Using Counterexamples for Improving the Precision of Reachability Computation with Polyhedra

Chao Wang, Zijiang Yang*, Aarti Gupta, Franjo Ivančić

NEC Laboratories America, Princeton, NJ

*Western Michigan University, Kalamazoo, MI

Motivation

- Static analysis based on abstract interpretation uses widening to accelerate fixpoint computation
  - **widening** is designed *a prior* – if it isn't precise enough
    - either report inconclusive (a warning),
    - or redesign the widening operator manually

- We use CEGAR to improve the precision of abstract interpretation
  - introduce a new operator – *extrapolation with a care set*
    - the care set is a forbidden area
    - iteratively expand the care set to remove false bugs
  - new algorithms for simplifying polyhedral representations

Related Work

- Widening/Extrapolation in polyhedral domains
  - widening [Cousot78], extrapolation [Henzinger95]
  - for polyhedral powersets [Bultan98,Bagnara04]
  - widening up-to [Halbwachs93,97], lookahead widening [Gopan06]

- CEGAR [Kushan94,Clarke00,Ball01]

- [Gulavani and Rajamani, TACAS06] uses CEGAR to refine abstract interpretation: after identifying the precision-loss step,
  - they use LUB (least upper bound) to replace widening
  - we expand the *care set* to improve widening precision

Outline

- Background
- **Preliminaries**
- Extrapolation with a Care Set
- Optimizations
- Application in Program Verification
- Conclusions

Widening [Cousot76]

- A widening operator on a partial order set $(L, \sqsubseteq)$ is a partial function $\nabla : L \times L \to L$ such that
  1. for all $x, y \in L$, $x \sqsubseteq x \nabla y$ and $y \sqsubseteq x \nabla y$;
  2. for all ascending chains $y_0 \sqsubseteq y_1 \sqsubseteq$ ..., the ascending chain defined by $x_0 := y_0$ and $x_{i+1} := x_i \nabla y_{i+1}$ for $i \geq 0$ is *not strictly increasing*.

- In reachability fixpoint computation $F = \mu Z \,.\, I \cup post(Z)$

$$\begin{aligned} y_0 &= I \\ y_1 &= I \;\cup post(I) \\ y_2 &= x_1 \cup post(x_1) \\ y_3 &= \ldots \end{aligned} \qquad\qquad \begin{aligned} x_0 &= I \\ x_1 &= I\; \nabla y_1 \\ x_2 &= x_1 \nabla y_2 \end{aligned}$$

