Generating Sharper and Simpler Nonlinear Interpolants for Program Verification

Takamasa Okudono[1], Yuki Nishida[2], Kensuke Kojima[2], Kohei Suenaga[2], Kengo Kido[1] and Ichiro Hasuo[3]

[1]University of Tokyo, Japan

[2]Kyoto University, Japan

[3]National Institute of Informatics, Japan

APLAS 2017, Suzhou, China. November 28th 2017

Purpose of This Work

Interpolant is effective at program verification

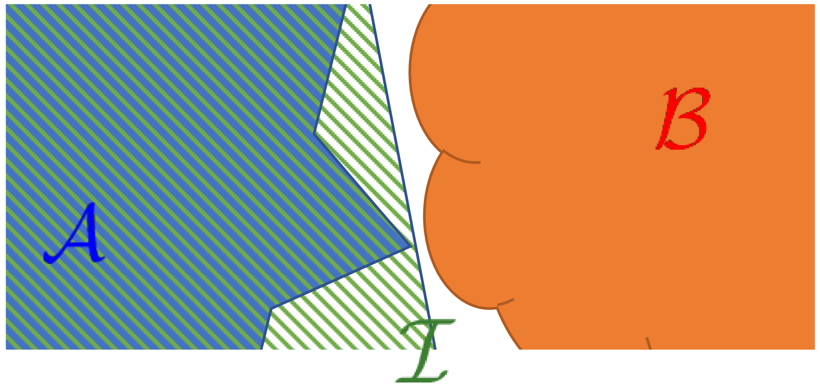
- Automatic generation of polynomial interpolants.

Disjointness of the regions

Def. [interpolant]

- A, B: Formulas satisfying $\vDash \neg(A \wedge B)$.
- Formula I is an *interpolant* of A and B if:
  1. $\vDash A \to I$
  2. $\vDash \neg(B \wedge I)$
  3. Variables in I appear in both of A, B

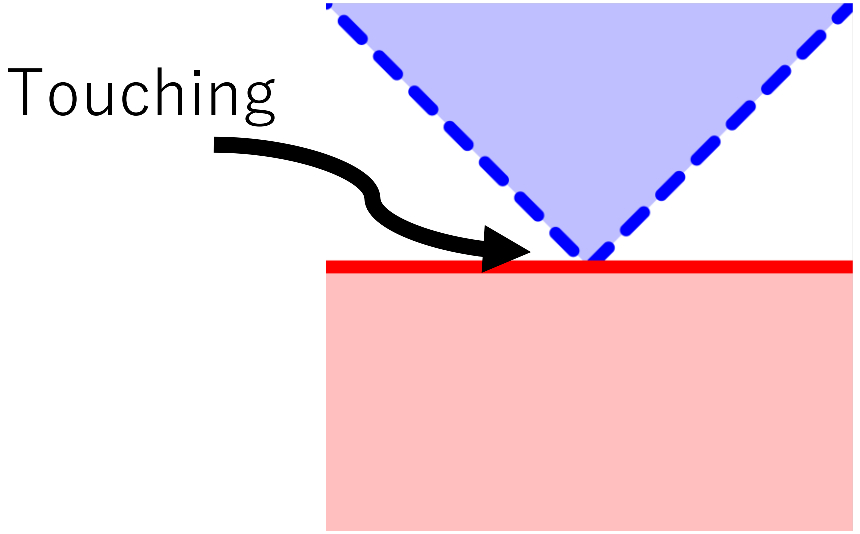
Essential to separate the regions



For polynomial interpolants, atomic propositions are:
(Poly.)≧0, (Poly.)>0, (Poly.)=0

Existing Work

- [Dai+, CAV'13]: generation of polynomial interpolants with numerical optimization
  - Challenge 1: Unable to generate any interpolants in "*touching*" cases
  - Challenge 2: Incorrect and complex due to numerical errors

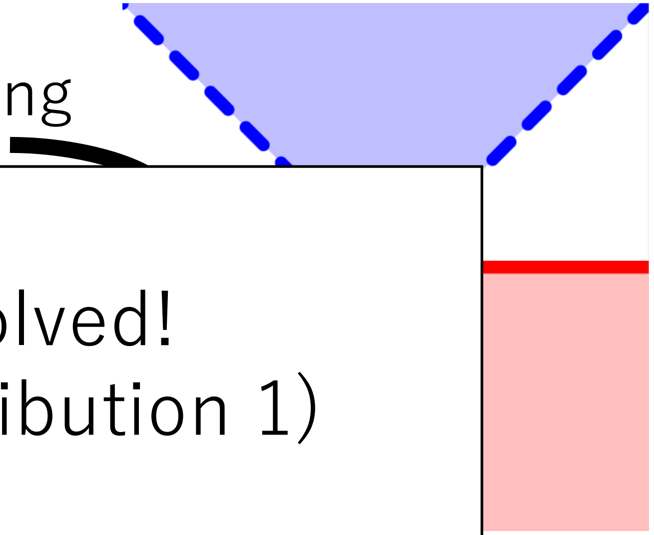


$$54.1800x + 108.3601y \geq 0$$

$$x + 2y \geq 0$$

Our Contribution

- [Dai+, CAV'13]: generation of polynomial interpolants with numerical optimization
  - Challenge 1: Unable to generate any interpolants in "*touching*" cases
  - Challenge 2: Incorrect and complex due to numerical errors



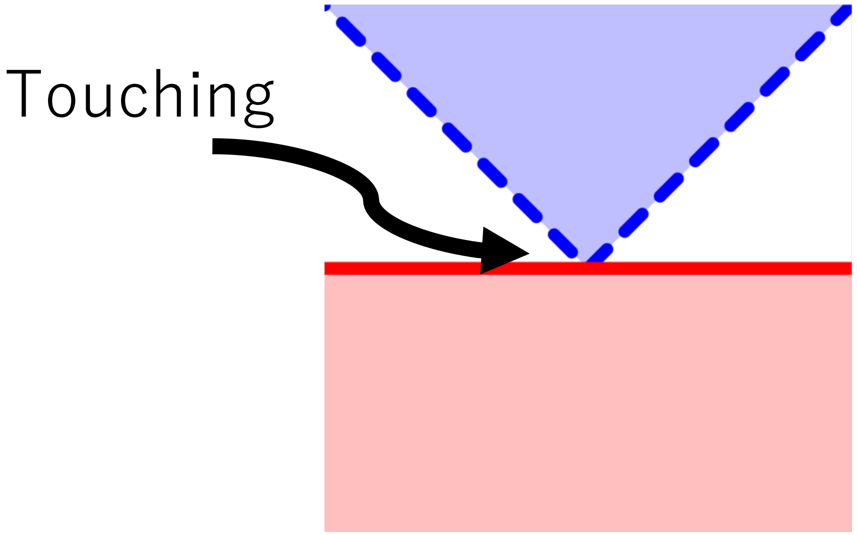
$54.1800\ldots\ 108.3601y \geq 0$

Solved!
(Contribution 1)

Solved!
(Contribution 2)

