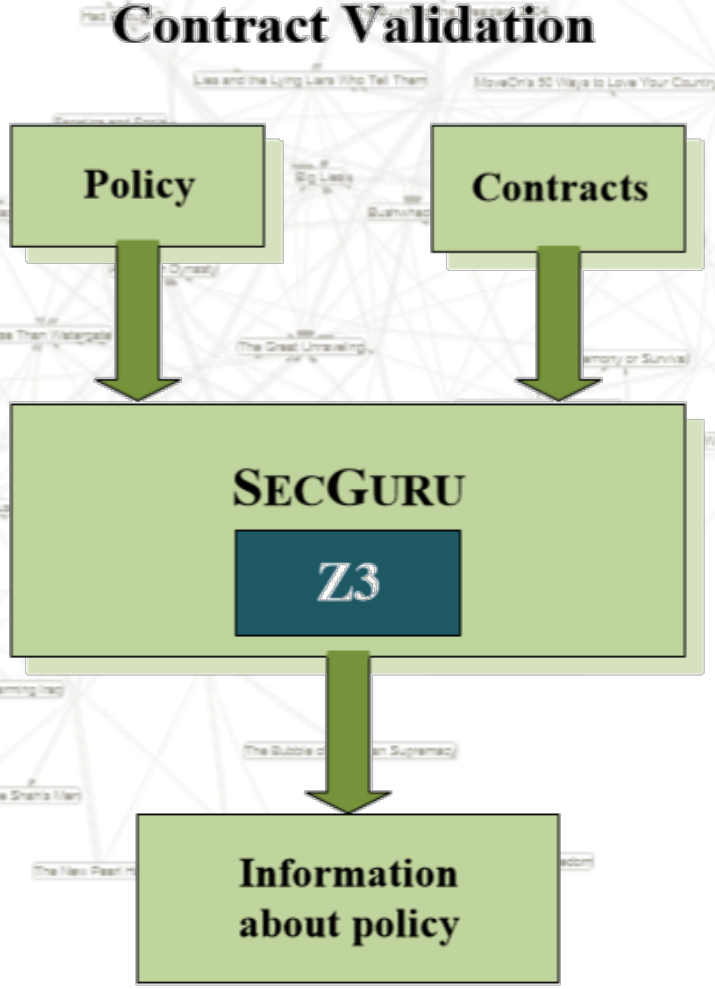
$$\blacksquare Allow{:}\& \blacksquare(10.20.0.0 \leq srcIp\ 10.20.31.255) \wedge (157.55.252.0 \leq dstIp \leq 157.55.252.255) \wedge$$

$$Policy\downarrow 0 = false$$

$$Policy\downarrow i+1 = if\ IsAllow\downarrow i$$

Two uses of **SecGuru**





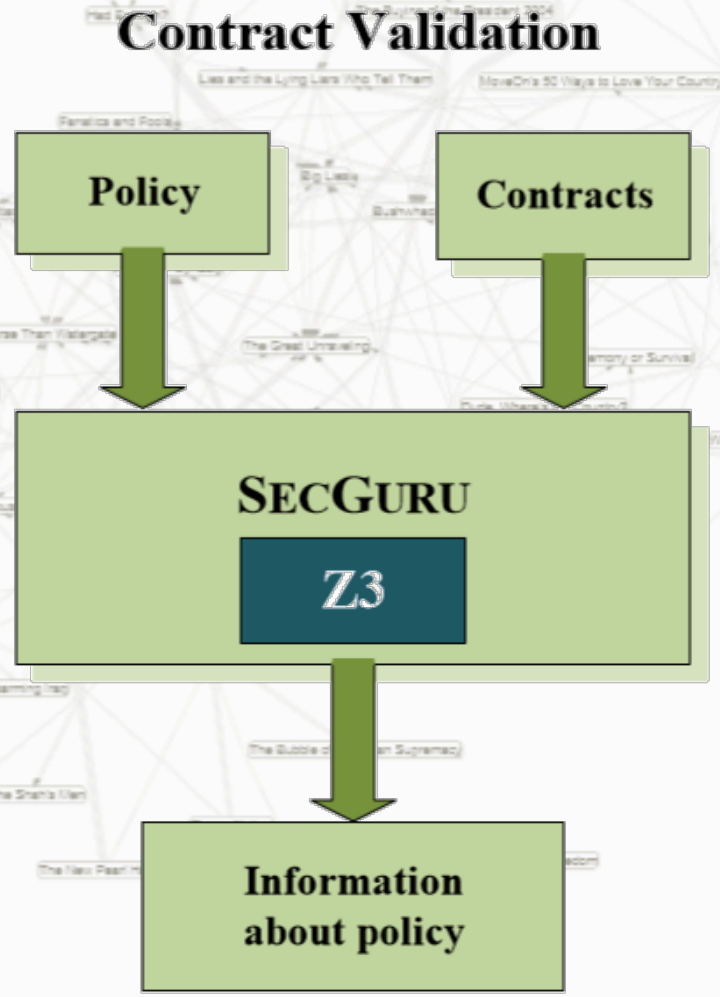
Two uses of **SecGuru**

*Does policy permit outgoing traffic to **some** address in 65.52.244.0/22?*

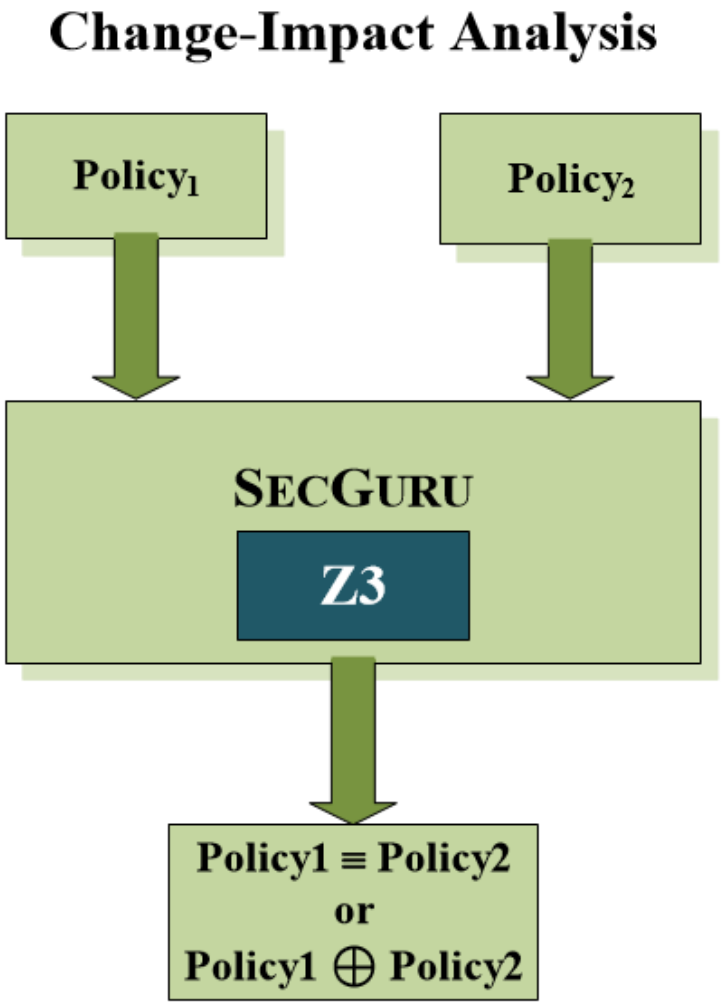
Check Satisfiability of

*Does policy permit connections to **all** the remote addresses in the range 65.52.244.0/22?*

Check Unsatisfiability of



Semantic Diff with **SecGuru**



Semantic diff between policies

Is $Policy{\downarrow}1 \equiv Policy{\downarrow}2$?