In model checking one would replace $x_i \nabla y_{i+1}$ by y_{i+1}

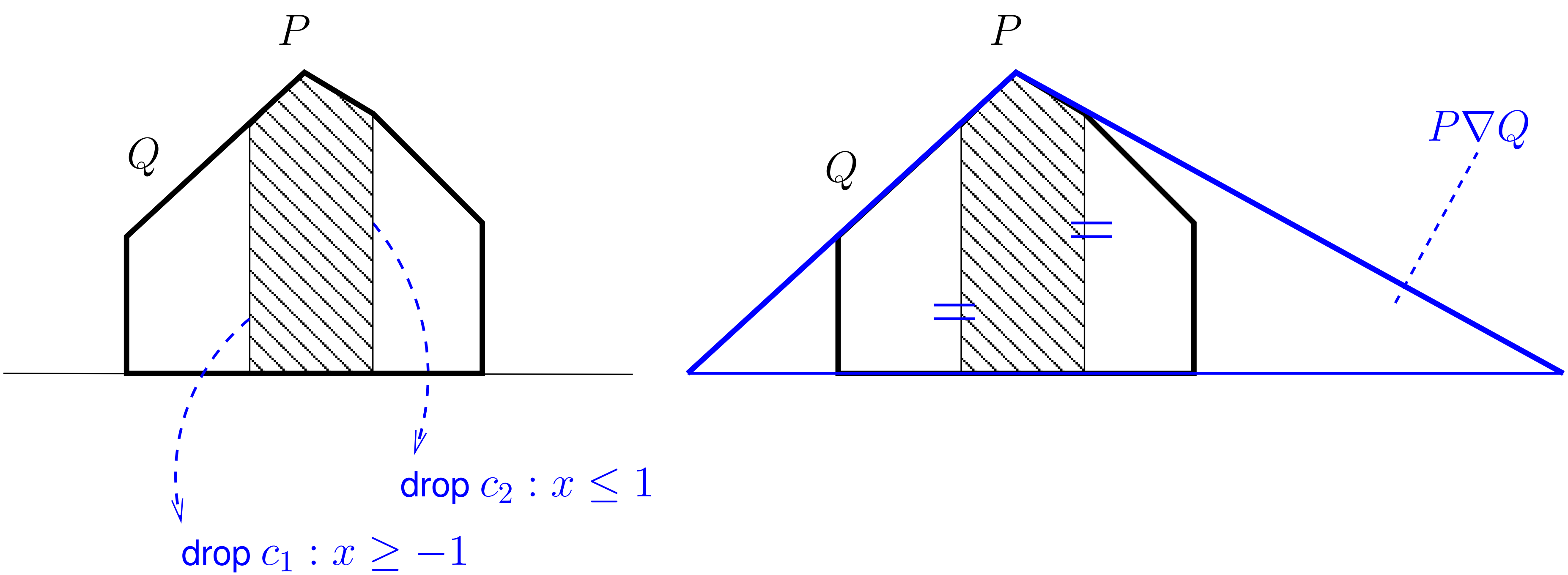
Polyhedral Abstract Domains

- A polyhedron is $P = \{\mathbf{x} \in \mathbb{Z}^n \mid \forall i : \mathbf{a}_i^T \cdot \mathbf{x} \leq b_i\}$
  - P is a finite conjunction of linear inequality constraints
  - $\mathbf{a}_i^T \cdot \mathbf{x} \leq b_i$ is a linear inequality constraint
    - a constraint is also called a “half-space”
- The *constraint system representation* of a polyhedron
  - implemented in `the Omega Library`
  - an alternative is the *generator system representation*
    - the two have complementary strengths
    - both are used in `PPL`

Standard Widening

Let $P \sqsubseteq Q$ be two convex polyhedra; $P \nabla Q$ is defined as:

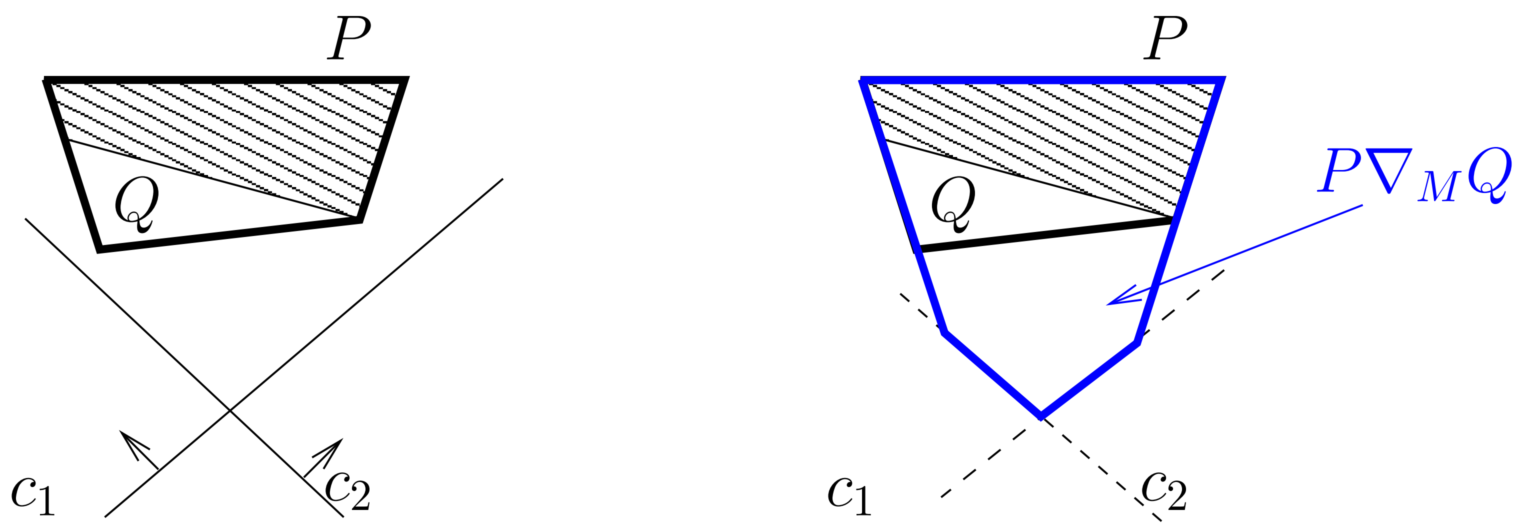
1. when P is empty, return Q;
2. otherwise, remove from P all inequalities not satisfied by Q.



[Cousot78,Halbwachs79]

Widening Up-To

- $P \nabla_M Q$ is the conjunction of $P \nabla Q$ with all the constraints in M that are satisfied by both P and Q.