Challenge 1 in [Dai+]: Sharpness

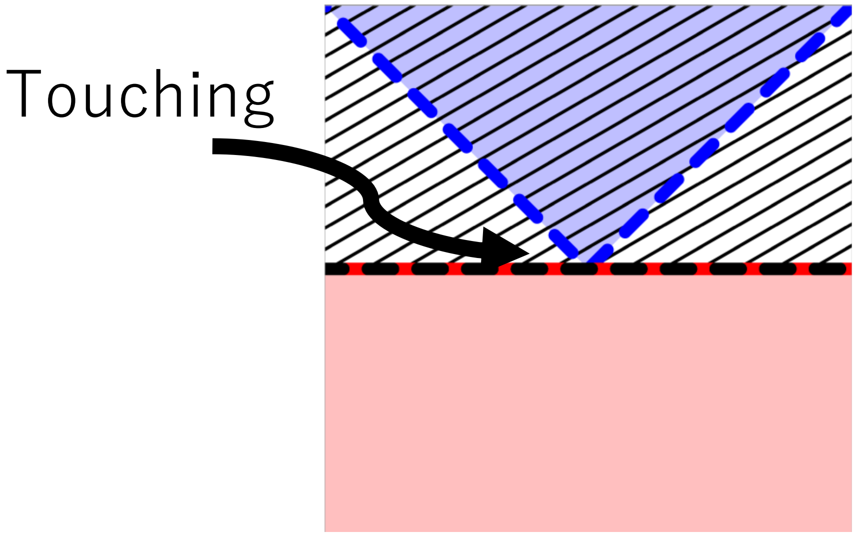
- If two regions of A, B are "touching", [Dai+, CAV'13] always fails at generating their interpolant.



- $A = (y - x > 0 \land y + x > 0)$
- $B = (-y \geq 0)$

Challenge 1 in [Dai+]: Sharpness

- If two regions of A, B are "touching", [Dai+, CAV'13] always fails at generating their interpolant.



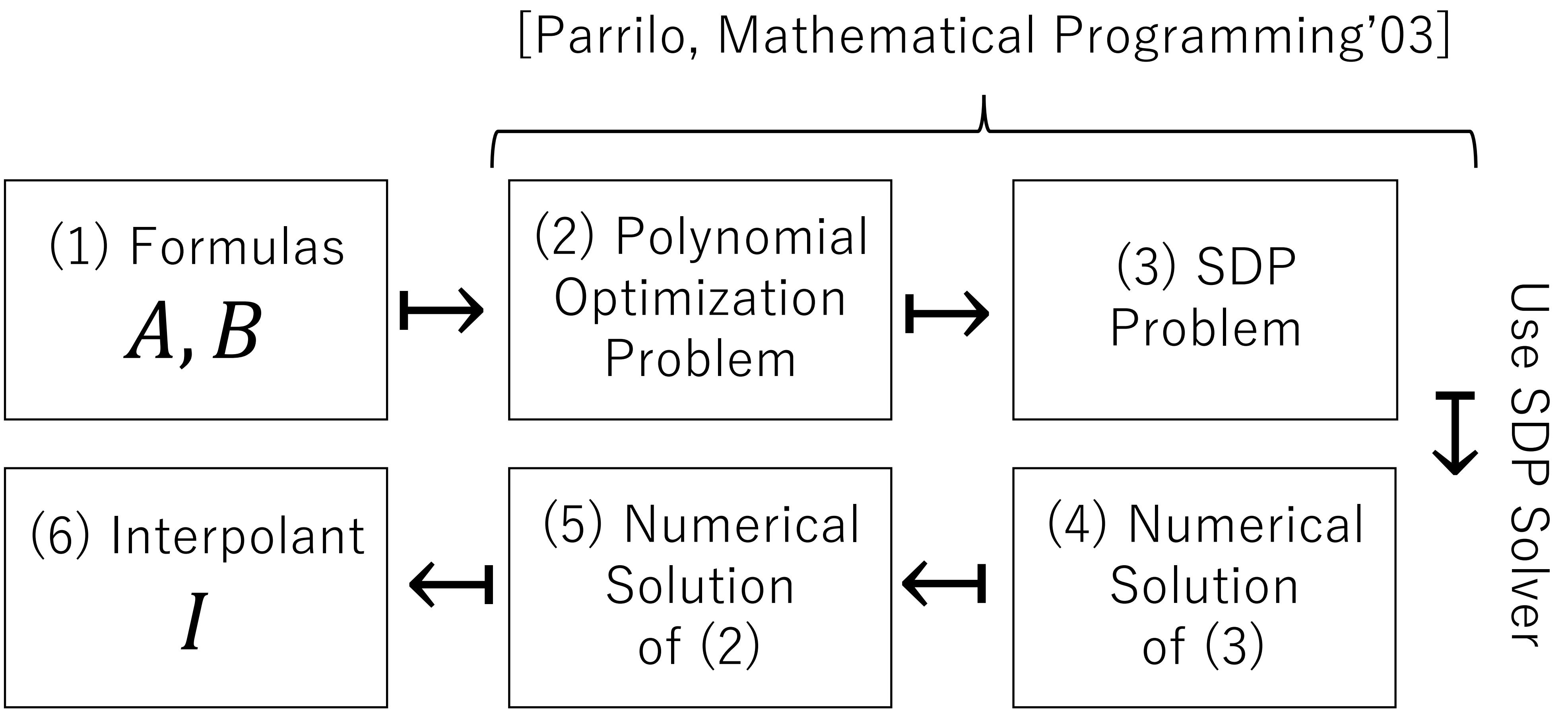
- $A = (y - x > 0 \land y + x > 0)$
- $B = (-y \geq 0)$
- $I = (y > 0)$
- There is an interpolant, but [Dai, CAV'13] cannot find it!