If not, print $Policy{\downarrow}1 \oplus Policy{\downarrow}2$

Traffic accepted by $P{\downarrow}1$, but not $P{\downarrow}2$. Models for $P{\downarrow}1 \wedge \neg P{\downarrow}2$

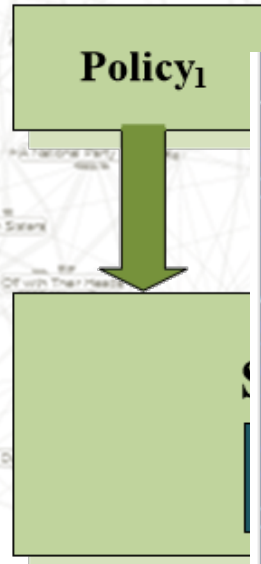
Traffic accepted by $P{\downarrow}2$, but not $P{\downarrow}1$.

Models for $P{\downarrow}2 \wedge \neg P{\downarrow}1$

Semantic Diff with **SecGuru**

Change-Impact Analysis

Semantic diff between policies




```
Traffic accepted by policy1, but not policy2
S.no, SrcIP, SrcPort, DstIP, DstPort, Protocol, TCP Flags, ICMP Type,
1, 10.25.252.0-10.25.252.255;, *;, 239.0.0.0-239.255.255.255;, *;, IP, ,
2, 10.62.12.0-10.62.12.255;, *;, 168.63.161.0-168.63.161.15;, 80;, TCP,
3, 10.62.12.0-10.62.12.255;, *;, 168.63.161.0-168.63.161.15;, 443;, TCP,
4, 10.62.12.0-10.62.12.255;, *;, 168.63.161.0-168.63.161.15;, 8080;, TCP
5, 10.146.192.0-10.146.192.255;, *;, 239.0.0.0-239.255.255.255;, *;, IP,
6, 210.126.9.11-210.126.9.12;, *;, 168.63.161.0-168.63.161.15;, 80;, TCP
7, 210.126.9.11-210.126.9.12;, *;, 168.63.161.0-168.63.161.15;, 443;, TC
8, 210.126.9.11-210.126.9.12;, *;, 168.63.161.0-168.63.161.15;, 8080;, T
9, 216.52.28.197;, *;, 168.63.161.0-168.63.161.15;, 80;, TCP, SYN, ,
10, 216.52.28.197;, *;, 168.63.161.0-168.63.161.15;, 443;, TCP, SYN, ,

Traffic accepted by policy2, but not policy1
S.no, SrcIP, SrcPort, DstIP, DstPort, Protocol, TCP Flags, ICMP Type,
1, 0.0.0.0-10.19.255.255;, *;, 168.63.161.0-168.63.161.15;, 19999;, TCP,
2, 1.202.140.226-1.202.140.227;, *;, 168.63.161.0-168.63.161.15;, 0-8549
3, 1.202.140.226-1.202.140.227;, *;, 168.63.161.0-168.63.161.15;, 8552-8
4, 1.202.140.226-1.202.140.227;, *;, 168.63.161.0-168.63.161.15;, 8571-6
5, 10.7.32.0-10.7.35.255;, *;, 168.63.161.0-168.63.161.15;, 0-8549;, TCP
6, 10.7.32.0-10.7.35.255;, *;, 168.63.161.0-168.63.161.15;, 8552-8569;,
7, 10.7.32.0-10.7.35.255;, *;, 168.63.161.0-168.63.161.15;, 8571-65535;,
8, 10.7.51.0-10.7.51.31;, *;, 168.63.161.0-168.63.161.15;, 0-79;, TCP, S
9, 10.7.51.0-10.7.51.31;, *;, 168.63.161.0-168.63.161.15;, 81-442;, TCP,
10, 10.7.51.0-10.7.51.31;, *;, 168.63.161.0-168.63.161.15;, 444-8079;, T
```

All-BVSAT: A compact model enumeration

Really naïve model enumeration:

- To generate the $(k+1)st$ model, negate all the k models seen so far
- $(Vi\uparrow\blacksquare Allow\downarrow i)\wedge(\wedge j\uparrow\blacksquare\neg Den y\downarrow j)\wedge(\wedge k\uparrow\blacksquare\neg Model\downarrow k)$ …
  $2\uparrow 32+16+32+16$ models

Smarter model enumeration in SecGuru using All-BVSAT (idea):

- Find initial $srcIp\downarrow 0$,$srcPort\downarrow 0 \vDash \boldsymbol{Policy\downarrow 1}\wedge\neg\boldsymbol{Policy\downarrow 2}$

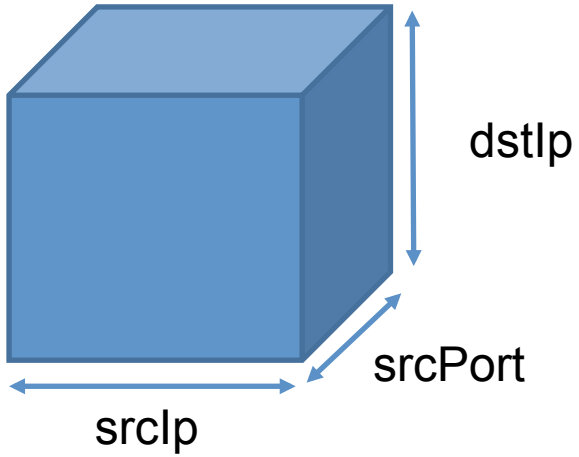
- Maximize bounds $lo \mid srcIp \leq srcIp \leq hi \mid srcIp$;

All-BVSAT: A compact model enumeration

Maximize bounds:

$$\blacksquare\blacksquare lo{\downarrow}srcIp \leq srcIp \leq hi{\downarrow}srcIp \wedge lo{\downarrow}srcPort \leq srcPort \leq hi{\downarrow}srcPort \wedge lo{\downarrow}dstIp \leq dstIp \leq hi{\downarrow}dstIp \;\&\models\& \boldsymbol{Policy{\downarrow}1} \wedge \neg \boldsymbol{Policy{\downarrow}2}$$