- The *up-to set* M – a finite set of linear inequality constraints
  - if x is of subrange type 1..10, add $x \geq 1$ and $x \leq 10$
  - for a loop `for (x=0; x<5; x++)`, add $x < 5$

[Halbwachs93,97]

Polyhedral Powerset Domains

- A polyhedral powerset is a finite union of convex polyhedra
- For two powersets P and Q, we compute $P_i \nabla Q_i$ for all pairs $P_i \in P$ and $Q_i \in Q$ such that $P_i \sqsubseteq Q_i$
- $P \nabla Q$ is an *extrapolation*
  - also used by [Bultan98]
  - no termination guarantee

Outline

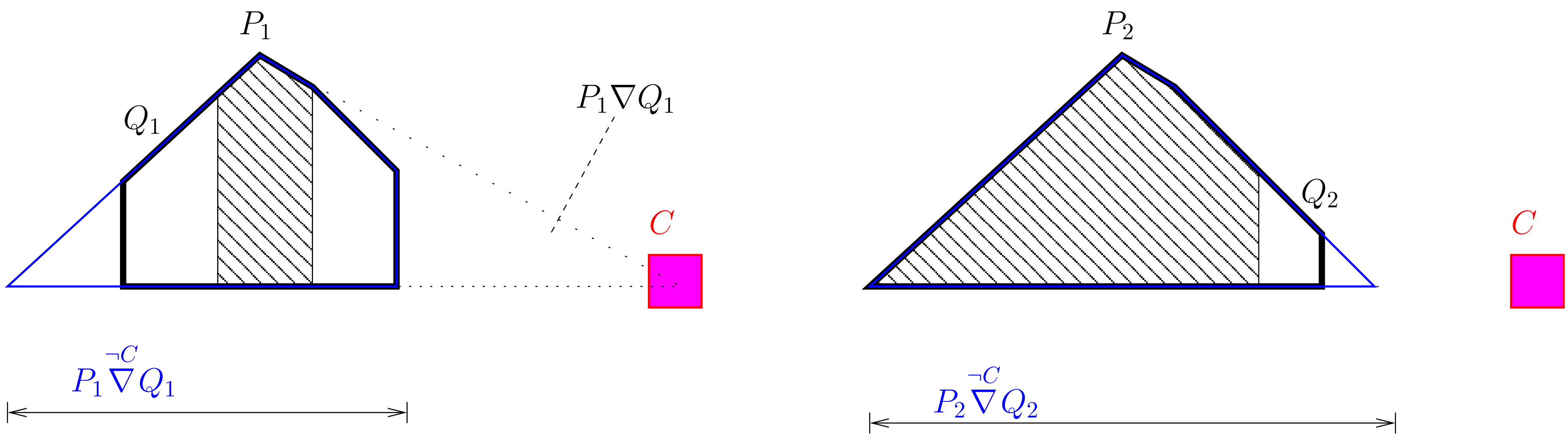
- Background
- Preliminaries
- **Extrapolation with a Care Set**
- Optimizations
- Application in Program Verification
- Conclusions

Extrapolation with a Care Set

- Definition: a partial function $\overset{\neg C}{\nabla} : L \times L \to L$ on set $(L, \sqsubseteq)$
  1. for all $x, y \in L$, we have $x \sqsubseteq x \overset{\neg C}{\nabla} y$ and $y \sqsubseteq x \overset{\neg C}{\nabla} y$;
  2. for all ascending chains $y_0 \sqsubseteq y_1 \sqsubseteq$ ..., the ascending chain defined by $x_0 := y_0$ and $x_{i+1} := x_i \overset{\neg C}{\nabla} y_{i+1}$ for $i \geq 0$ satisfies: if $y_i \cap C = \emptyset$, then $x_i \cap C = \emptyset$.

- We use a care set C to restrict "directions-of-growth"
  - shall not add new states to C (forbidden area)