Challenge 1 in [Dai+]: Sharpness

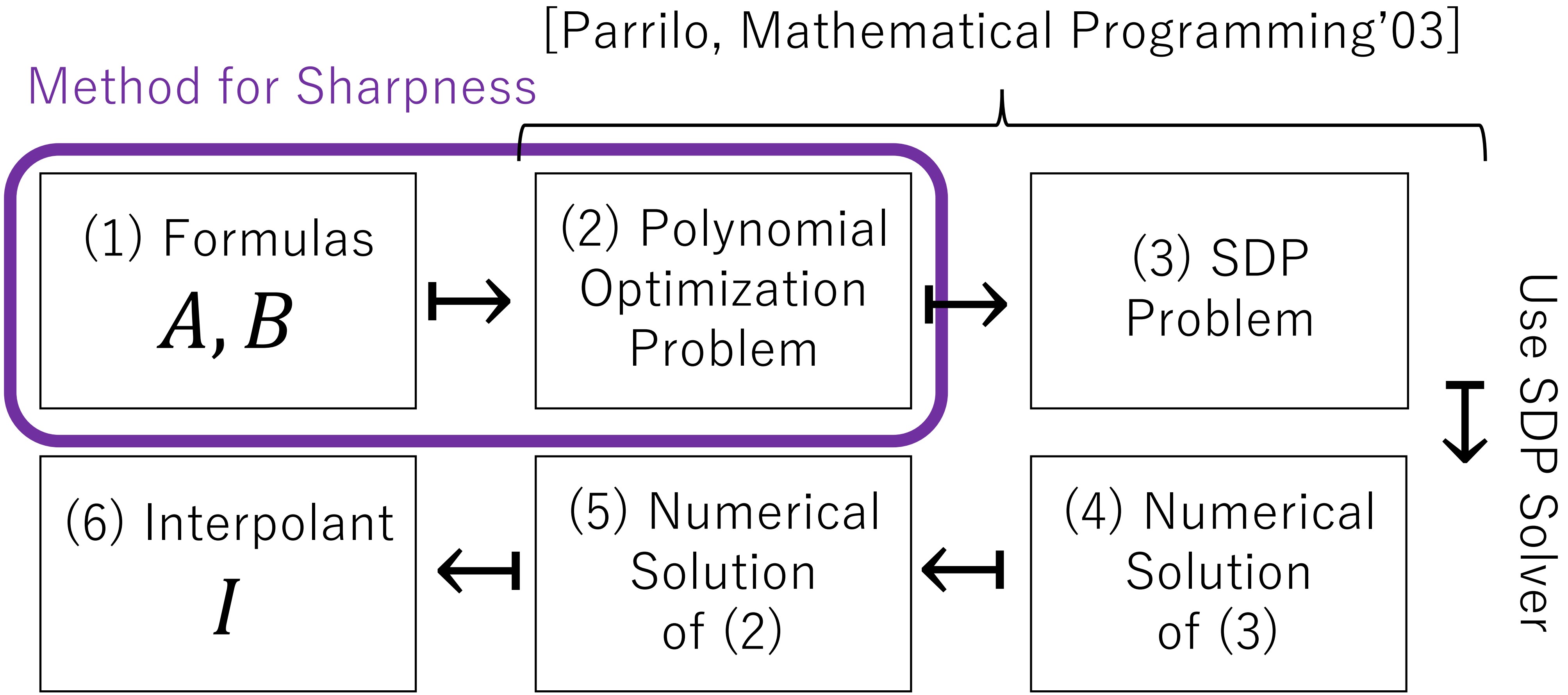
- If two regions of A, B are "touching", [Dai+, CAV'13] **always** fails at generating their interpolant.

Proposition 3.3 *Let $\mathcal{T}$ and $\mathcal{T}'$ be the SAS$_{\neq}$'s in (2). If $\mathcal{T}$ and $\mathcal{T}'$ are barely disjoint (in the sense of Def. 3.2), there do not exist polynomials $\widetilde{f} \in \mathcal{C}(\overrightarrow{f}, \overrightarrow{f'})$, $g \in \mathcal{M}(\overrightarrow{g}, \overrightarrow{g'})$ and $\widetilde{h} \in \mathcal{I}(\overrightarrow{h}, \overrightarrow{h'})$ such that $1 + \widetilde{f} + g^2 + \widetilde{h} = 0$.* □

Challenge 1: Flow of [Dai+]



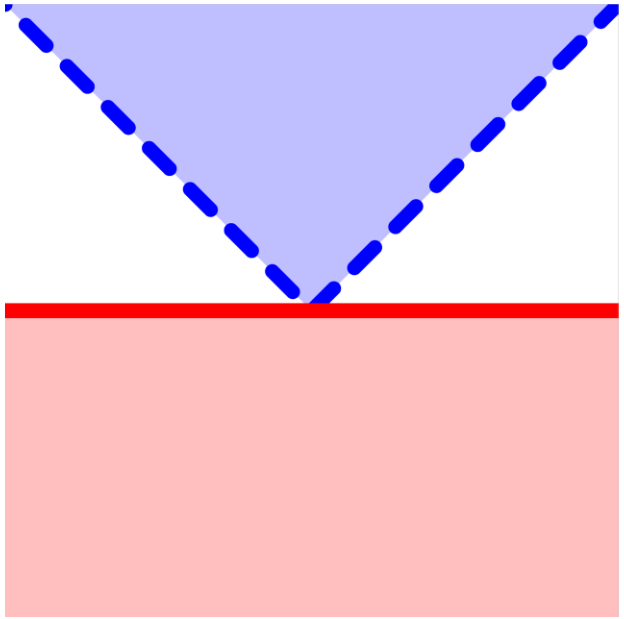
Contribution 1: Method for Sharpness



Contribution 1: Example

Method for Sharpness

(1) Formulas A, B $\mapsto$ (2) Polynomial Optimization Problem



$$A = (\mathrm{y} - \mathrm{x} > 0, \mathrm{y} + \mathrm{x} > 0),$$
$$B = (-y \geq 0)$$

Contribution 1: Example [Dai+, CAV'13]

- $A = (\mathrm{y} - \mathrm{x} > 0, \mathrm{y} + \mathrm{x} > 0), B = (-y \geq 0)$
- [Dai+, CAV'13]
  - Find polynomials $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5 \in \mathbb{R}[X]$ s.t.
    - $I \coloneqq \frac{1}{2} + \sigma_1 + \sigma_2(y - x) + \sigma_3(y + x) + \sigma_4(y - x)(y + x) + (y - x)^2(y + x)^2$
    - $I' \coloneqq \frac{1}{2} + \sigma_5(-y)$
    - $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are sums of squares
    - $I + I' = 0$
    - (I contains only y)
  - Then $I > 0$ is an interpolant

σ is a sum of squares

$\Leftrightarrow$

$\exists \varphi_1, \dots, \varphi_n \in \mathbb{R}[X]; \sigma = \varphi_1^2 + \cdots + \varphi_n^2$

Contribution 1: Example [Dai+, CAV'13]

- $A = (\mathrm{y} - \mathrm{x} > 0, \mathrm{y} + \mathrm{x} > 0), B = (-y \geq 0)$
- [Dai+, CAV'13]
  - Find polynomials $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5 \in \mathbb{R}[X]$ s.t.
    - $I \coloneqq \frac{1}{2} + \sigma_1 + \sigma_2(y - x) + \sigma_3(y + x) + \sigma_4(y - x)(y + x) + (y - x)^2(y + x)^2$
    - $I' \coloneqq \frac{1}{2} + \sigma_5(-y)$
    - $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are sums of squares
    - $I + I' = 0$
    - (I contains only y)
  - Then $I > 0$ is an interpolant

Infeasible and unable to generate any interpolants!

Contribution 1: Example [Dai+, CAV'13]

- $A = (y - x > 0, y + x > 0)$, $B = (-y \geq 0)$
- [Dai+, CAV'13]
  - Find $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5 \in \mathbb{R}[X]$ s.t.
    - $I := \frac{1}{2} + \sigma_1 + \sigma_2(y - x) + \sigma_3(y + x) + \sigma_4(y - x)(y + x) + (y - x)^2(y + x)^2$
    - $I' := \frac{1}{2} + \sigma_5(-y)$
    - $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are sums of squares
    - $I + I' = 0$
    - (I contains only y)
  - Then $I > 0$ is an interpola

∵ Assume the feasibility.

$0 = (I + I')(0,0)$

$= 1 + \sigma_1(0,0) > 0.$

Contradiction. □

Infeasible and unable to generate any interpolants!