Result is a *cube:*

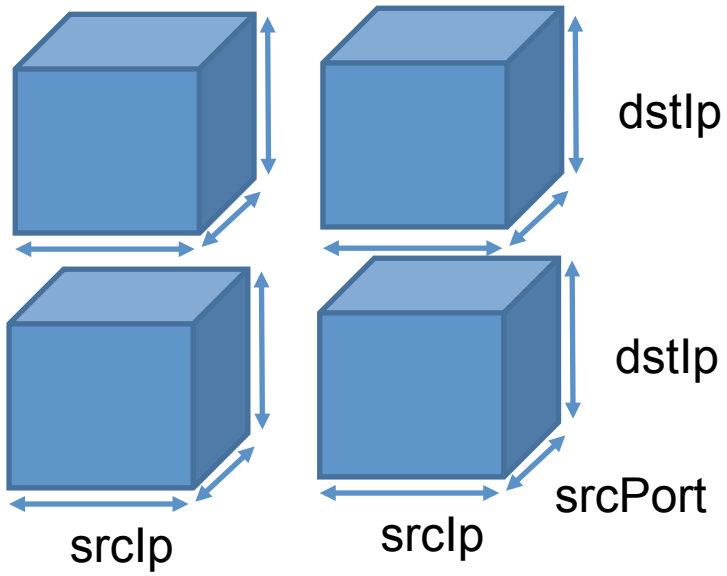


All-BVSAT: A compact model enumeration

More succinct: Maximize *multiple* bounds

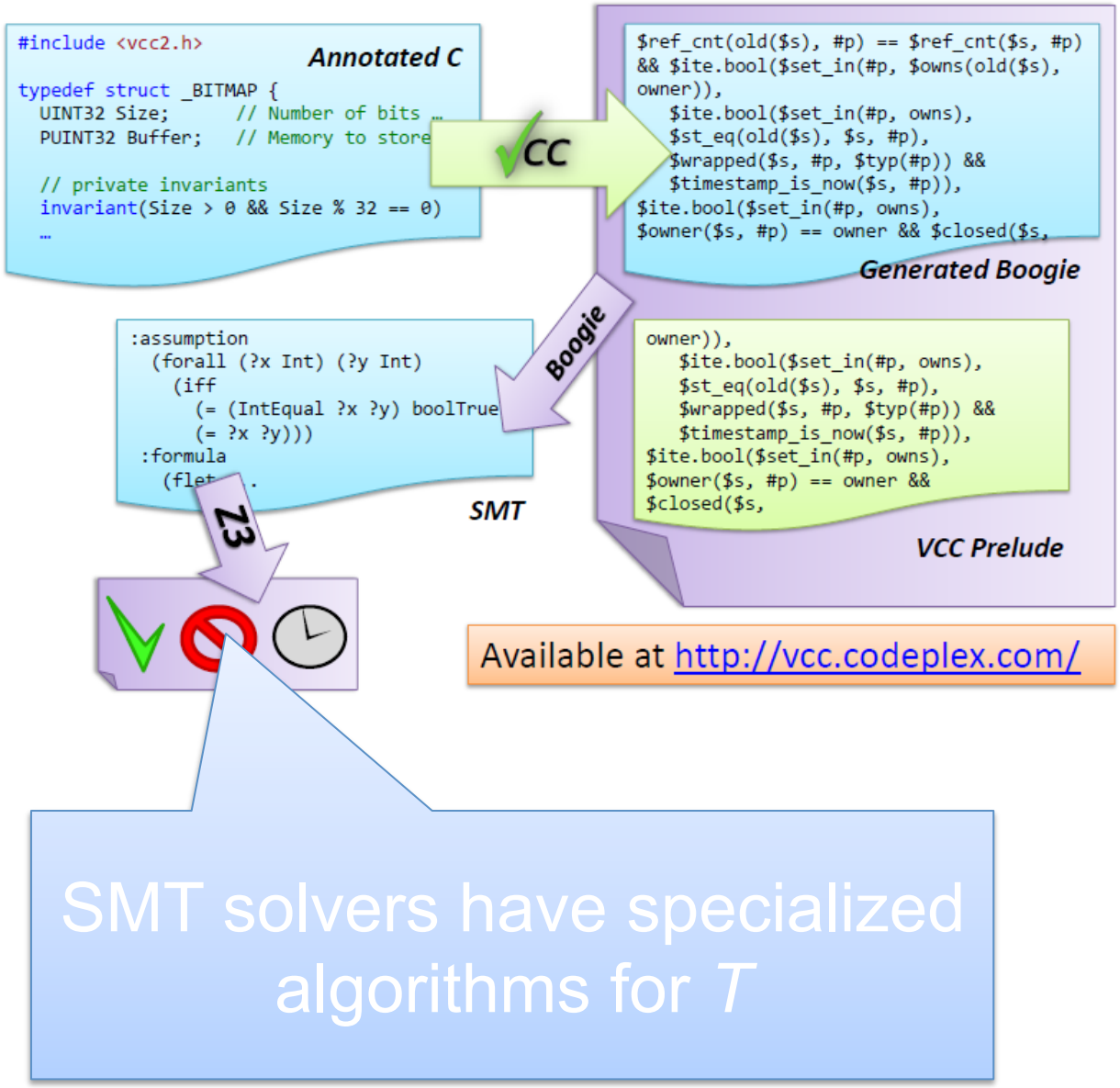
$$\blacksquare\blacksquare(lo{\downarrow}1 \le srcIp \le hi{\downarrow}2 \vee lo{\downarrow}2 \le srcIp \le hi{\downarrow}2) \wedge lo{\downarrow}srcPort \le srcPort \le hi{\downarrow}srcPort \wedge (lo{\downarrow}3 \le dstIp \le hi{\downarrow}3 \vee lo{\downarrow}4 \le dstIp \le hi{\downarrow}4) \;\&\models\& \boldsymbol{Policy{\downarrow}1} \wedge \neg \boldsymbol{Policy{\downarrow}2}$$

Result is a multi-*cube:*



Satisfiability of Horn Clauses

Divide and Conquer



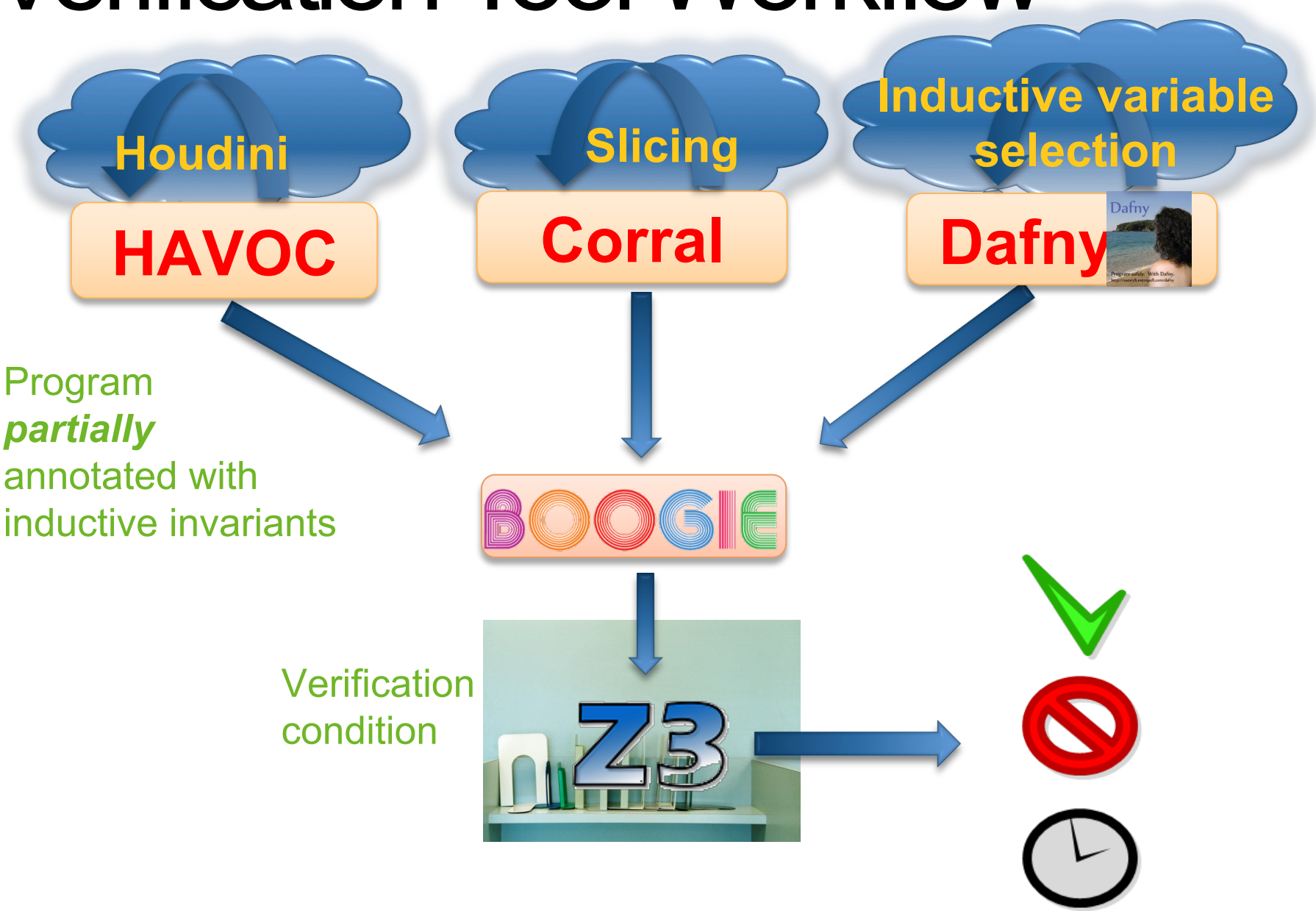
Front-end
Programming language semantics