Extrapolation with a Care Set

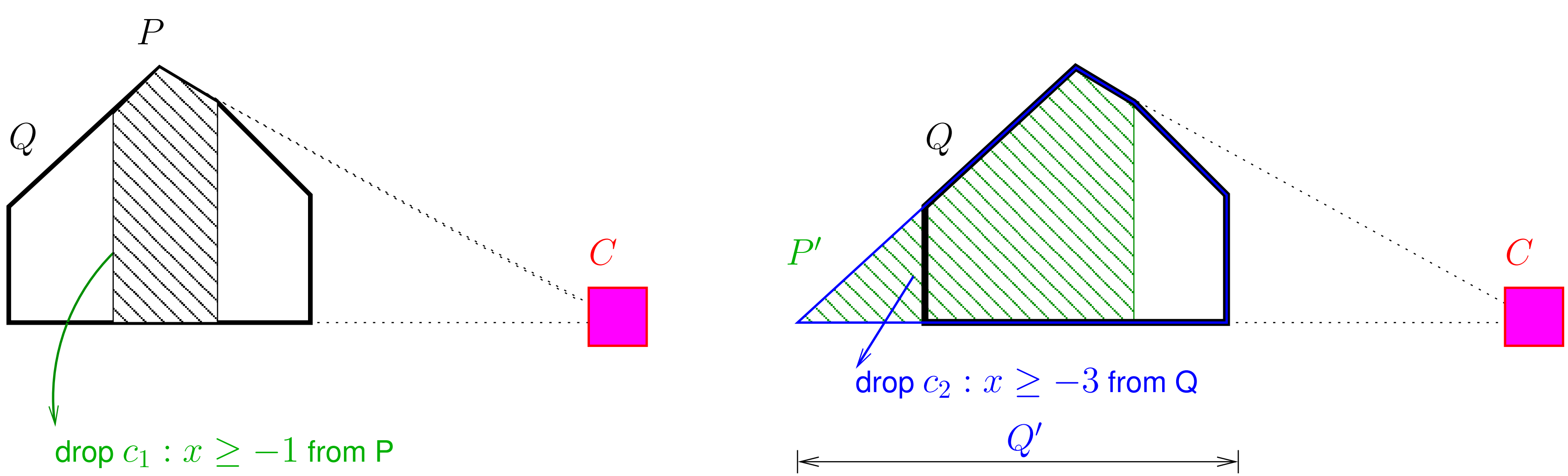


- $P_1 \overset{\neg C}{\nabla} Q_1$ is smaller than $P_1 \nabla Q_1$
- $P_2 \overset{\neg C}{\nabla} Q_2$ is the same as $P_2 \nabla Q_2$

Extrapolation with a Care Set (algorithm)

- Given $P \sqsubset Q$ and the care set C
  1. build a new polyhedron P': for each constraint c of P whose half-space does not contain Q, if $P^c_{true} \cap C = \emptyset$, then drop c.
  2. build a new polyhedron Q': drop constraint c of Q whose half-space does not contain P', if $Q^c_{true} \cap C = \emptyset$.

  return Q' as the result.



For Program Verification

- We use a polyhedral powerset domain on linear programs
  - where $pre()$, $post()$, and LUB $\cup$ are precise
  - only ∇ causes precision loss
- In reachability fixpoint computation
  - $\hat{F}_0 = I$, and $\hat{F}_{i+1} = \hat{F}_i \nabla (\hat{F}_i \cup post(\hat{F}_i))$ for all $i \geq 0$.
  - check invariant property ψ (bad states are $\neg\psi$)
- Both $\widehat{F_i}$ and $\neg\psi$ are powersets of polyhedra