Contribution 1: Example [Dai+, CAV'13]

- $A = (\mathrm{y} - \mathrm{x} > 0, \mathrm{y} + \mathrm{x} > 0), B = (-y \geq 0)$
- Our method for sharpness
  - Find polynomials $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5 \in \mathbb{R}[X]$and $r_1, r_2, r_3 \in \mathbb{R}_{\geq 0}$ s.t.
    - $I \coloneqq \sigma_1 + \sigma_2(y - x) + \sigma_3(y + x) + \sigma_4(y - x)(y + x) + r_1 + r_2(y - x) + r_3(y + x)$
    - $I' \coloneqq \sigma_5(-y)$
    - $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are sums of squares
    - $r_1 + r_2 + r_3 > 0$
    - $I + I' = 0$
    - I contains only y
  - Then $I > 0$ is an interpolant

Contribution 1: Example [Dai+, CAV'13]

- $A = (y$ … $0,$ … 0 … $-y \ge 0)$
- … method for sharpness
  - Find polynomials $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5 \in \mathbb{R}[X]$ and $r_1, r_2, r_3 \in \mathbb{R}_{\geq 0}$ s.t.
    - $I \coloneqq \sigma_1 + \sigma_2(y-x) + \sigma_3(y+x) + \sigma_4(y-x)(y+x) + r_1 + r_2(y-x) + r_3(y+x)$
    - $I' \coloneqq \sigma_5(-y)$
    - $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are sums of squares
    - $r_1 + r_2 + r_3 > 0$
    - $I + I' = 0$
    - I contains only y
  - Then $I > 0$ is an interpolant