VCC

Z3

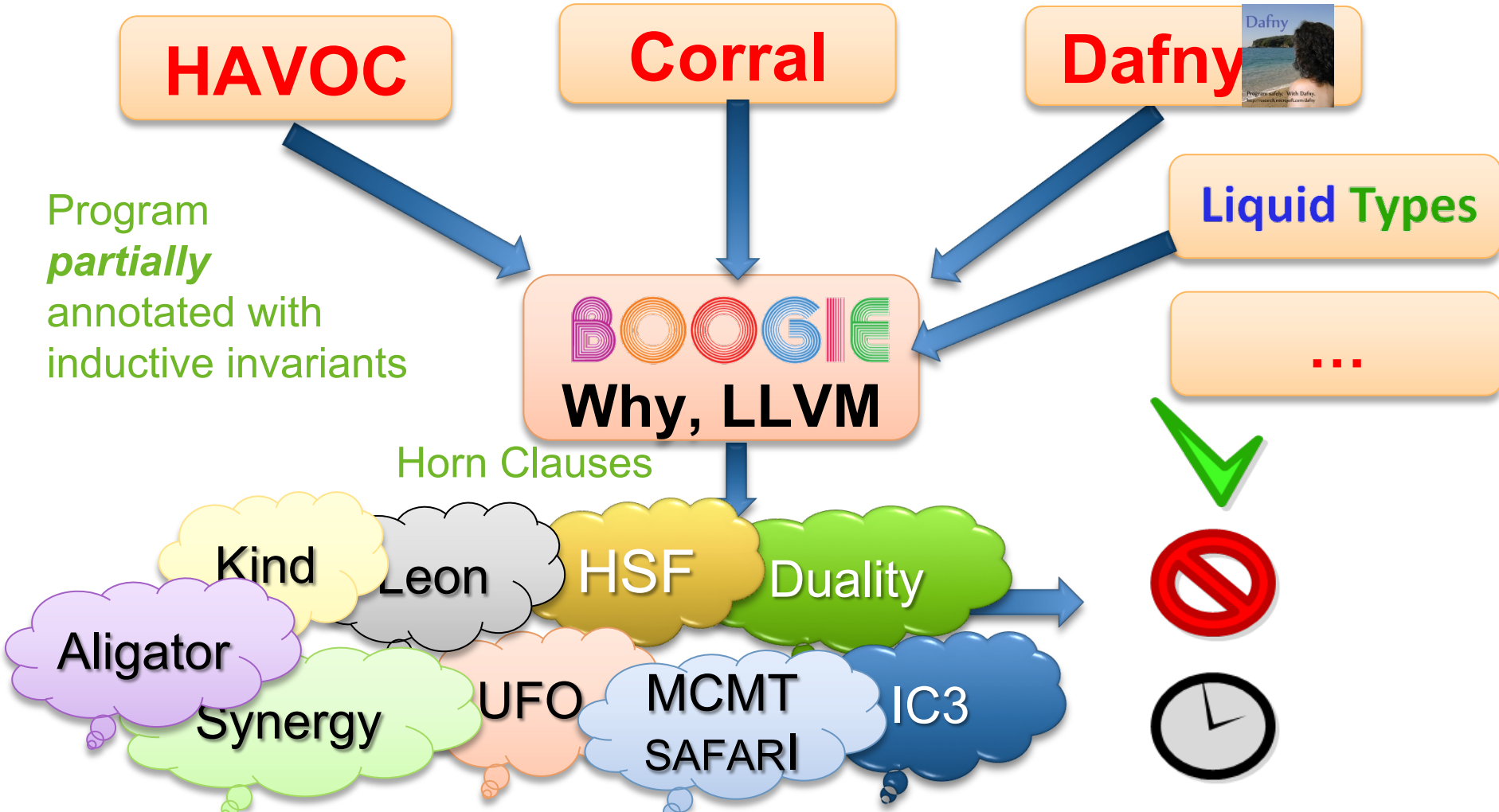
Back-end
Logic engine

Verification Tool Workflow



Envisioned: Verification Tool Workflow

Verification Condition Generators can already produce Horn Clauses



Motivation: Recursive Procedures

mc(x) = x-10 **if x > 100**

mc(x) = mc(mc(x+11)) **if x ≤ 100**

assert (mc(x) ≥ 91)

Motivation: Recursive Procedures

Formulate as Horn clauses:

$$\forall X.\ X > 100 \rightarrow \mathrm{mc}(X, X-10)$$

$$\forall X, Y, R.\ X \leq 100 \wedge \mathrm{mc}(X+11, Y) \wedge \mathrm{mc}(Y, R) \rightarrow \mathrm{mc}(X, R)$$

$$\forall X, R.\ \mathrm{mc}(X, R) \rightarrow R \geq 91$$

Solve for mc

Motivation: Recursive Procedures

Formulate as Predicate Transformer:

$$\blacksquare F\ (mc)(X,R) = \&\blacksquare X > 100 \wedge R = X - 10\ @\vee X \leq 100 \wedge \exists Y.\ mc(X+11,Y) \wedge mc(Y,R)$$

Check: $\mu F\ (mc)(X,R) \rightarrow R \geq 91$

Motivation: Recursive Procedures

Instead of computing $\mu F\ (mc)(X,R)$,

then checking $\mu F\ (mc)(X,R) \rightarrow R \geq 91$

Suffices to find post-fixed point $mc{\downarrow}\boldsymbol{post}$ **satisfying:**

$$\forall \boldsymbol{X},\boldsymbol{R}.\ \ F\ (mc{\downarrow}\boldsymbol{post}\ )(\boldsymbol{X},\boldsymbol{R}) \rightarrow mc{\downarrow}\boldsymbol{post}\ (\boldsymbol{X},\boldsymbol{R})$$

Program Verification as SMT

[Bjørner, McMillan, Rybalchenko, SMT workshop 2012]
Hilbert Sausage Factory: [Grebenshchikov, Lopes, Popeea, Rybalchenko, PLDI 2012]

Program Verification (Safety)

as ***Solving fixed-points***

as ***Satisfiability of Horn clauses***

Many problems expressible in SMT-LIB2

```
(set-logic HORN)
(declare-fun f (Int Int) Bool)

(assert (forall ((p Int)) (=> (> p 100) (f p (- p 10)))))
(assert (forall ((p Int) (p2 Int) (p3 Int) (p4 Int) (r Int))
   (=> (and (<= p 100) (f (+ p 31) p2)
            (f p2 p3) (f p3 p4) (f p4 r))
       (f p r))))


(assert (forall ((p Int) (r Int))
        (=> (and (f p r) (<= p (- 10)) (not (= r 91))) false)))
(check-sat)
```

```
Raw:T-----XEmacs: f4-pure.smt2      (Fundamental)----All----------------------------------------
```

```
C:\tvm\src\z3_2\regression\pdr>z3 f4-pure.smt2 /v:1
Entering level 1
Entering level 2
Entering level 3
Entering level 4
Entering level 5
Entering level 6
(define-fun f ((x!1 Int) (x!2 Int)) Bool
  (let ((a!1 (or (<= (+ x!2 (* (- 1) x!1)) (- 10)) (<= x!1 100))))
    (and (or (>= x!1 101) (<= x!2 91))
         a!1
         (<= (+ x!1 (* (- 1) x!2)) 10)
         (>= x!2 91))))
sat
```

