BOXES: A Symbolic Abstract Domain of Boxes

Arie Gurfinkel and Sagar Chaki

Software Engineering Institute
Carnegie Mellon University

September 16th, 2010
SAS 2010

Disjunctive Refinement of an Abstract Domain

Bounded disjunctions

- extend base domain with disjunctions of size at most k
- *all* operations are done by lifting corresponding base domain operations
- easy to implement by modifying program control flow graph

Finite Powerset Domain [Bagnara et al.]

- extend base domain with all *finite* disjunctions
- *most* operations are done by lifting corresponding base domain opertions
- finding a good widening is complex (and often tricky)

Predicate Abstraction

- extend *finite* base domain with *all* disjunctions
- domain elements are represented by BDDs
- no widening required

OUR WORK

Outline

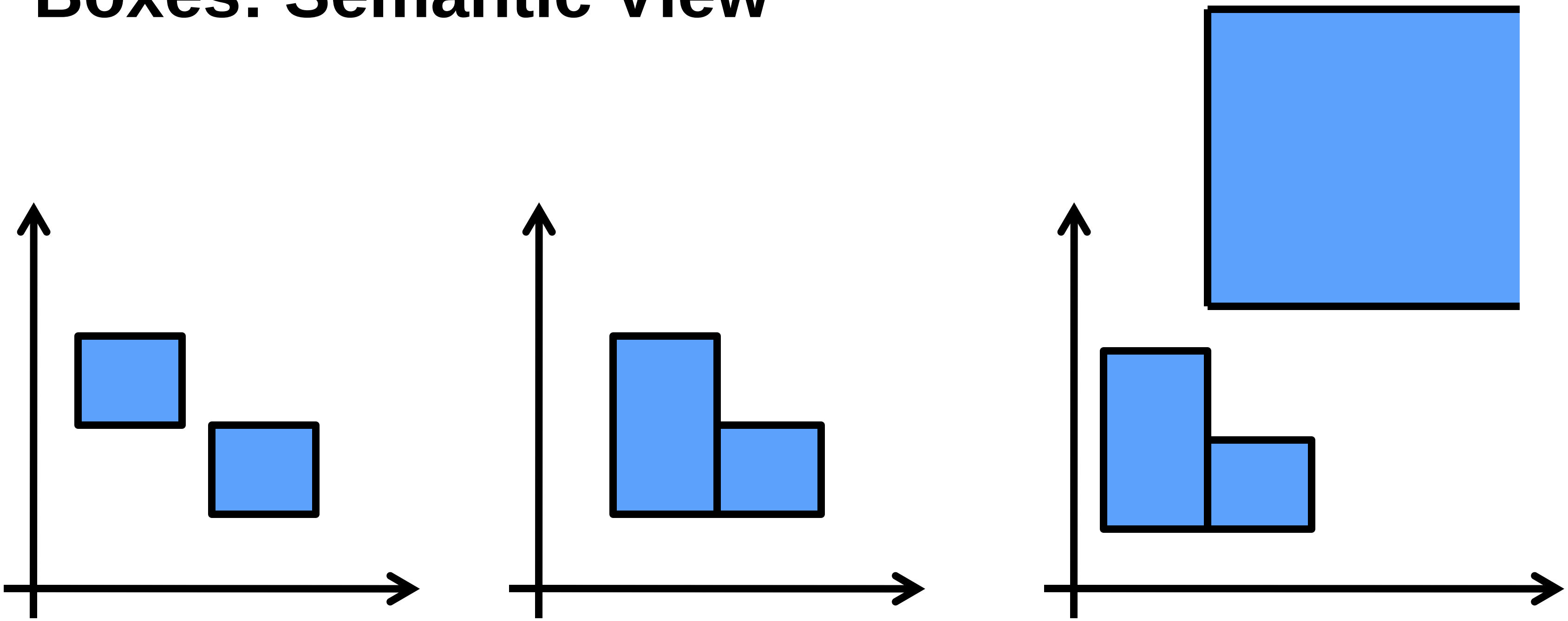
Boxes: semantics, representation, operations

Widening

Experiments

Conclusion

Boxes: Semantic View



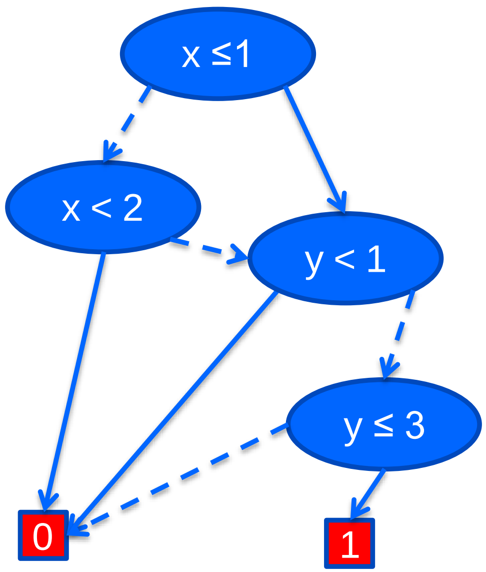
Boxes are “finite union of box values”

(alternatively)