$\sigma_1 = 0$

$\sigma_2 = 0$

$\sigma_3 = 1$

$\sigma_4 = 0$

$r_1 = 0$

$r_2 = 1$

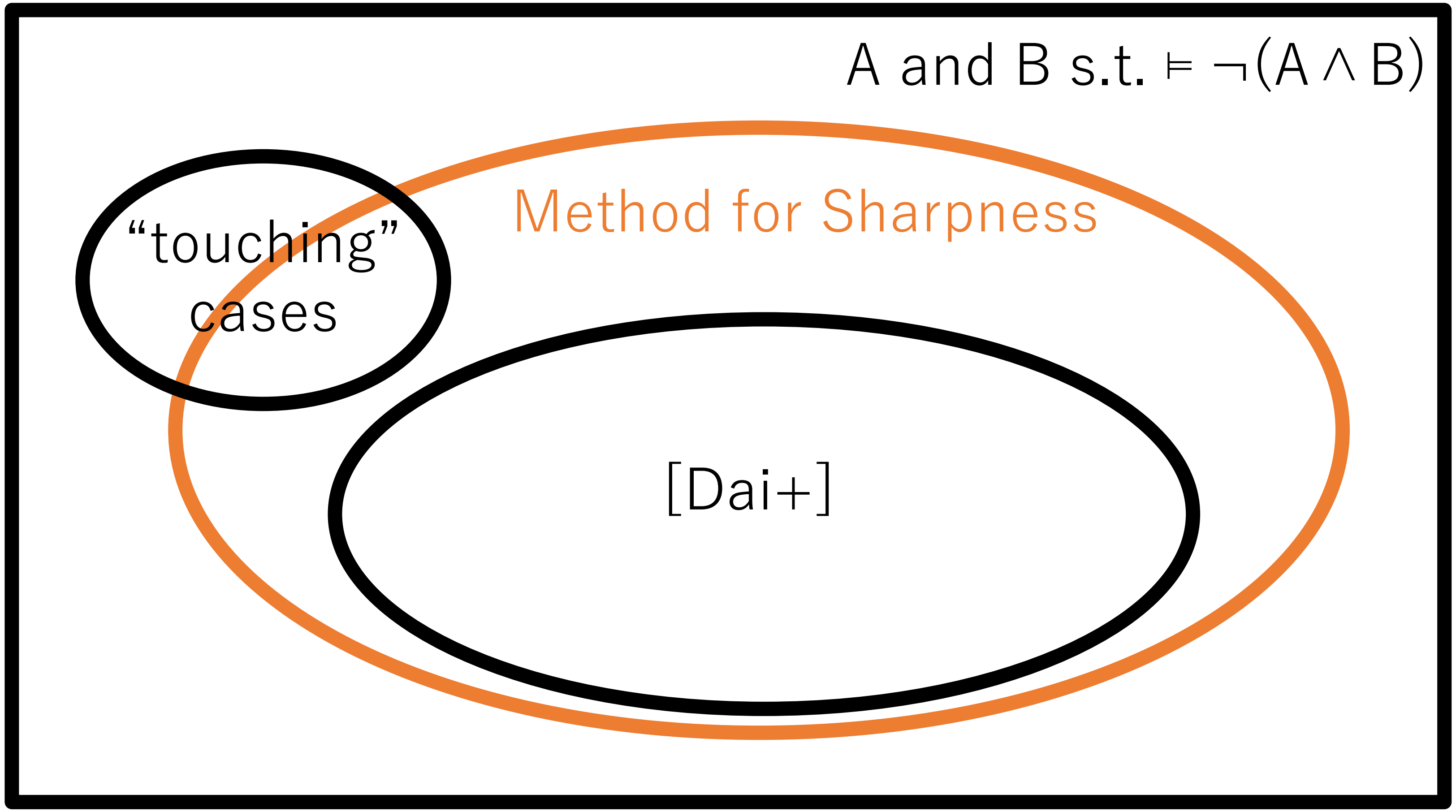
$r_3 = 0$

$r_5 = 2$

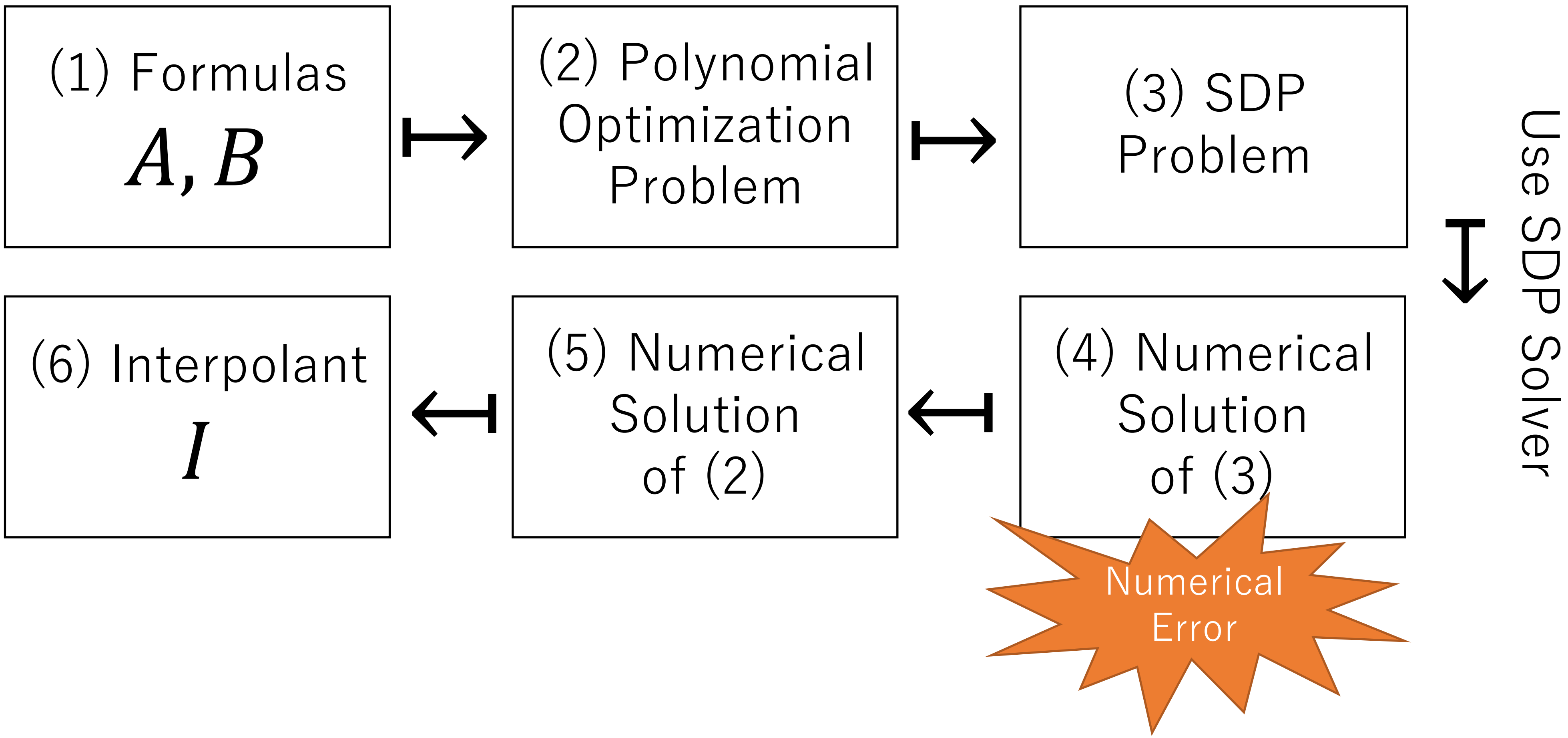
$I = 2y$

Feasible and able to generate an interpolant!

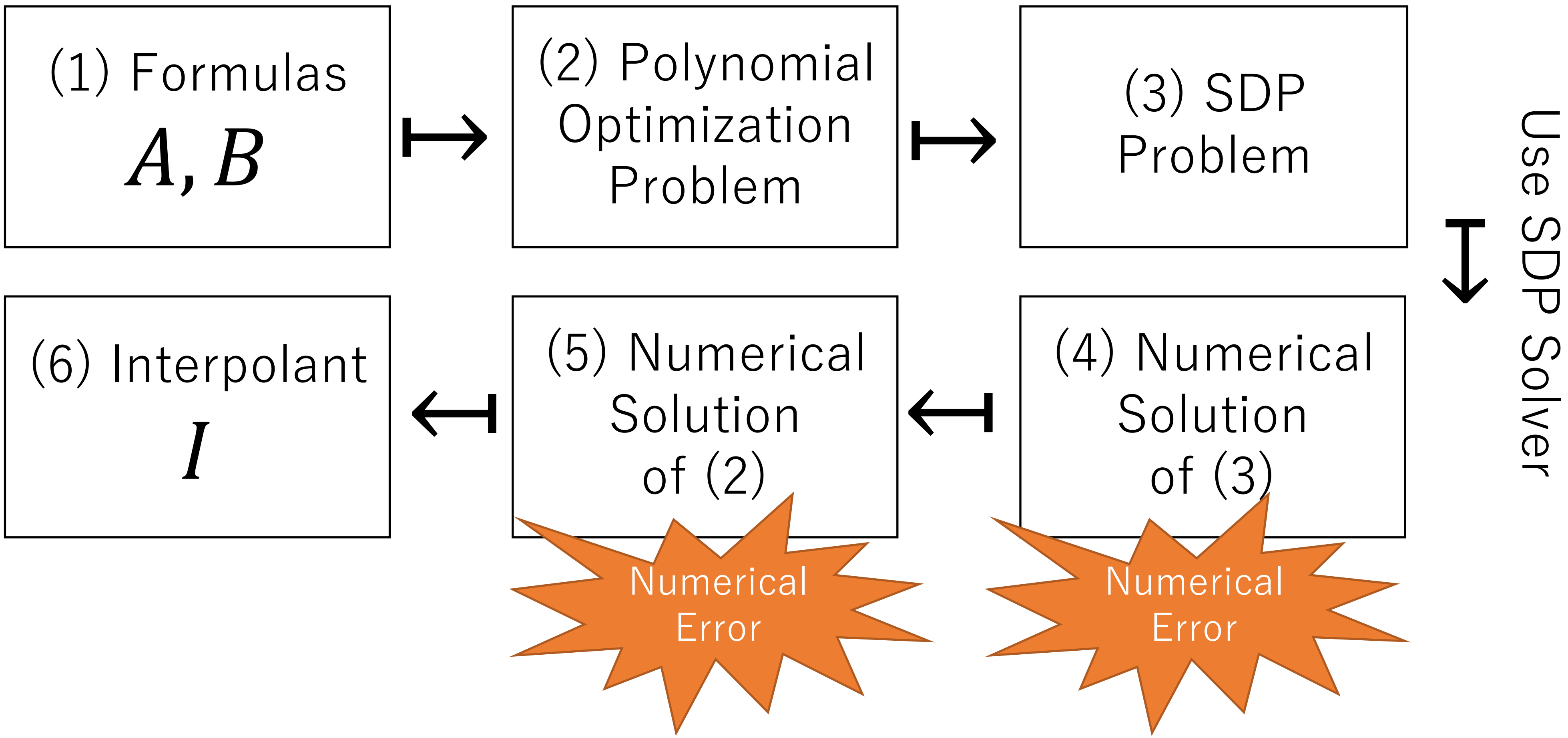
Contribution 1: Completeness



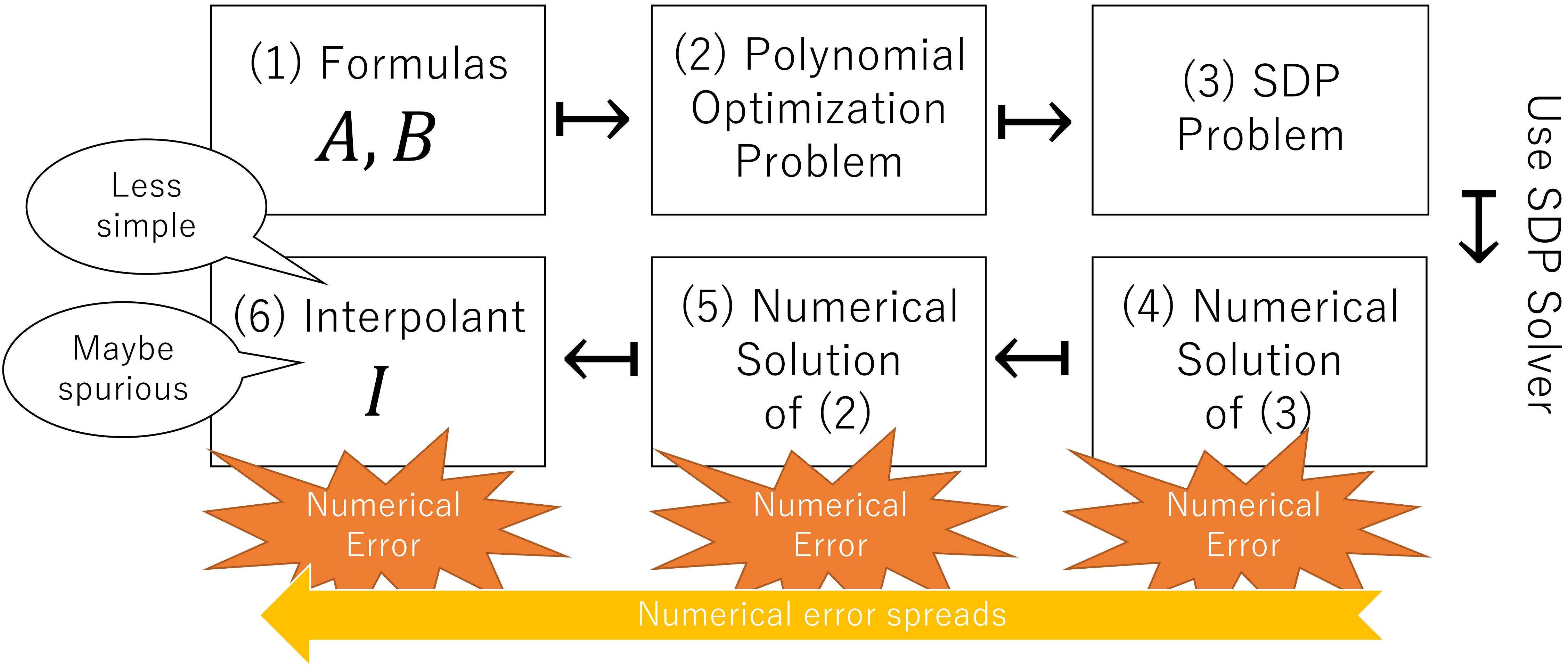
Challenge 2: Numerical Error in [Dai+]