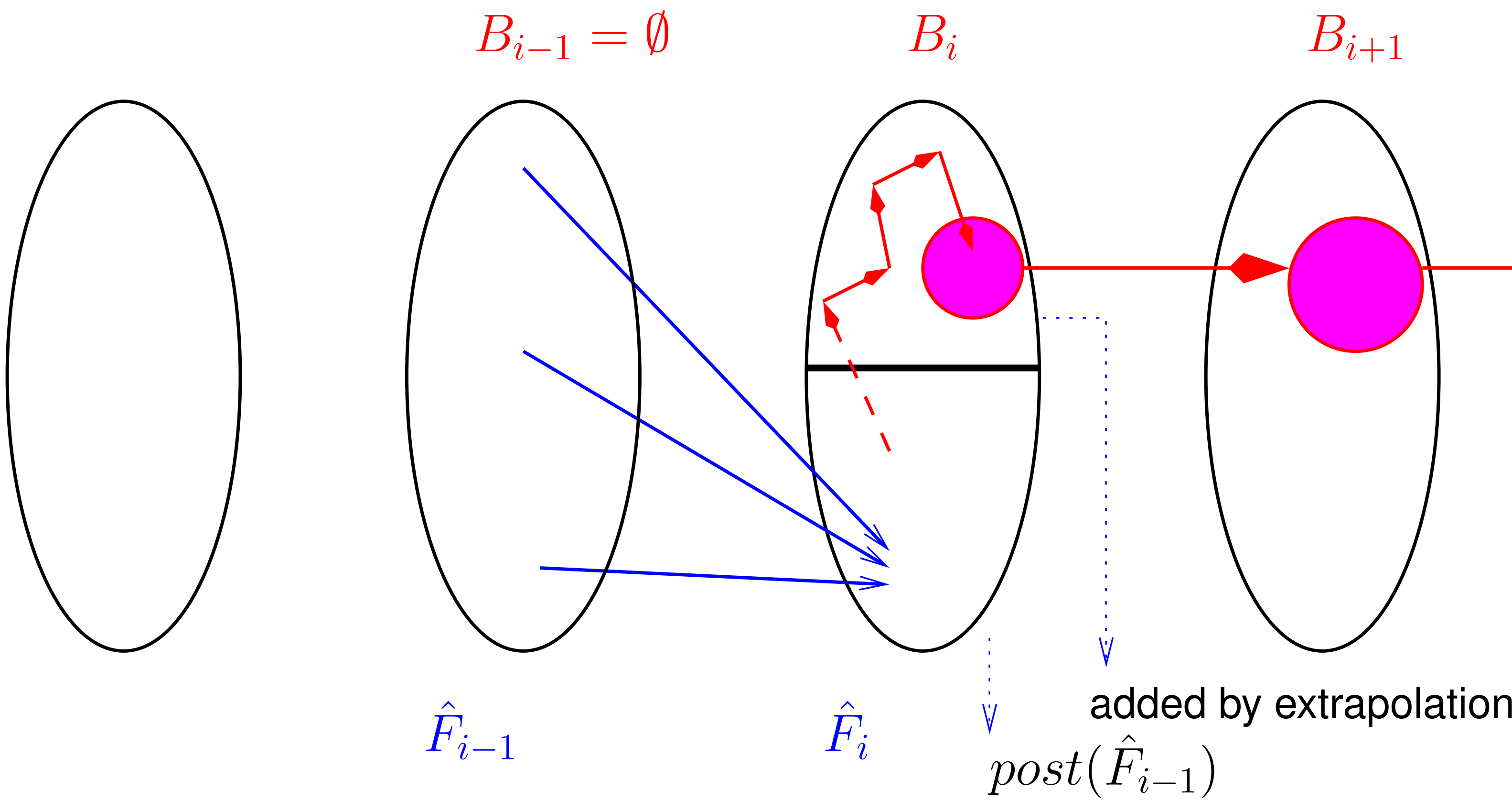
The Iterative Procedure

Initialize the care set $C = \emptyset$

1. Start forward reachability fixpoint computation
   - $\hat{F}_0 = I$, and $\hat{F}_{i+1} = \hat{F}_i \overset{\neg C}{\nabla} (\hat{F}_i \cup post(\hat{F}_i))$ for all $i \geq 0$.
2. If $\hat{F}_{fi} \cap \neg\psi \neq \emptyset$, start precise backward analysis
   - $B_{fi} = \hat{F}_{fi} \cap \neg\psi$, and
   - $B_{i-1} = \hat{F}_{i-1} \cap pre(B_i)$ for all $i \leq fi$ and $i > 0$.
3. If $B_0 \neq \emptyset$, we have found a counterexample
   - otherwise, $B_{i-1} = \emptyset$; we <u>expand care set C to include B_i</u>
   - go back to step 1

Expanding the Care Set

Theorem 1 *If there exists an index $0 < i < fi$ such that $B_{i-1} = \emptyset$, then B_i must have been introduced into $\hat{F}_i$ by extrapolation during forward fixpoint computation.*



Correctness and Termination

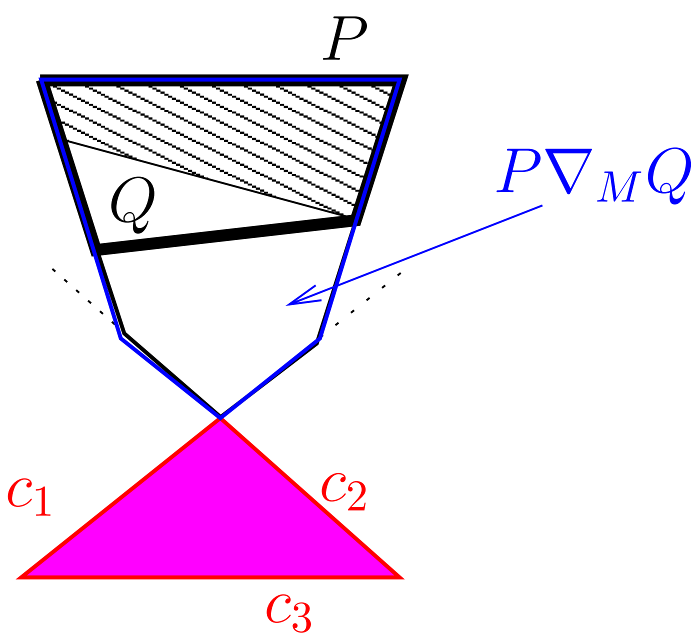
- We keep expanding the care set C until
  1. a counterexample is found (ψ fails), or
  2. $\hat{F} \cap \neg\psi$ becomes empty (ψ holds), or
  3. time/memory out (undecided).