Solvers for recursive Horn Clauses

- Several newer tools:
  - PDR in Z3 [Hoder, B] (more about this later)
  - Duality [McMillan]
  - HSF [Rybalchenko et.al.]
  - Eldarica [Rümmer, Hojjati, Kuncak]
  - SPACER [Komuravelli, Gurfinkel, Chaki, Clarke]

- Many cousins (indirectly solve Horn clauses)
  - FLATA, Corral, Yogi, UFO, Kind, Leon, Safari, MCMT

sv-benchmarks — Revision 213

/trunk/clauses/ALIA/sdv

[Parent Directory]

sdv0.smt2
sdv1.smt2
sdv10.smt2
sdv100.smt2
sdv1000.smt2
sdv1001.smt2
sdv1002.smt2
sdv1003.smt2
sdv1004.smt2
sdv1005.smt2
sdv1006.smt2
sdv1007.smt2
sdv1008.smt2
sdv1009.smt2
sdv101.smt2
sdv1010.smt2
sdv1011.smt2
sdv1012.smt2
sdv1013.smt2
sdv1014.smt2
sdv1015.smt2

IC3/PDR: Property Directed Reachability in Z3

The IC3 Algorithm for Symbolic Model Checking by Aaron Bradley

Procedures	*Regular vs. Push Down systems* As a Conflict-driven solver for recursive Horn clauses
Beyond Propositional Logic	*Linear Real Arithmetic* - *Timed Automata Decision Procedure* - *Interpolants from models*

[SAT 2012. Kryštof Hoder & Nikolaj Bjørner]

PDR – the algorithm

Objective is to solve for *R* such that

,

Key elements of PDR algorithm:

Over-approximate reachable states

Propagate back from
Resolve conflicts
Strengthen/propagate using induction

PDR – the algorithm

Objective is to solve for *R* such that

,

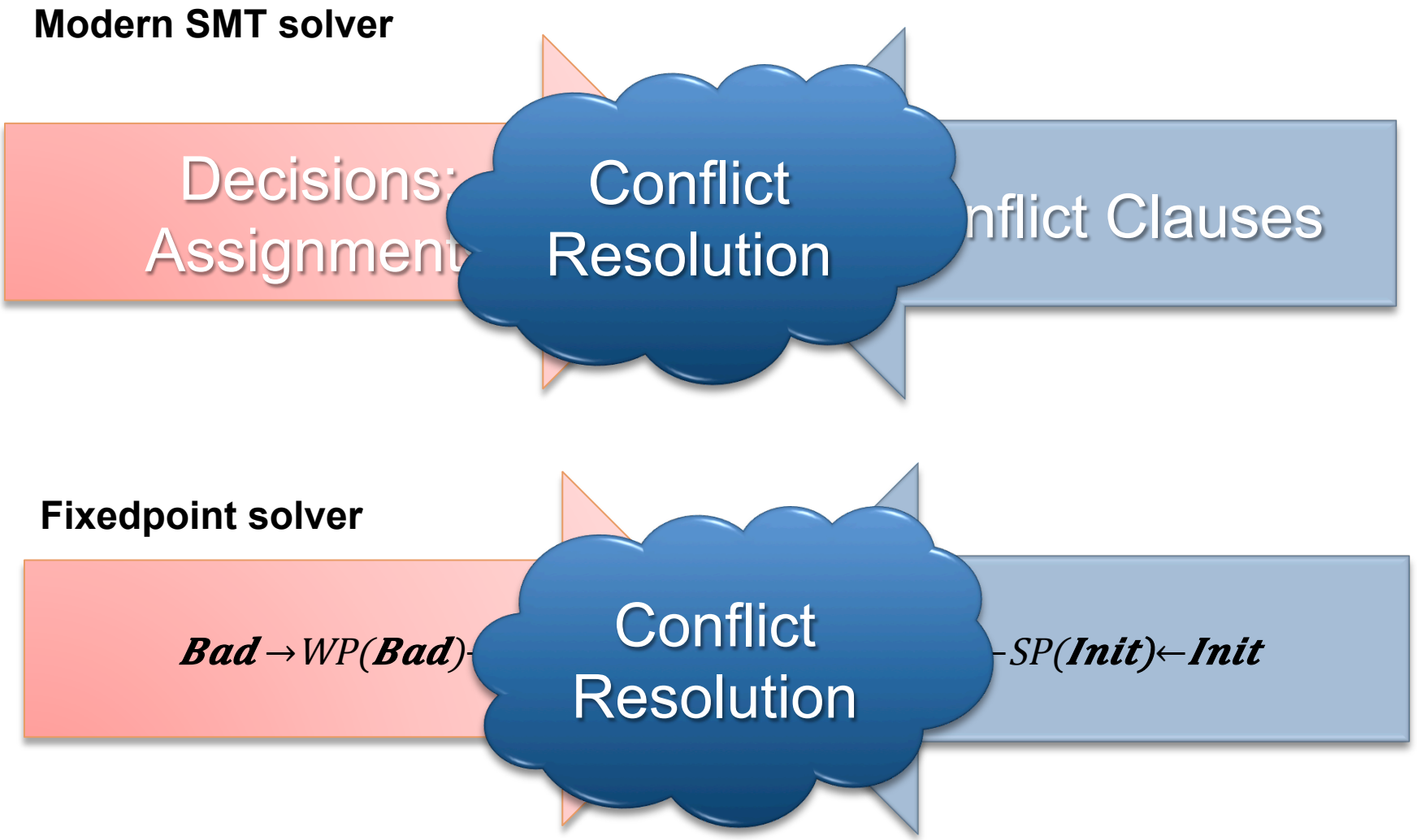
Initialize:

$$\blacksquare Safe \&\&\&\& R\downarrow 1 := true \&\&@\&\nwarrow \&\&\nearrow \&\&\nwarrow \&@\&\& R\downarrow 0 := F\ (fa$$

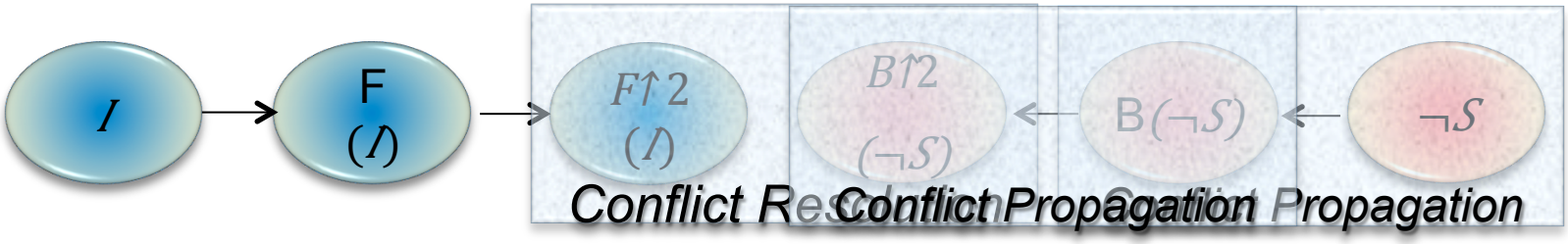
Main invariant:

$$\blacksquare Safe \&\&\&\& R\downarrow i+1\ \&\&@\&\nwarrow \&\&\nearrow\ \&\&\nwarrow\ \&@$$