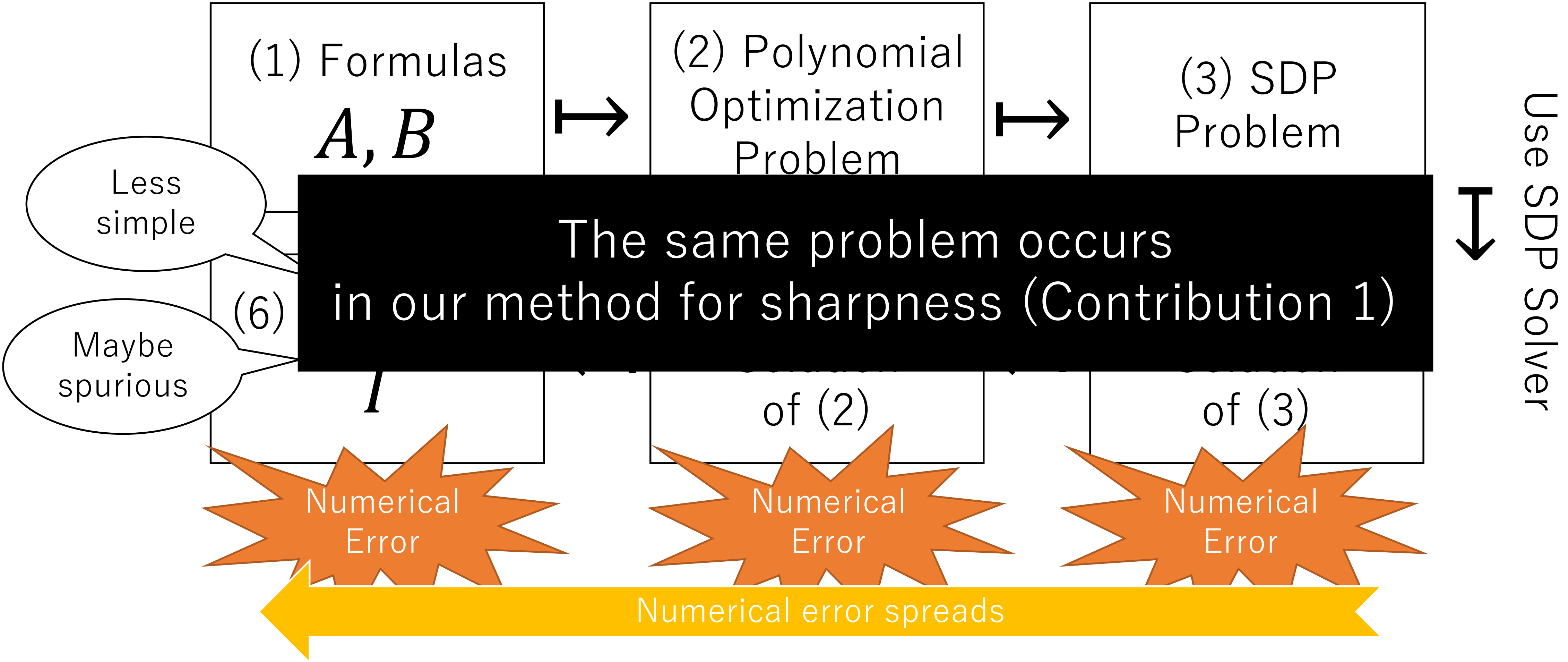
Challenge 2: Numerical Error in [Dai+]



Challenge 2: Numerical Error in [Dai+]



Challenge 2: Numerical Error in [Dai+]



The same problem occurs
in our method for sharpness (Contribution 1)

Challenge 2: Example

- Example: $A = (x = 0 \wedge y = 0), B = (x + 2y < 0)$
- Spurious interpolant $I = (54.1800x + 108.3601y \geq 0)$
  - $(x, y) = (-108.3601, 54.1800)$ satisfies both B and I

Def. [interpolant]

- A, B: Formulas satisfying $\vDash (A \wedge B)$.
- Formula I is an *interpolant* of A and B if:
  1. $\vDash A \rightarrow I$
  2. $\vDash \neg(B \wedge I)$
  3. Variables in I appears in both of A, B

Contribution 2: Observation

Spurious Interpolant: $I = (54.1800x + 108.3601y \geq 0)$

$\approx \times 2$

Simplified Interpolant: $I = (x + 2y \geq 0)$

Correct and simple interpolant of
$A = (x = 0 \wedge y = 0), B = (x + 2y < 0)$

Contribution 2: Working Assumption

- Working Assumption: Simple interpolants tend to be correct and useful to capture the program's nature.
- Strategy:
  - Simplify the ratio that appears in the interpolant
  - Find and guess simple integers

Contribution 2: Technique

- Continued Fraction Expansion
  - Input: real number x
  - Output: “best” approximations $\frac{a_1}{b_1}, \frac{a_2}{b_2}, \ldots$ of x
- Example:
  - $3.1416 = 3 + \frac{1}{7+\frac{1}{16+\frac{1}{11}}}$
  - 1st approximation: $3.1416 \simeq 3$
  - 2nd approximation: $3.1416 \simeq 3 + \frac{1}{7} = \frac{22}{7}$
  - 3rd approximation: $3.1416 \simeq 3 + \frac{1}{7+\frac{1}{16}} = \frac{355}{113}$

Contribution 2: Technique

- Continued Fraction Expansion
  - Input: real number x
  - Output: "best" approximations $\frac{a_1}{b_1}, \frac{a_2}{b_2}, \ldots$ of x
- Example:
  - $3.1416 = 3 + \cfrac{1}{7 + \cfrac{1}{16 + \cfrac{1}{11}}}$ — $(3.1416 : 1)$
  - 1st approximation: $3.1416 \simeq 3$ — $(3 : 1)$
  - 2nd approximation: $3.1416 \simeq 3 + \frac{1}{7} = \frac{22}{7}$ — $(22 : 7)$
  - 3rd approximation: $3.1416 \simeq 3 + \cfrac{1}{7 + \cfrac{1}{16}} = \frac{355}{113}$ — $(355 : 113)$