- Conclusive results
  - $\hat{F}$ is an upper bound $\longrightarrow$ proofs are correct
  - backward analysis is precise $\longrightarrow$ CEX's are valid

- Trade-off (due to undecidability)
  - expanding C guarantees precision improvement
  - does not guarantee termination of reachability computation

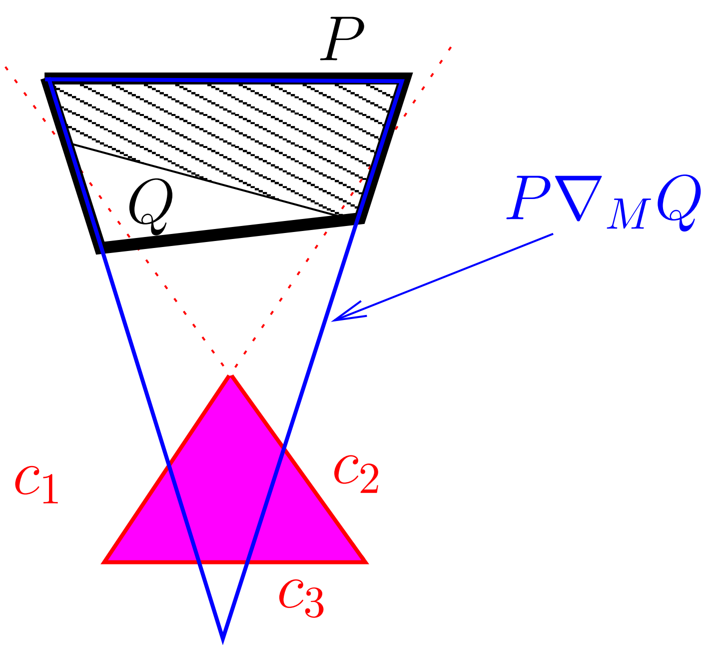
Improving Widening Up-To

- $M = \{\neg c_i \mid c_i$ is a constraint of a polyhedron in $C\}$.



- $M = \{\neg c_1, \neg c_2, \neg c_3\}$
  - if $Q \sqsubseteq \neg c_i$, make sure $(P\nabla_M Q) \sqsubseteq \neg c_i$
  - otherwise, ignore $\neg c_i$

Improving Widening Up-To



- $M = \{\neg c_1, \neg c_2, \neg c_3\}$
  - if $Q \sqsubseteq \neg c_i$, make sure $(P\nabla_M Q) \sqsubseteq \neg c_i$
  - otherwise, ignore $\neg c_i$

- Trade-off
  - ∇_M is a widening – with termination guarantee
  - However, expanding C may not always improve precision