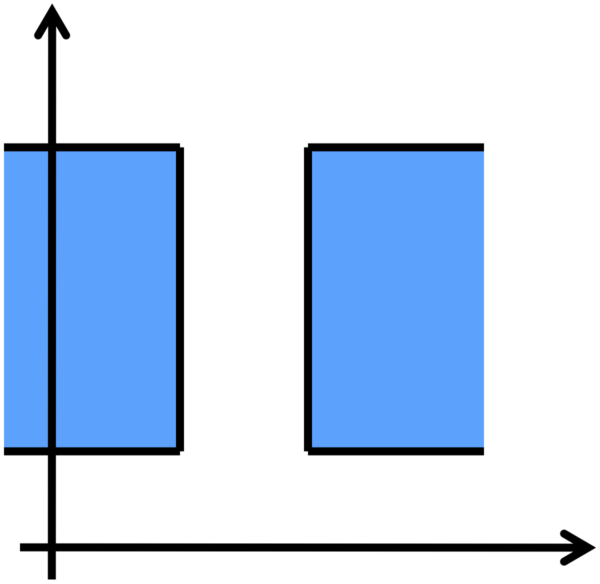
Boxes are “Boolean formulas over interval constraints”

Boxes: Representation

LDD



Semantics



Represented by (Interval) Linear Decision Diagrams (LDD)

- BDDs + non-terminal nodes are labeled by interval constraints + extra rules
- retain complexity of BDD operations
- canonical for Boxes
- available at http://lindd.sf.net

Domain Operations

Basic domain operations are ed by LDD operations

meet

order
(semantic)

join

Operation	Complexity
$f \wedge g$	O(|f||g|)
ITE(h, f, g)	O(|h||f||g|)
$\neg f$	O(1)

Operation	Complexity
$f \vee g$	O(|f||g|)
$f \Rightarrow g$	O(|f||g|)
$\exists U.\ f$	$O(\|f\|\ 2^{\|U\|})$

projection

Additional operations

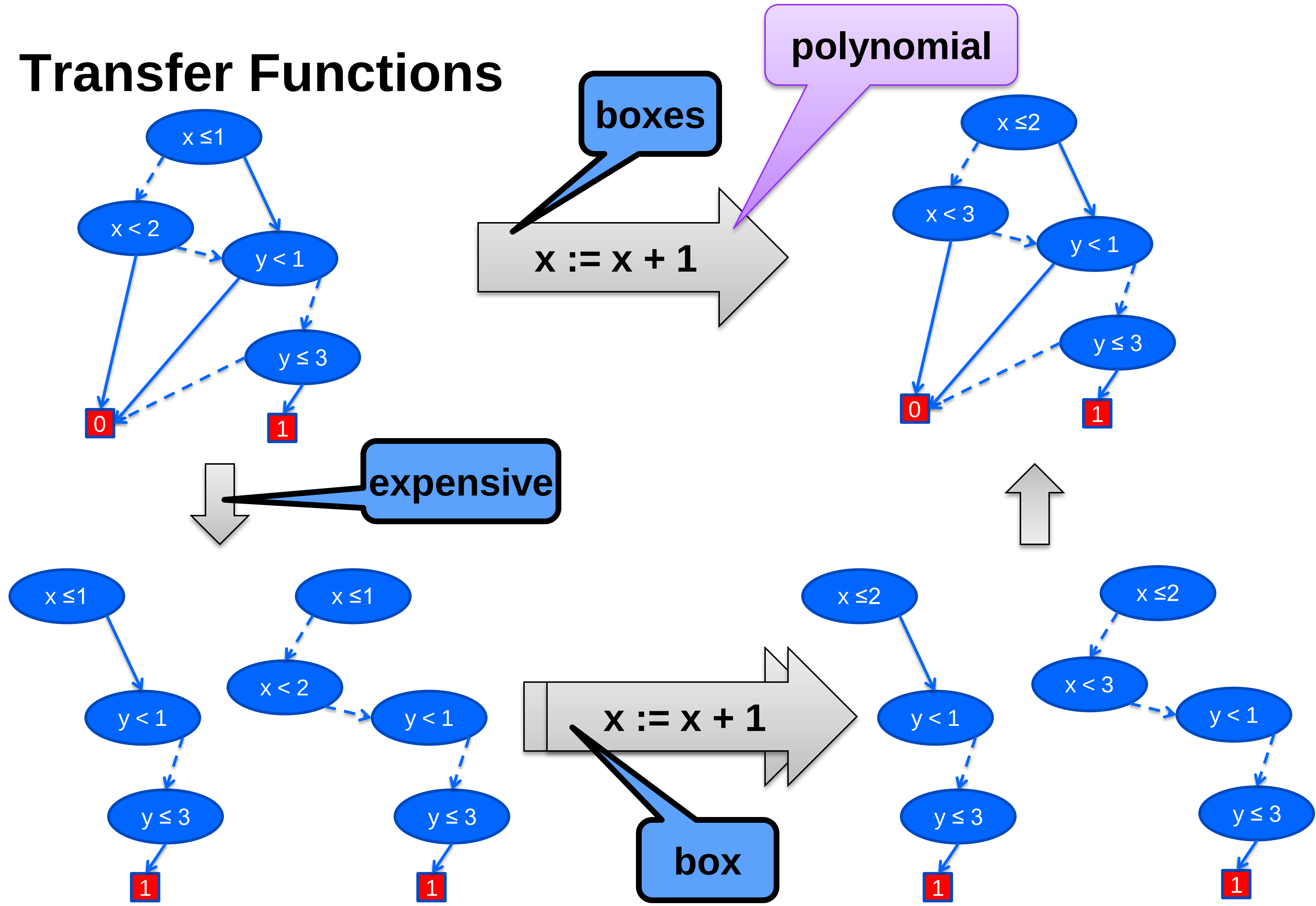
- set difference $f \setminus g$ implemented by $f \wedge \neg g$
- BoxHull (f) – smallest Box containing f
- BoxJoin (f, g) – smallest Box containing the union of Box f and Box g