Simple ↕ Faithful

Contribution 2: Simplification of Ratio

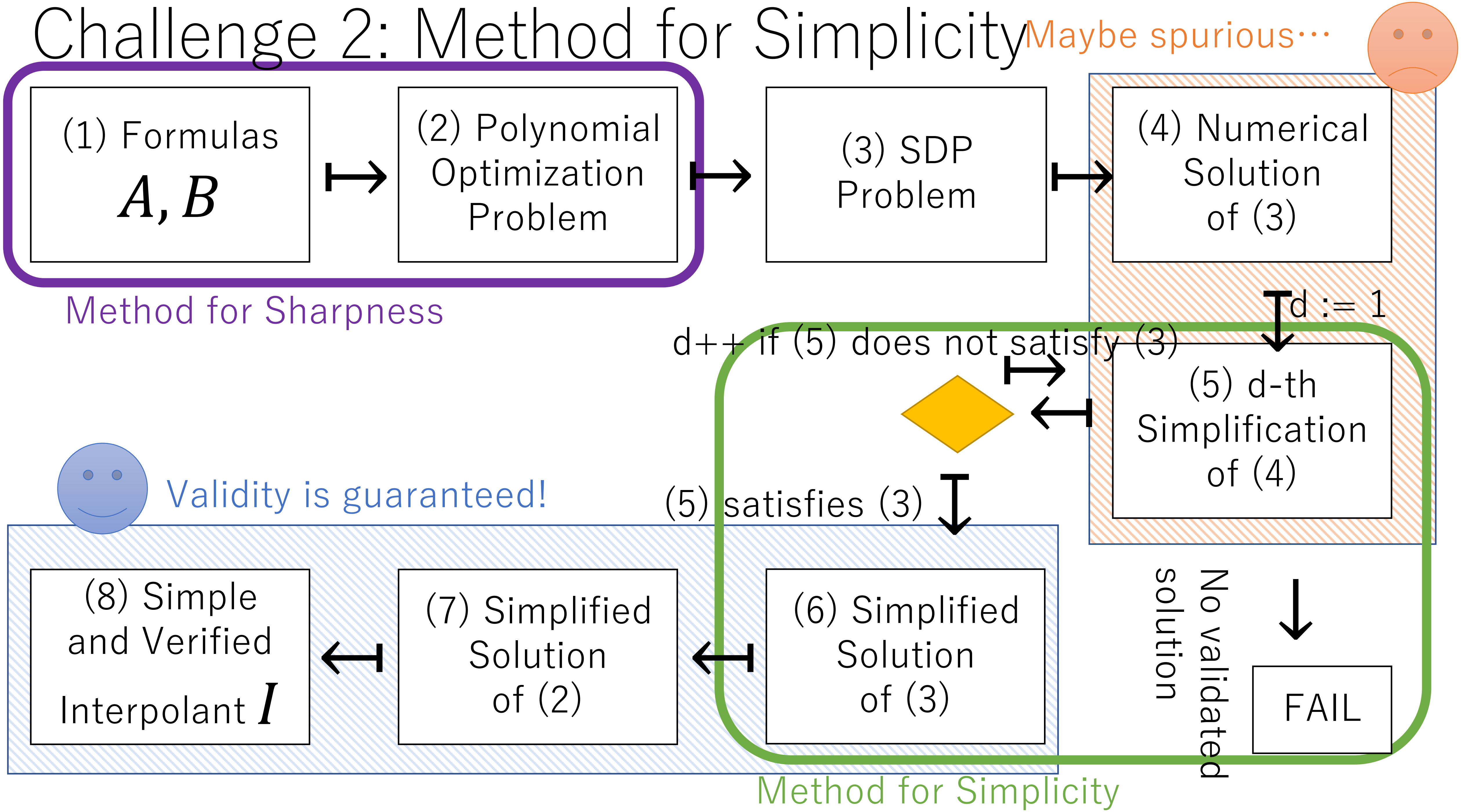
- The simplification starts from the simplest ratio
- Make it more faithful to the original solution and less simple iteratively.

Original

More Faithful
Less Simple

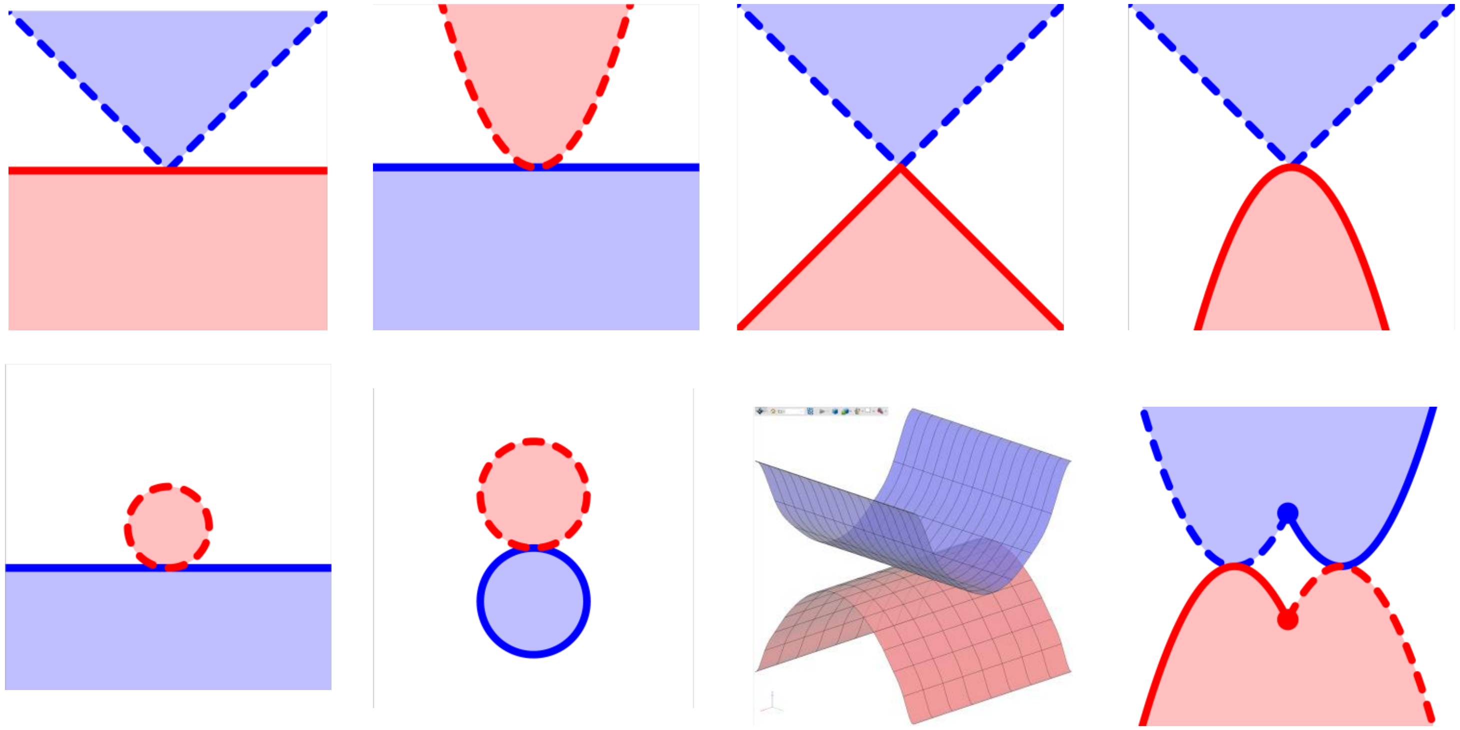
46.7375	155.0975	60.1733
1	3	1
3	10	4
31	103	40
97	322	125
…	…	…
467375	1550975	601733