Outline

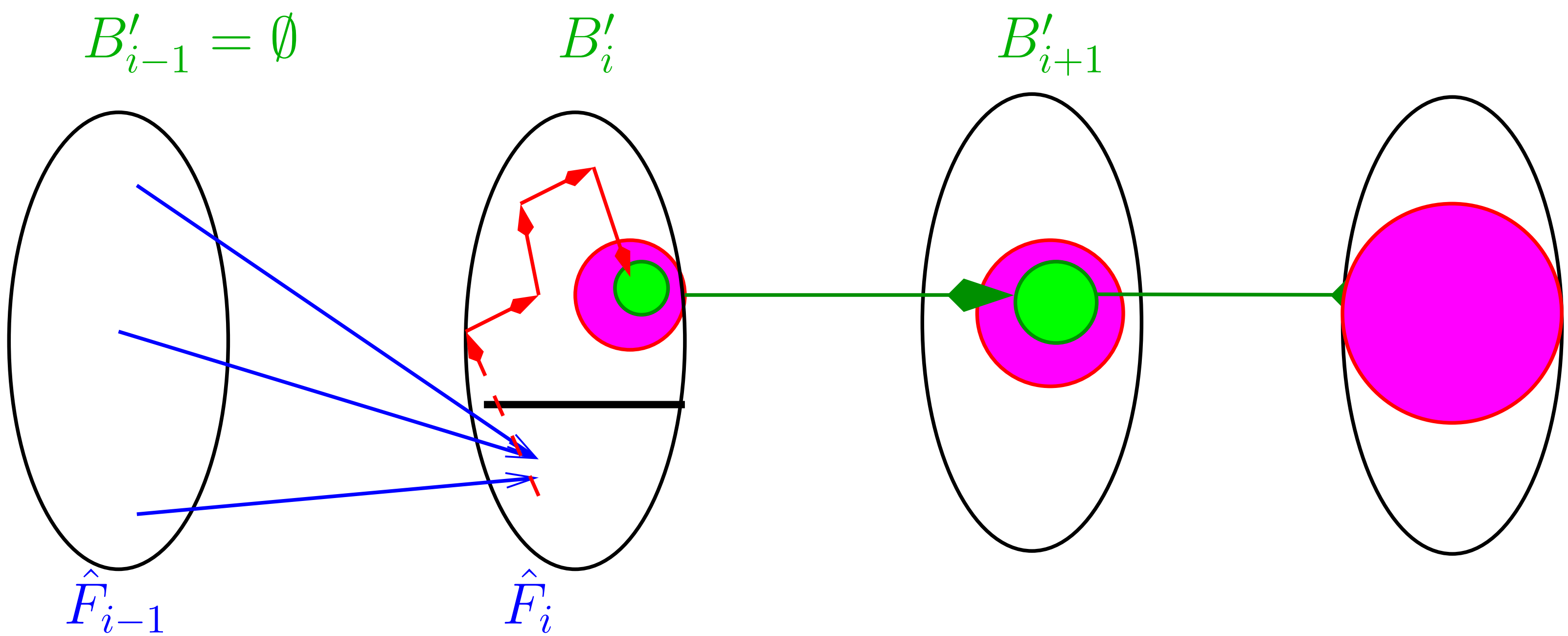
- Background
- Preliminaries
- Extrapolation with a Care Set
- **Optimizations**
- Application in Program Verification
- Conclusions

LUB can be overapproximated

- We have assumed that LUB is precise – to ease presentation
  - It isn't a necessary requirement – we can overapproximate
- We can selectively merge some polyhedra in a powerset
  - as long as the merging result does not intersect C
  - i.e. use $\overset{\neg C}{\cup}$ instead of $\cup$
- Proof of correctness, or Theorem 1, still holds

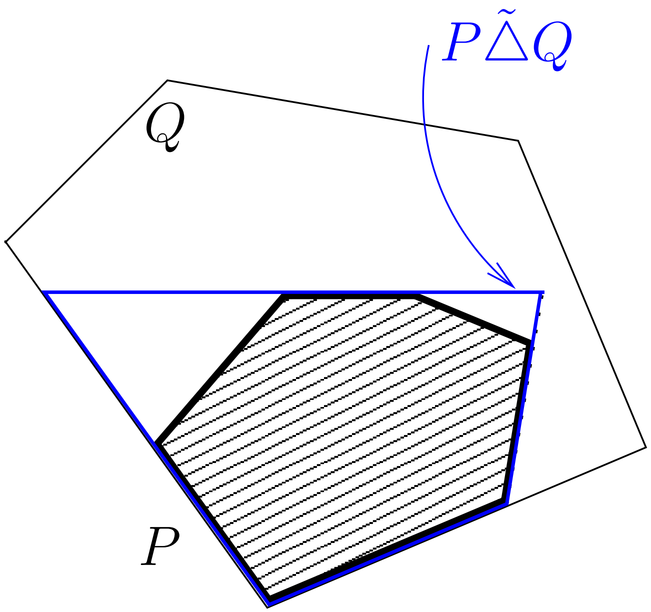
Simplifying Backward Analysis

- Under-approximating backward analysis
  - $B_{i-1} = \hat{F}_{i-1} \cap pre(\text{ SUBSET}(B_i)\ )$
  - B'_i can be a polyhedron of the powerset B_i