used to compare

Box and Boxes

All operations are polynomial in the size of the representation

Transfer Functions



Outline

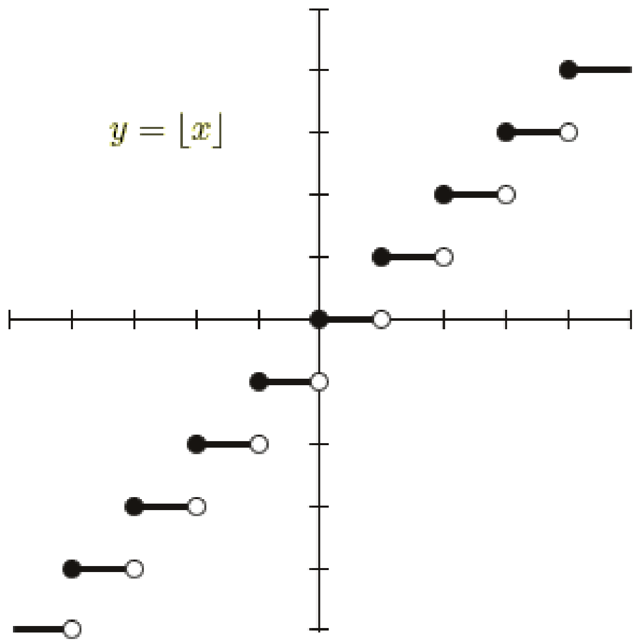
Boxes: semantics, representation, operations

Widening

Experiments

Conclusion

Step Function



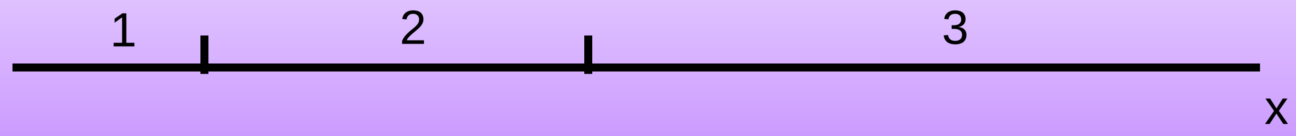
A function on the reals $\mathbb{R}$ is a *step function* if it can be written as a *finite* linear combination of semi-open intervals

$$f(x) = \alpha_1 f_1(x) + \cdots + \alpha_n f_n(x)$$

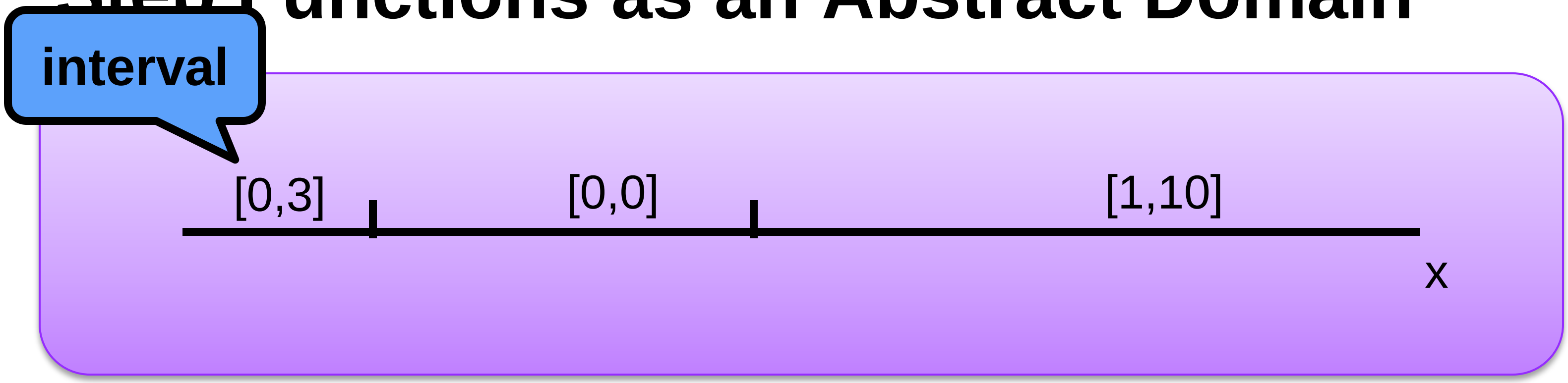
where $f_i \in \mathbb{R}$ and $\alpha_i(x)=1$ if $x \in [a_i, b_i)$ and 0 otherwise, for i=1,…,n

Weisstein, Eric W. "Step Function." From *MathWorld*--A Wolfram Web Resource.
http://mathworld.wolfram.com/StepFunction.html