Search: Mile-high perspective



Search: Mile-high perspective

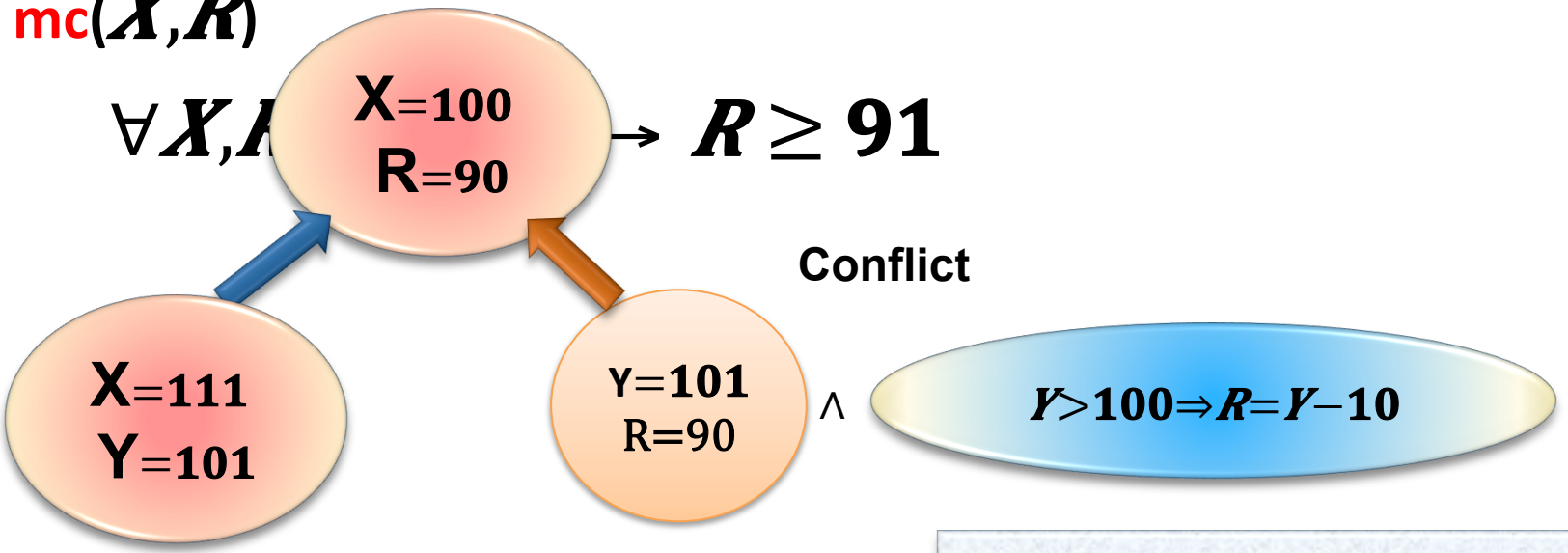


PDR(LRA): Conflict resolution

$$\forall X.\ X > 100 \rightarrow \mathrm{mc}(X, X-10)$$

$$\forall X,Y,R.\ X \leq 100 \wedge \mathrm{mc}(X+11, Y) \wedge \mathrm{mc}(Y, R) \rightarrow \mathrm{mc}(X, R)$$

$$\forall X, I \ldots \rightarrow R \geq 91$$



Resolution

Get Generalization from Farkas Lemma

Conflict resolution with arithmetic

$$\textbf{initially } y_1 := y_2 := 0;$$
$$P_1 :: \left[\left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_1 := y_2 + 1;\\ \ell_1 : \textbf{await } y_2 = 0 \vee y_1 \leq y_2;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_1 := 0;\end{array}\right]\end{array}\right]\right] \;\|\; P_2 :: \left[\left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_2 := y_1 + 1;\\ \ell_1 : \textbf{await } y_1 = 0 \vee y_2 \leq y_1;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_2 := 0;\end{array}\right]\end{array}\right]\right]$$

R(0,0,0,0).
T(L,M,Y1,Y2,L',M',Y1',Y2')∧R(L,M,Y1,Y2) → R(L',M
R(2,2,Y1,Y2) → false

Step(L,L',Y1,Y2,Y1') → T(L,M,Y1,Y2,L',M,Y1',Y2)
Step(M,M',Y2,Y1,Y2') → T(L,M,Y1,Y2,L,M',Y1,Y2')

Step(0,1,Y1,Y2,Y2+1).
(Y1 ≤ Y2 ∨ Y2 = 0) → Step(1,2,Y1,Y2,Y1).
Step(2,3,Y1,Y2,Y1).
Step(3,0,Y1,Y2,0).

Mutual Exclusion

⇔

Clauses have model

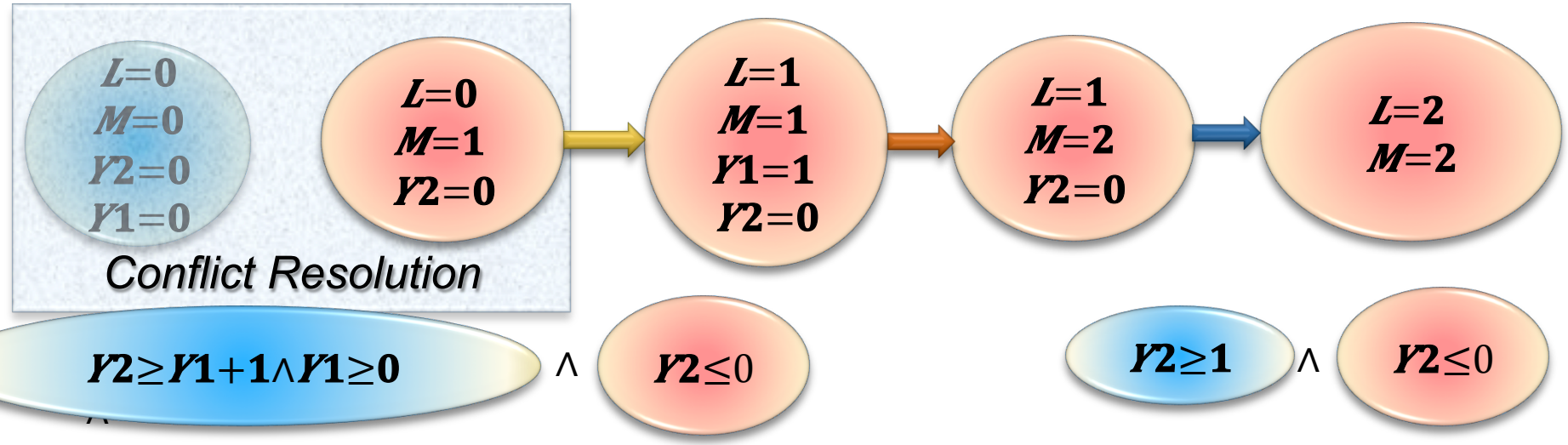
P_2 takes a step

$\ell\downarrow 0 : y := y + 1; \textit{goto}\ \ell\downarrow 1$
$\ell\downarrow 1 : \textit{await}\ y = 0 \vee y \leq y\ ; \textit{goto}\ \ell\downarrow 2$
$\ell\downarrow 2 : \textit{critical}; \textit{goto}\ \ell\downarrow 3$
$\ell\downarrow 3 : y := 0; \textit{goto}\ \ell\downarrow 0$