Simplifying Polyhedral Representations

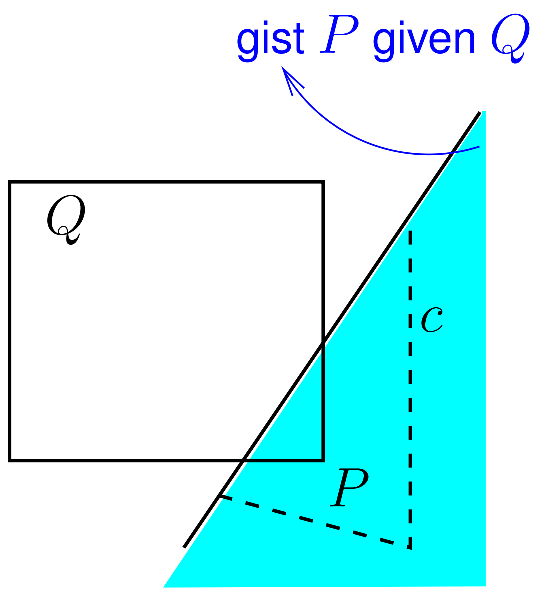
Interpolate: Let P and Q be two sets such that $P \sqsubset Q$. The interpolate $P\tilde{\triangle}Q$ is a new set such that $P \sqsubseteq (P\tilde{\triangle}Q) \sqsubseteq Q$.



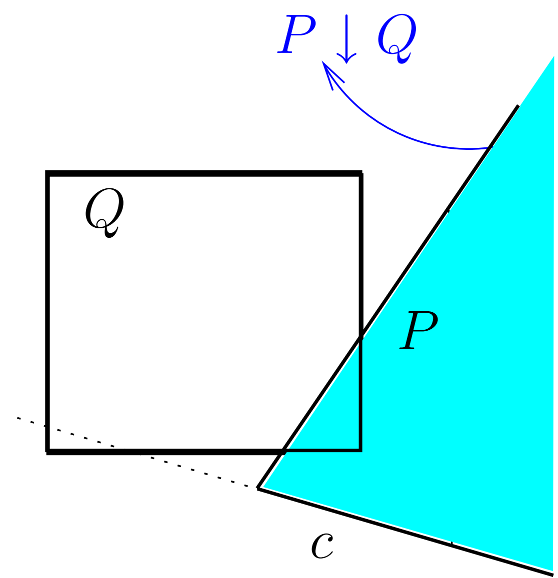
- In forward fixpoint computation
  - $post(\hat{F}_i)$
  - $post(\hat{F}_i \setminus \hat{F}_{i-1})$
  - $post(S)$ where
    $S = (\hat{F}_i \setminus \hat{F}_{i-1})\tilde{\triangle}\hat{F}_i$

Simplifying Polyhedral Representations

Restrict: Let P and Q be two sets. The restrict $P \downarrow Q$ is defined as the new set $\{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{x} \in P \cap Q$, or $\mathbf{x} \notin Q\}$.



GIST: empty $(P^c_{\neg c} \cap Q)$?



OUR alg: empty $(\neg c \cap Q)$?

- Compared to **gist** [Pugh94], our algorithm is cheaper, though it may return larger polyhedra

Outline

- Background
- Preliminaries
- Extrapolation with a Care Set
- Optimizations
- **Application in Program Verification**
- Conclusions

Implementation

- Implemented in F-Soft verification platform [Ivancic et al. CAV05]
  - With MC + SA: still many unresolved properties
    - static analysis isn't precise enough
    - model checking runs out of time/memory
- We want to resolve these properties using extrapolation with an iteratively improved care set
- Builds upon CUDD and the Omega Library
  - BDDs to track control flow logic
  - Polyhedral powersets to represent numerical constraints

Preliminary Experiments

- comparison: reachability fixpoint computation on C programs

Program name	Analysis Result				Total CPU Time (s)			
	widen only	extra refine	MIX m.c.	BDD m.c.	widen only	extra refine	MIX m.c.	BDD m.c.
bakery	?	true	true	true	18	5	13	2
tcas-1	?	true	true	true	18	34	128	433
tcas-2	?	true	true	true	18	37	132	644
tcas-3	?	true	true	true	18	49	135	433
tcas-4	?	true	true	true	18	19	137	212
tcas-5	?	false	false	false	18	80	150	174
appl-a	true	true	?	?	17	22	$>$1800	$>$1800
appl-b	?	false	false	?	11	94	277	$>$1800
appl-c	?	false	false	?	13	111	80	$>$1800
appl-d	?	false	false	?	13	68	78	$>$1800

```
tcas:  from traffic alert and collision avoidance system
appl:  from embedded software on a portable device
```

Conclusions

- We use CEGAR to improve the precision of extrapolation in abstract interpretation by iteratively expanding a care set
  - algorithms for computing *extrapolation with a care set* in polyhedral domain
- Promising results: can retain scalability of static analysis and at the same time improve precision
- Future work: apply to other abstract domains (e.g. octagon and interval domains)