Step Functions as an Abstract Domain



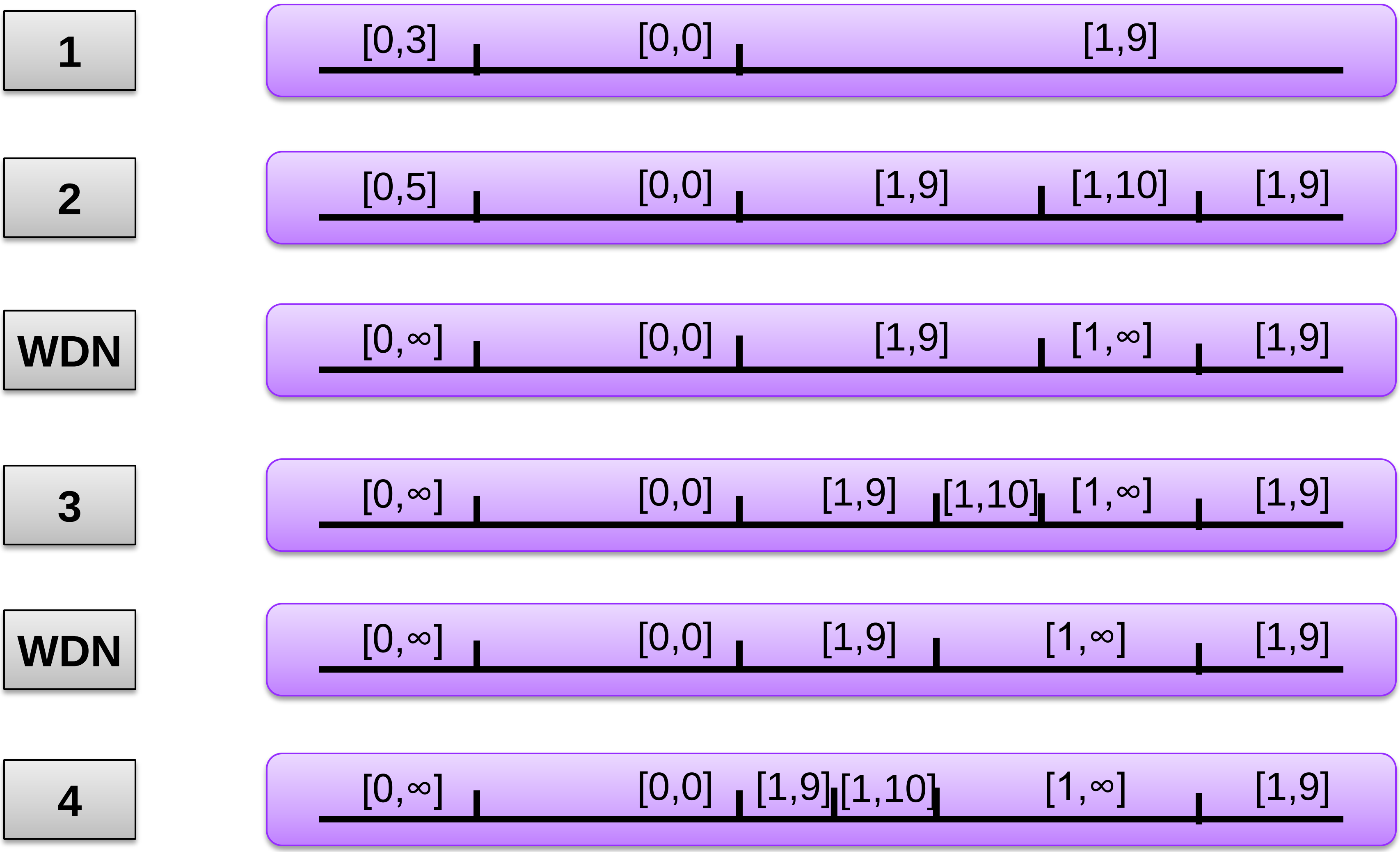
Step Functions as an Abstract Domain



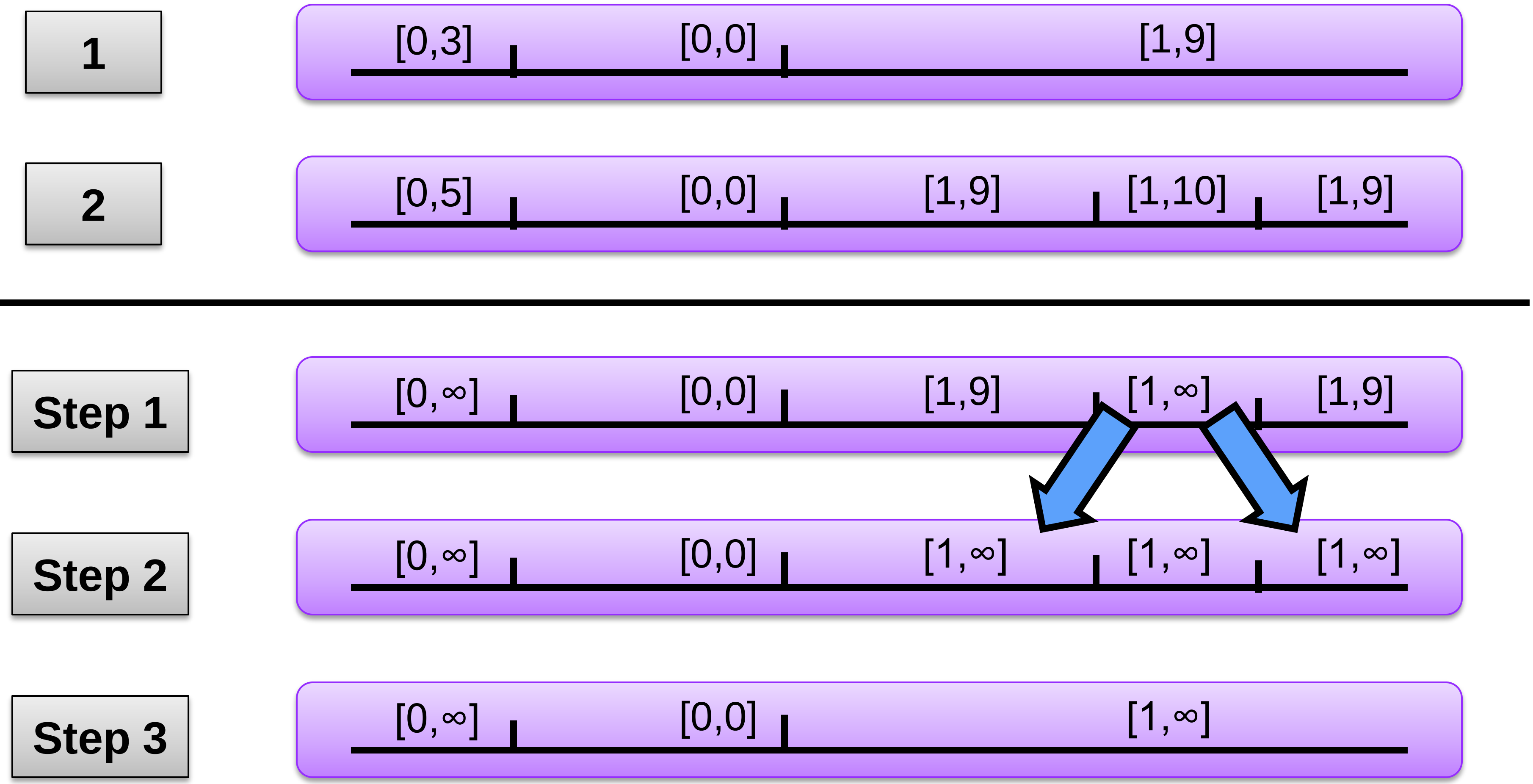
STEP(D) an abstract domain of step functions over an abstract domain D