PDR(T): Conflict Resolution

initially $y_1 := y_2 := 0$;

$$P_1 :: \left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_1 := y_2 + 1;\\ \ell_1 : \textbf{await } y_2 = 0 \vee y_1 \leq y_2;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_1 := 0;\end{array}\right]\end{array}\right] \;\|\; P_2 :: \left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_2 := y_1 + 1;\\ \ell_1 : \textbf{await } y_1 = 0 \vee y_2 \leq y_1;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_2 := 0;\end{array}\right]\end{array}\right]$$



Conflict

Resolution

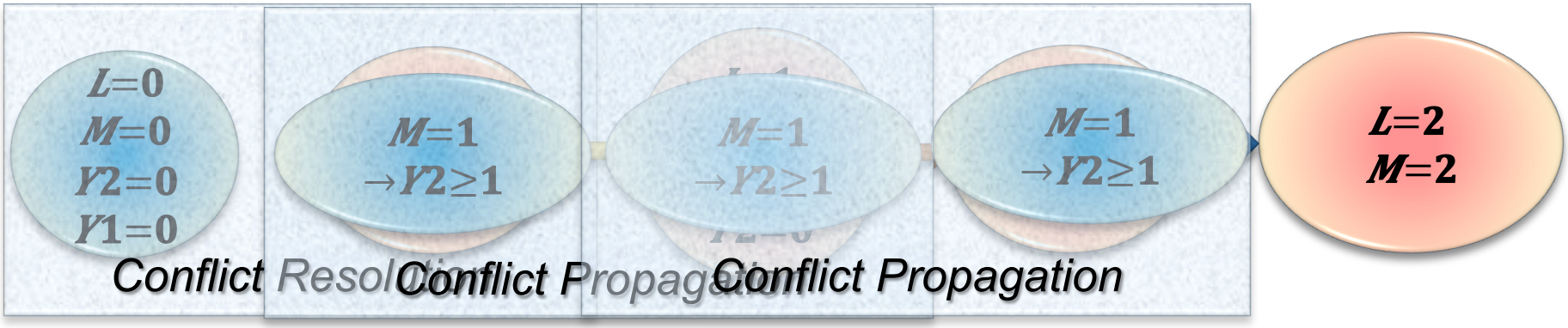
Get Generalization from Farkas Lemma

Eg., resolve away blue internal variables

PDR(T): Conflict Resolution

initially $y_1 := y_2 := 0$;

$$P_1 :: \left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_1 := y_2 + 1;\\ \ell_1 : \textbf{await } y_2 = 0 \vee y_1 \leq y_2;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_1 := 0;\end{array}\right]\end{array}\right] \;\|\; P_2 :: \left[\begin{array}{l}\textbf{loop forever do}\\ \left[\begin{array}{l}\ell_0 : y_2 := y_1 + 1;\\ \ell_1 : \textbf{await } y_1 = 0 \vee y_2 \leq y_1;\\ \ell_2 : \textbf{critical};\\ \ell_3 : y_2 := 0;\end{array}\right]\end{array}\right]$$



PDR(T): Generalization from T-lemmas

Can we satisfy?

$R(0,0,0,0)$. ***Initial states***

$T(L,M,Y1,Y2,L',M',Y1',Y2'), R(L,M,Y1,Y2) \vee R(L',M',Y1',Y2')$ ***Reachable states***

$R(L,M,Y1,Y2) \vee \neg(L=2 \wedge M=2)$. ***Unsafe state is unreachable***

$\sqcup L=0 \wedge M=1 \wedge Y2=0 \sqsubset M \wedge \sqcup F\ (R{\downarrow}0\ ) \sqsubset Pre$ **is unsatisfiable**

E.g., there is unsat core of: $\sqcup \wedge j{\uparrow}\blacksquare c{\downarrow}j \le x{\downarrow}j \le c{\downarrow}j\ \sqsubset M\ \wedge \sqcup F\ (R{\downarrow}i\ ) \sqsubset Pre$

Unsat proof uses T-lemmas

$(\sqcup 5 > x{\downarrow}1 \vee 3 < x{\downarrow}3\ \sqsubset From \neg M \vee \sqcup x{\downarrow}1 - x{\downarrow}2 > 2 \vee 2x{\downarrow}2 - x{\downarrow}3 > 1\ \sqsubset From \neg Pre\ )$

PDR(T): Generalization from T-lemmas

Can we satisfy?

$R(0,0,0,0)$. ***Initial states***

$T(L,M,Y1,Y2,L',M',Y1',Y2'), R(L,M,Y1,Y2) \Rightarrow R(L',M',Y1',Y2')$ ***Reachable states***

$R(L,M,Y1,Y2) \Rightarrow \neg(L=2 \wedge M=2)$. ***Unsafe state is unreachable***

Unsat proof uses T-lemmas

$(\_5 > x\downarrow 1 \vee 3 < x\downarrow 3 \ \top \mathit{From} \neg M \vee \_\ x\downarrow 1 - x\downarrow 2 > 2 \vee 2x\downarrow 2 - x\downarrow 3 > 1 \ \top \mathit{From} \neg \mathit{Pre})$

$\blacksquare 2 \cdot (-x\downarrow 1 \leq -5) @ x\downarrow 3 \leq 3 @ 2 \cdot (x\downarrow 1 - x\downarrow 2 \leq 2) @ 2x\downarrow 2 - x\downarrow 3 \leq 1 @$
$\blacksquare -2x\downarrow 1 \leq -10 @ x\downarrow 3 \leq 3 @ 2x\downarrow 1 - 2x\downarrow 2 \leq 4 @ \blacksquare 2x\downarrow 2 - x\downarrow 3 \leq 1 - - - - - - @ 0 \leq -2$ $\_2x\downarrow 1 - x\downarrow 3 \leq 5 \ \top \blacksquare$ **Block any model** $\mathit{satisfying}$ **this** $\downarrow$

PDR(LRA): Timed automata

Observation:

PDR + Model refinement using Farkas strengthening is a decision procedure for timed push-down systems

Justification:

Every lemma produced is a sum of differences from the input

~

Acyclic path in difference graph.

⇒ Finite set of Farkas lemmas possible.

N+1 degrees of separation

Objective: Synthesize inductive invariant to prove property.

Reaching objective with interpolants:

Synthesize interpolants, use for proving invariants.	Admirable
Synthesize interpolants, evaluate on random formulas.	Admire them
Write papers about theory of interpolants.	Admire the theorems
Review papers about generating interpolants.	Write admirable review
Participate in PC discussion about paper.	Admire Kevin Bacon
Write paper about proof transformations, maybe good for interpolation.	Who knows, it may be used?

Reaching objective with PDR:

…. Nevertheless, interpolants arise.

Side-effect: Horn Craig Interpolants

Suppose $A \Rightarrow B$

A Craig Interpolant is formula I:

$$Lang(I) \subseteq Lang(A) \cap Lang(B)$$

$$A \Rightarrow I,\ I \Rightarrow B$$

Horn version. Establish satisfiability of:

$$\forall x,y.\ A[x,y] \Rightarrow I(x),$$

$$\forall x,z.\ I(x) \Rightarrow B[x,z]$$

and find solution for I.