- elements are step functions $\mathbb{R} \rightarrow D$
- order is pointwise: $f \sqsubseteq g$ iff $\forall x . f(x) \sqsubseteq_D g(x)$
- join is pointwise: $f \sqcup g$ is $\lambda x . f(x) \sqcup_D g(x)$
- meet is pointwise: $f \sqcap g$ is $\lambda x . f(x) \sqcap_D g(x)$
- widen is pointwise: $f \nabla g$ is $\lambda x . f(x) \nabla_D g(x)$????

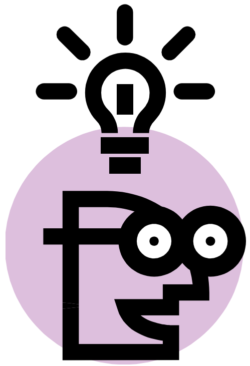
Pointwise Extension of Widen Diverges



Widening for Step Functions



Back to Boxes



Boxes are Step functions!

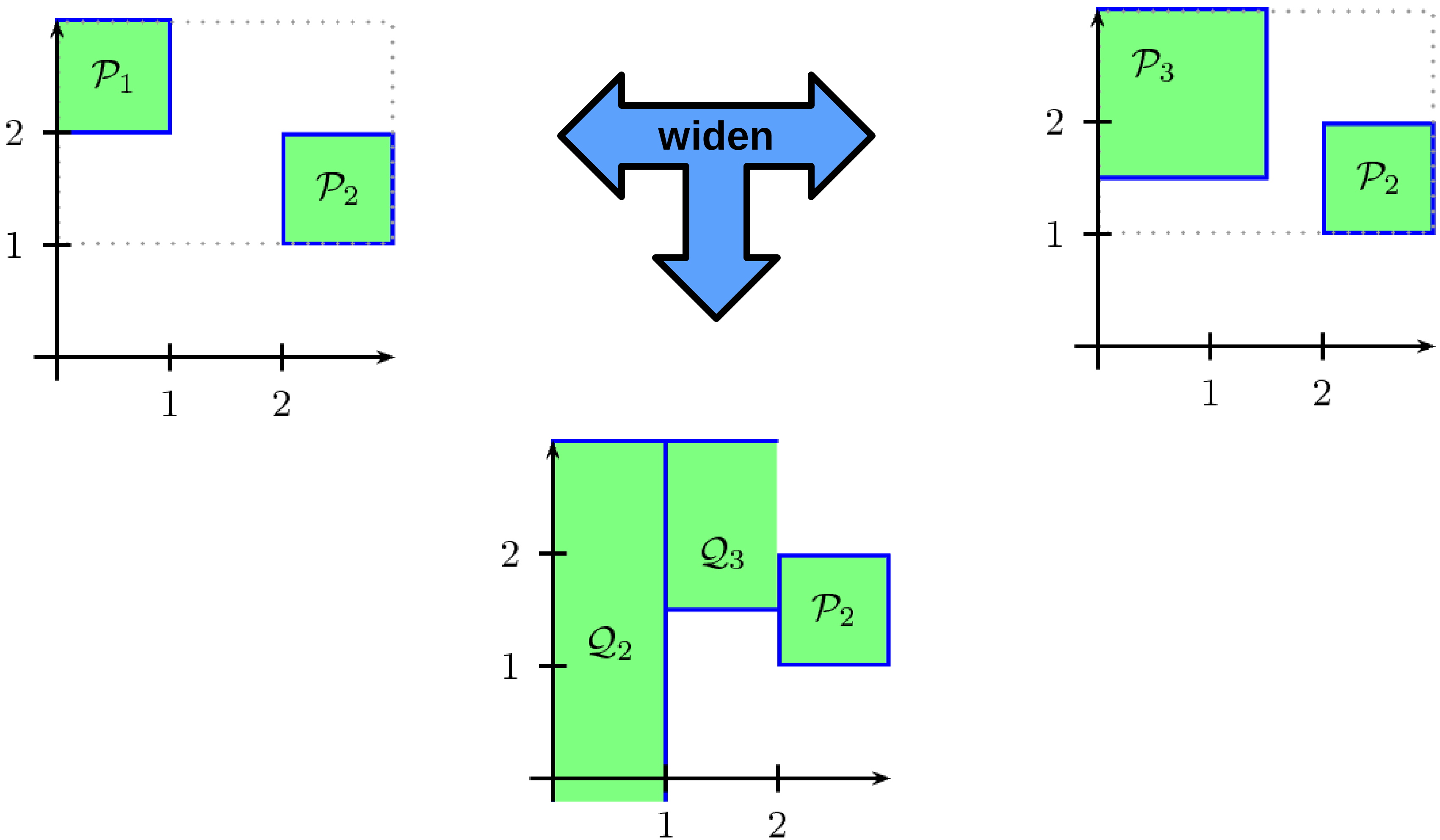
- 1-dim Boxes are STEP($\{\bot,\top\}$) $\mathbb{R}\to\{\bot,\top\}$
- 2-dim Boxes are STEP (STEP ($\{\bot,\top\}$) $\mathbb{R}\to\mathbb{R}\to\{\bot,\top\}$
- n-dim Boxes are STEP^n ($\{\bot,\top\}$) $\mathbb{R}^n\to\{\bot,\top\}$

Widen for $\{\bot,\top\}$ is trivial

Widen for n-dim Boxes is defined recursively on dimensions

In the paper, a polynomial time algorithm that implements this widen operator directly on LDDs.

Widen: An Example



Boxes versus Finite Powersets

	Boxes	Finite Powerset
Base domain	Box	Any
Representation	Decision Diagram	Set / DNF
Domain order	semantic	syntactic
Complexity	polynomial in representation	polynomial in representation
Singleton Widen	Box	base domain
Widen	Step Function	Multiple Choices

Experiments: Invariant Computation

Abstract Domains

- LDD Box – Box domain using LDDs
- LDD Boxes – Our Boxes domain using LDDs
- PPL Box – `Rational_Box` of Parma Polyhedra Library (PPL)
- PPL Boxes – `Pointset_Powerset<Rational_Box>` of PPL

Analyzer

- custom analyzer on top of LLVM compiler infrustructure
- computes loop invariants for all loops over all SSA variables in a function

Benchmark

- from open source software: mplayer, CUDD, make, …
- Stats: 5,727 functions

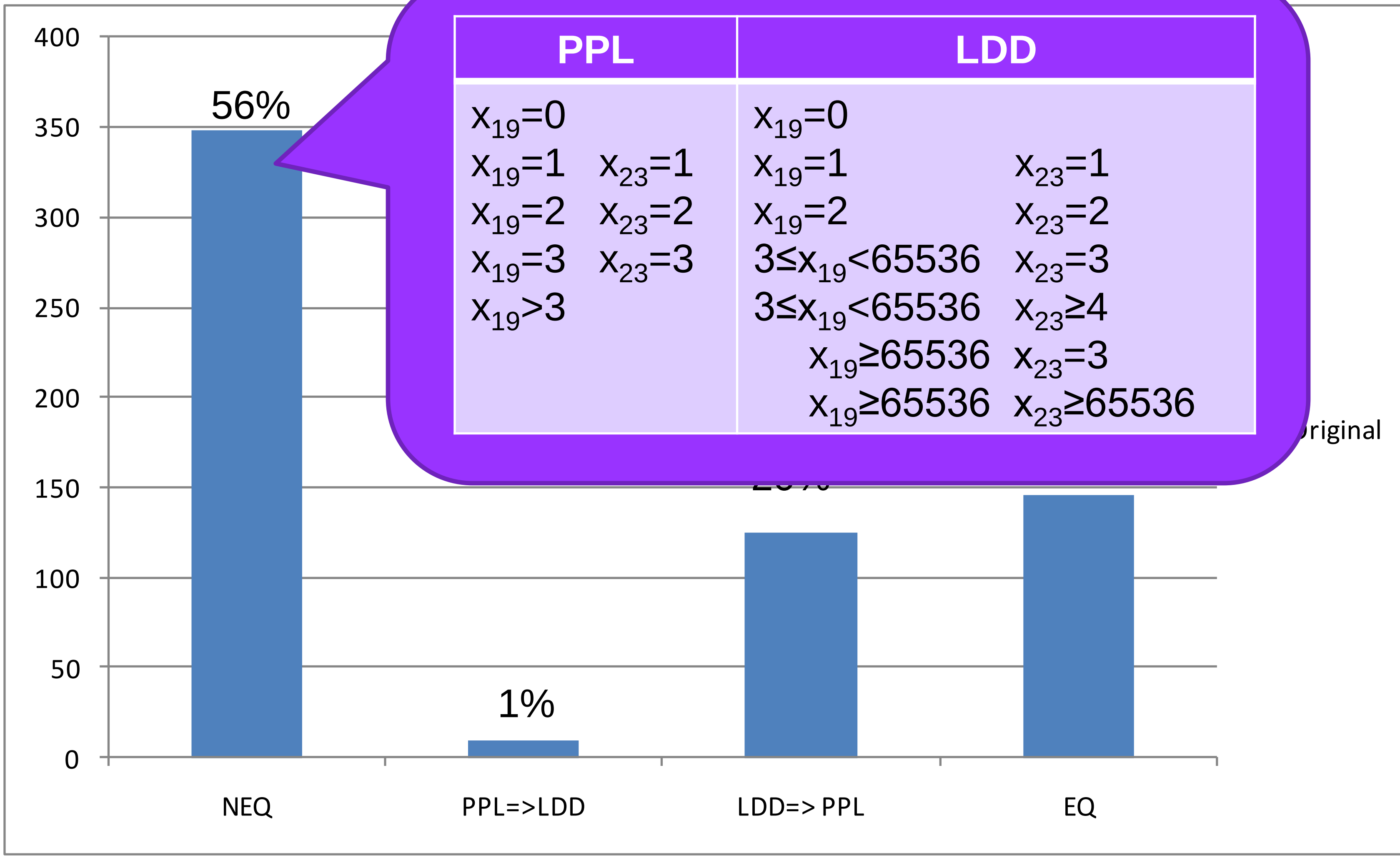
  9 – 9,052 variables (avg. 238, std. 492)