Side-effect: Horn Craig Interpolants

Suppose $A \Rightarrow B$

A Craig Interpolant is formula I:

$$Lang(I) \subseteq Lang(A) \cap Lang(B)$$

$$A \Rightarrow I,\ I \Rightarrow B$$

$\forall X.\quad X > 100 \rightarrow \mathbf{mc_0}(X, X-10)$

$\forall X,Y,R.\quad X \leq 100 \wedge \mathbf{mc_0}(X+11, Y) \wedge \mathbf{mc_0}(Y, R) \rightarrow \mathbf{mc_1}(X, R)$

$\forall X.\quad X > 100 \rightarrow \mathbf{mc_1}(X, X-10)$

$\forall X,Y,R.\quad X \leq 100 \wedge \mathbf{mc_1}(X+11, Y) \wedge \mathbf{mc_1}(Y, R) \rightarrow \mathbf{mc_2}(X, R)$

$\forall X.\quad X > 100 \rightarrow \mathbf{mc_2}(X, X-10)$

$\forall X,R.\quad \mathbf{mc_2}(X, R) \rightarrow R \geq 91$

Solve for $\mathbf{mc_0}, \mathbf{mc_1}, \mathbf{mc_2}$

PDR(T): Interpolants as a side-effect

Intermediary solutions:

$$\forall \boldsymbol{X}.\ F\ (\boldsymbol{R}\downarrow\mathbf{0}\ )(\boldsymbol{X}) \rightarrow \boldsymbol{R}\downarrow\mathbf{1}\ (\boldsymbol{X}),$$
$$\forall \boldsymbol{X}.\ F\ (\boldsymbol{R}\downarrow\mathbf{1}\ )(\boldsymbol{X}) \rightarrow \boldsymbol{R}\downarrow\mathbf{2}\ (\boldsymbol{X}),$$
$$\forall \boldsymbol{X}.\ F\ (\boldsymbol{R}\downarrow\mathbf{2}\ )(\boldsymbol{X}) \rightarrow \boldsymbol{Safe}(\boldsymbol{X}),$$

Observation: ***Farkas strengthening computes a "Horn Interpolant" for LRA***

i.e., solves for non-recursive Horn clauses

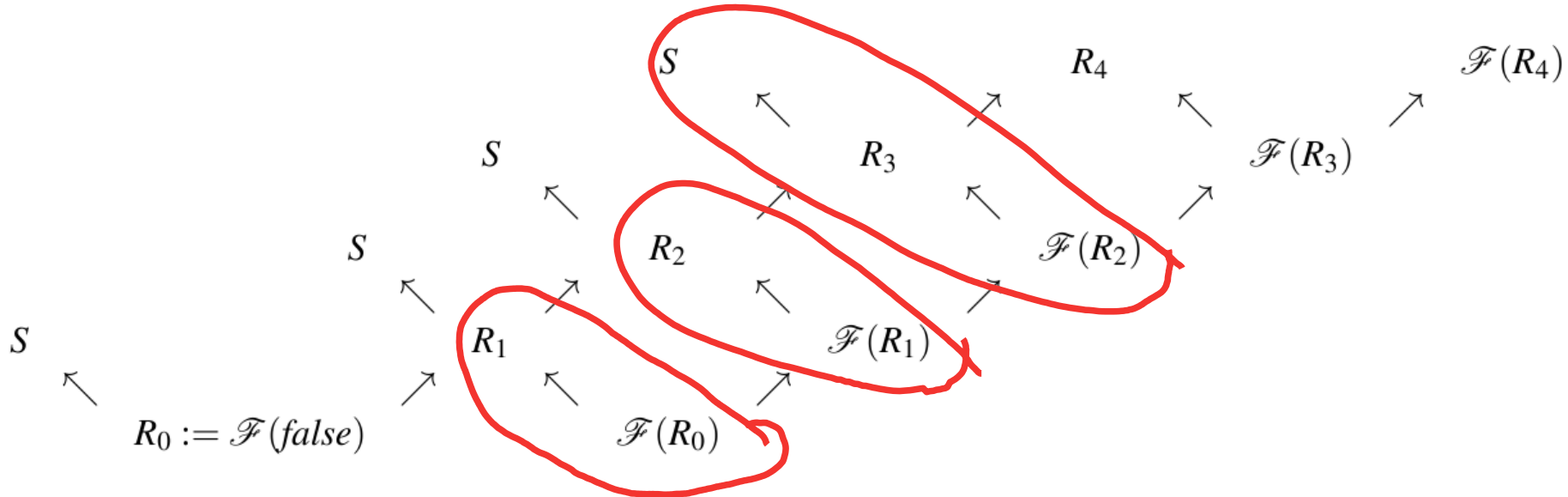
PDR(T): Interpolants as a side-effect

Intermediary solutions:

$$\forall X.\ \mathscr{F}(R_0)(X) \rightarrow R_1(X),$$
$$\forall X.\ \mathscr{F}(R_1)(X) \rightarrow R_2(X),$$
$$\forall X.\ \mathscr{F}(R_2)(X) \rightarrow Safe(X),$$

Observation: ***Farkas strengthening computes a "Horn Interpolant" for LRA***

i.e., solves for non-recursive Horn clauses



Universally Quantified Horn Clauses

```
void init(int n, int* A, inc c) {
    for (int i = 0; i < n; ++i) {
        A[i] = c;
    }
    assert(∀j. 0≤j<n⇒A[j]=c);
}
```

Find R to satisfy:

$$\forall c,n,A. \quad R(0,c,n,A).$$
$$\forall i,c,n,A. \quad R(i,c,n,A) \Rightarrow R(i+1,c,n,A[i \mapsto c])$$
$$\forall i,c,n,A,j. \quad R(i,c,n,A) \wedge 0 \leq j < n = i \Rightarrow A[j] = c$$

Problem: Inductive invariant is **quantified**

$$R(i,c,n,A) \equiv \forall j.\ 0 \leq j < i \Rightarrow A[j] = c$$

Horn clause solvers compute *quantifier-free* symbolic solutions.

Universally Quantified Horn Clauses

```
void init(int n, int* A, inc c) {
    for (int i = 0; i < n; ++i) {
        A[i] = c;
    }
    assert(∀j. 0≤j<n⇒A[j]=c);
}
```

Find instead Q to satisfy:

$$\forall c,n,A. \quad \forall k.Q(0,c,n,A[k],k).$$
$$\forall i,c,n,A,k. \quad (\forall k.Q(i,c,n,A[k],k))\Rightarrow Q(i+1,c,n,A[i\mapsto c][k],k)$$
$$\forall i,c,n,A,j. \quad (\forall k.Q(i,c,n,A[k],k))\wedge 0\leq j< n=i\Rightarrow A[j]=c$$

Problem: Horn clause solvers don't handle $\forall k.Q(..)$ formulas

Universally Quantified Horn Clauses

```
void init(int n, int* A, inc c) {
    for (int i = 0; i < n; ++i) {
        A[i] = c;
    }
    assert(∀j. 0≤j<n⇒A[j]=c);
}
```

SAS 2013 paper approach: Pre-instantiate $\forall k.\ Q$ using pattern matching

$$\forall c,n,A,k. \quad Q(0,c,n,A[k],k).$$
$$\forall i,c,n,A,k. \quad Q(i,c,n,A[k],k) \Rightarrow Q(i+1,c,n,A[i \mapsto c][k],k)$$
$$\forall i,c,n,A,j. \quad Q(i,c,n,A[j],j) \wedge 0 \leq j < n = i \Rightarrow A[j] = c$$

Solution: $Q(i,c,n,a,j) \equiv 0 \leq j < i \Rightarrow a = c$

Other Horn-clause solving methods

- Datalog engine over abstract tables
  - Hash-tables for JavaScript points-to analysis
  - Header-space algebra tables (ternary simulation) for checking routers [Lopes, Varghese & B]
- Tabulation as Infinite Descente
  (a.k.a cyclic induction proofs)
  Popular in CLP community.
  New super-acceleration [Voronkov & B]
- Recursive functions in *Leon*

Summary

An efficient SMT solver

http://research.microsoft.com/projects/z3

Testing, Verification, Validation, Modeling

http://rise4fun.com

Fixedpoints:

Satisfiability of *Horn Clauses* Modulo Theories

http://rise4fun.com/z3py/tutorial/fixedpoints.htm