  0 – 241 loops (avg. 7, std. 12)

Results: Time

Domain	%Solved (w/ 60s TO)	Time (m)	% in Basic	% in Image	% in Widen
LDD Box	99.8%	4	77%	23%	0%
PPL Box	96.1%	117	86%	14%	0%
LDD Boxes	87.9%	118	61%	38%	1%
PPL Boxes	14.2%	201	95%	1%	3%

Results: Precision



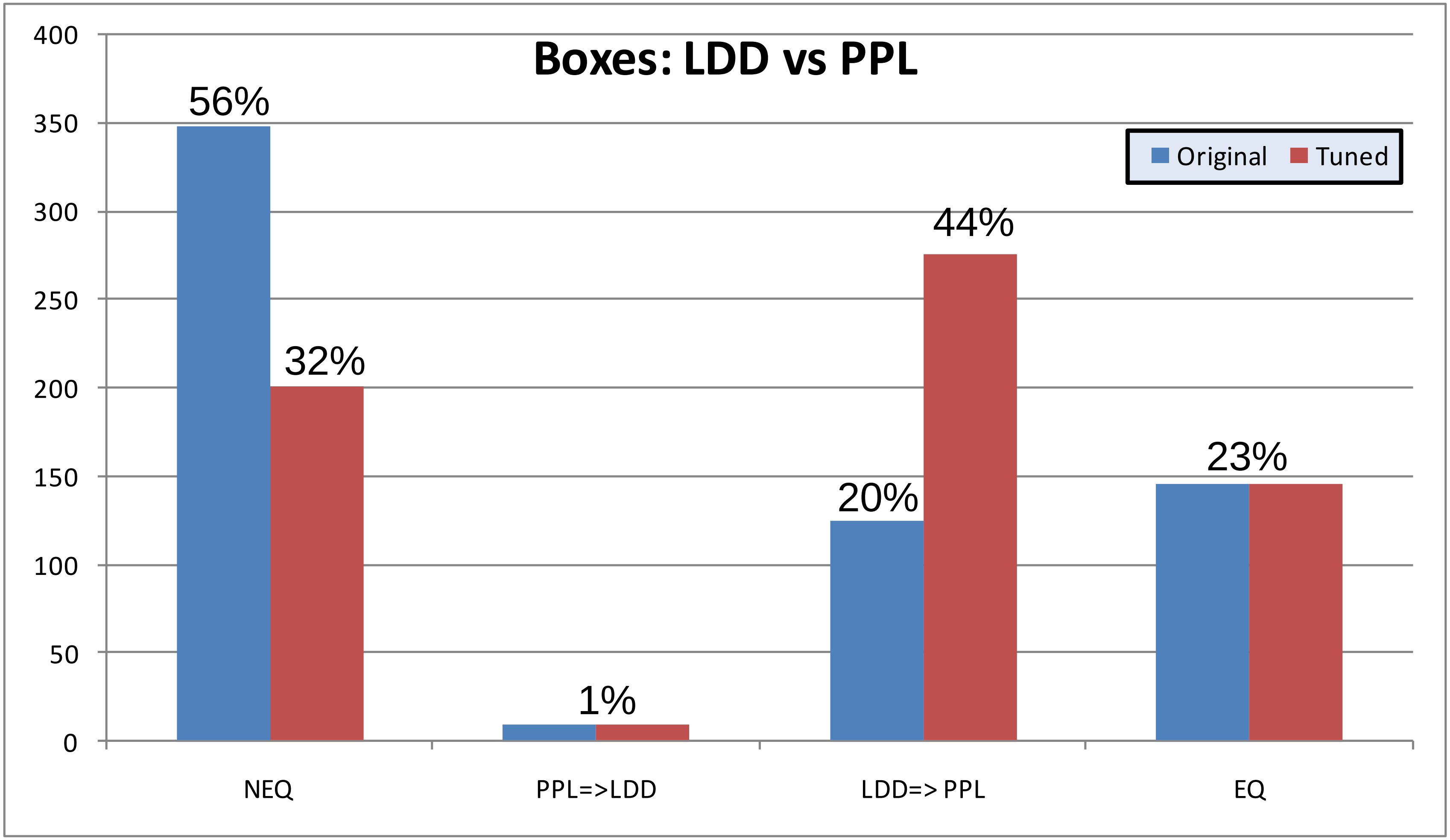
PPL		LDD	
$x_{19}=0$		$x_{19}=0$	
$x_{19}=1$	$x_{23}=1$	$x_{19}=1$	$x_{23}=1$
$x_{19}=2$	$x_{23}=2$	$x_{19}=2$	$x_{23}=2$
$x_{19}=3$	$x_{23}=3$	$3 \leq x_{19} < 65536$	$x_{23}=3$
$x_{19}>3$		$3 \leq x_{19} < 65536$	$x_{23} \geq 4$
		$x_{19} \geq 65536$	$x_{23}=3$
		$x_{19} \geq 65536$	$x_{23} \geq 65536$

Widening: PPL vs LDD

```
x = 0;
y = 0
while (1){
   x++;
   y++;
}
```

	PPL		LDD
Iteration 1	x=0 y=0	≡	x=0 y=0
Iteration 2	x=0 y=0 x=1 y=1	≡	x=0 y=0 x=1 y=1
Widen 1	x=0 y=0 x=1 y=1 1<x	⊇	x=0 y=0 1≤x y=1
Iteration 3			x=0 y=0 1≤x y=1 2≤x y=2
Widen 2			x=0 y=0 1≤x y=1 1≤x 2≤y

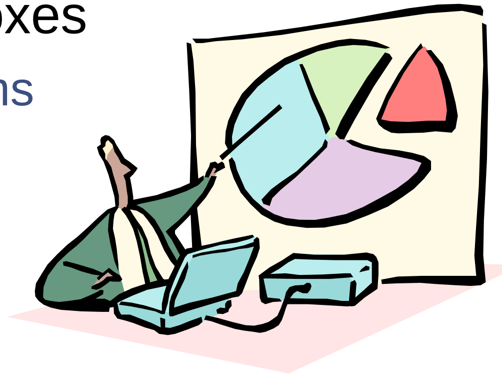
Results: Precision w/ Tuned Widening



Conclusion

Boxes: A new disjunctive abstract domain of sets of boxes

- efficient representation based on Linear Decision Diagrams
- semantic order relation
- efficient operations and widening
- more precise and efficient than finite powersets of box



A new widening scheme

- lifting widening from a base domain to the domain of step functions

Future Work

- applications
- extending the technique to richer base domains, i.e., octagons, TVPI
  - representation and base operations are easy (already exist in LDD)
  - widening?

http://lindd.sf.net

THE END

Transfer Functions: PPL vs LDD

PPL

$x=1$
$x=2$

y := x →

$x=1 \; y=1$
$x=2 \; y=2$

$\equiv$ $\qquad$ $\subseteq$

LDD

$1 \leq x \leq 2$

y := x →

$1 \leq x \leq 2, 1 \leq y \leq 2$

Contact Information

Presenter
Arie Gurfinkel
RTSS
Telephone: +1 412-268-5800
Email: arie@cmu.edu

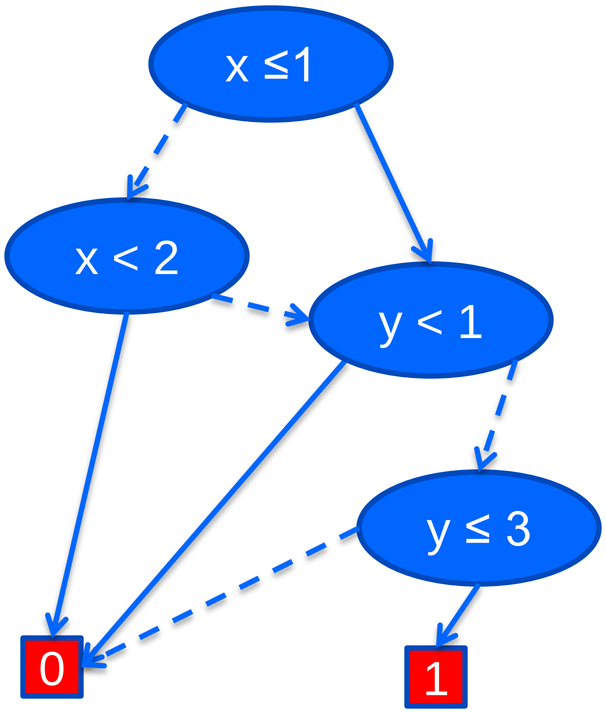
U.S. mail:
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

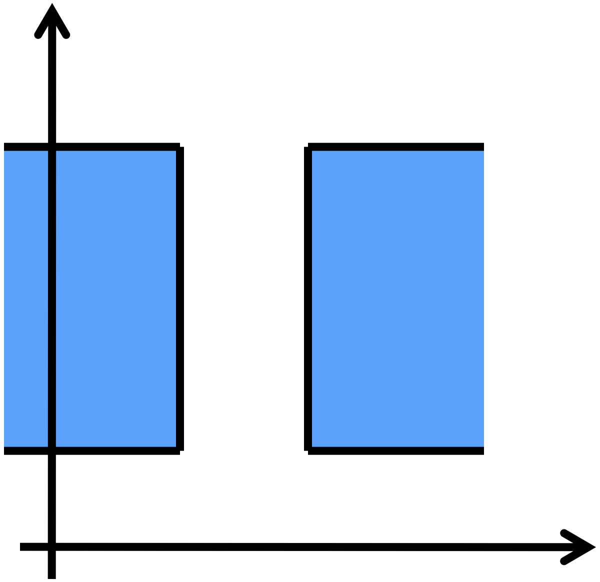
Web:
www.sei.cmu.edu
http://www.sei.cmu.edu/contact.cfm

Customer Relations
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

Boxes: Representation





Represented by (Interval) Linear Decision Diagrams (LDD)

- BDDs + non-terminal nodes are labeled by interval constraints + extra rules
- retain complexity of BDD operations
- canonical for Boxes
- available at http://lindd.sf.net