Challenge 2: Method for Simplicity



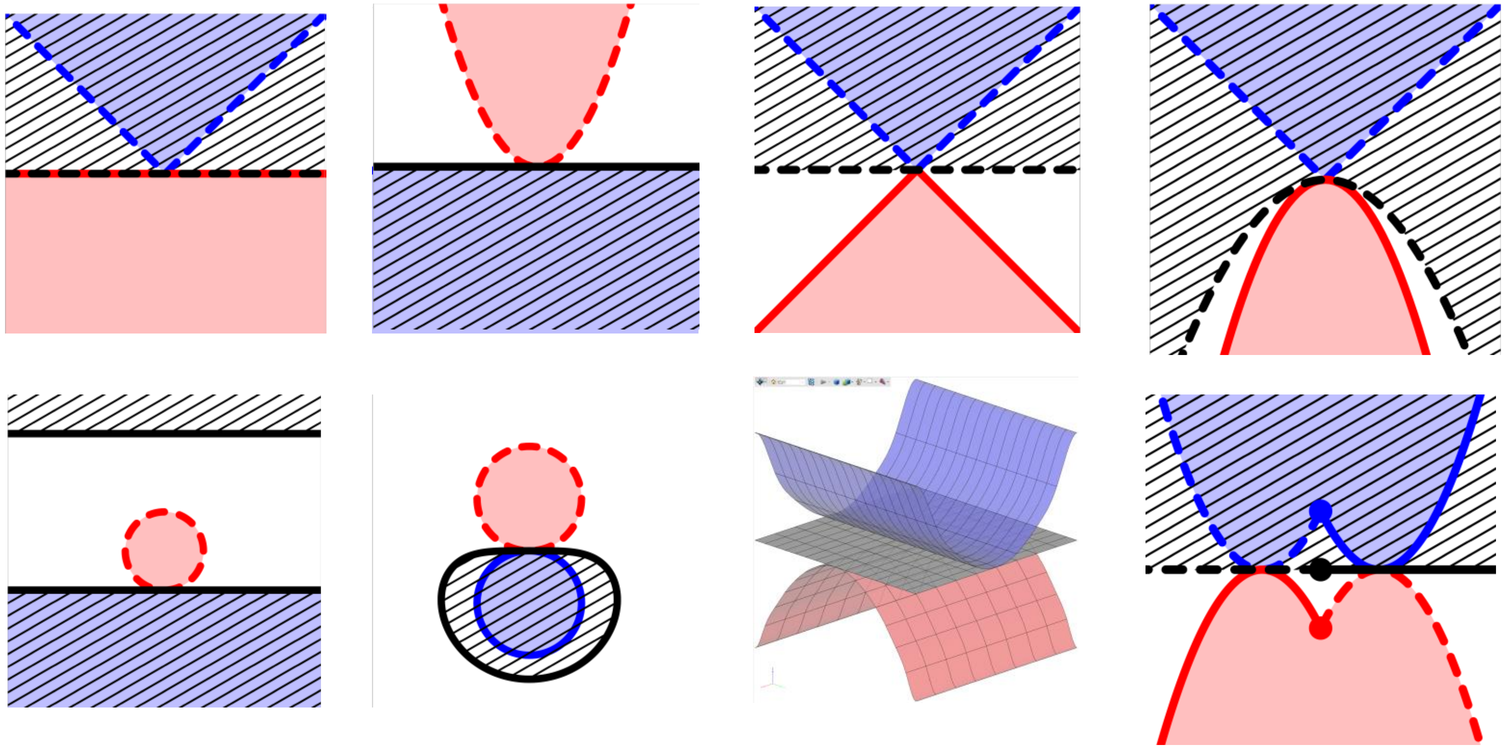
Experiments: Geometric Examples

- A: blue regions, B: red regions, I: black hatched regions.



Experiments: Geometric Examples

- A: blue regions, B: red regions, I: black hatched regions.



Experiments: Program Examples

- These examples are rather simple, but [Dai+, CAV'13] cannot verify them because of numerical errors (Challenge 2).

Listing 1.1. Code 1.3 of [8]

```
real x,y;
real xa = 0;
real ya = 0;
while(nondet()){
 x = xa + 2*ya;
 y = -2*xa + ya;
 x++;
 if(nondet()){
  y = y + x;
 }else{
  y = y - x;
 }
 xa = x - 2*y;
 ya = 2*x + y;
}
assert(xa + 2*ya >= 0);
```

Listing 1.2. Constant Acceleration

```
real x,v;
(x, v) = (0, 0);
while(nondet()){
 (x, v) = (x+2*v, v+2);
}
 assert(x >= 0);
```

Experiments: Program Examples

```
(x, v) = (0, 0)
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Abstraction with $x \geq 0$ and $v \geq 0$

(x, v) = $[x \geq 0, v \geq 0]$
```
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Find good predicates
by CEGAR[Clarke+, CAV'00] and our interpolant generation

Experiments: Program Examples

```
(x, v) = (0, 0)
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Abstraction with nothing

```
(x, v) = [anything]
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Find Counterexample

```
(x, v) = [anything];
 Assert(x >= 0)
```

Extract the original trace

```
(x, v) = (0, 0);
Assert(x >= 0)
```

Make Formulas

$$A = (x = 0, v = 0)$$
$$B = (x < 0)$$

Touching

Generate an interpolant

$$I = (x \geq 0)$$

Add $x \geq 0$

Experiments: Program Examples

(x, v) = (0, 0)
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)

Abstraction with $x \geq 0$

(x, v) = [$x \geq 0$, v: any]
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)

Find Counterexample

(x, v) = [$x \geq 0$, v: any];
(x, v) = (x + 2*v, v + 2);
Assert(x >= 0)

Extract the original trace

(x, v) = (0, 0);
(x, v) = (x + 2*v, v + 2);
Assert(x >= 0)

Make Formulas

$$A = (v_1 = 0)$$
$$B = (x_1 = 0 \land v_2 = v_1 + 2 \land x_2 = x_1 + 2v_1 \land x_2 < 0)$$

Generate an interpolant

$$I = (v_1 \geq 0)$$

Add $v \geq 0$

Experiments: Program Examples

```
(x, v) = (0, 0)
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Abstraction with
$x \geq 0$ and $v \geq 0$

```
(x, v) = [x ≥ 0, v ≥ 0]
While(nondet()){
  (x, v) = (x + 2*v, v + 2);
}
Assert(x >= 0)
```

Our Challenge

- Our method works only for fairly simple examples:
  - Geometric examples: at most quadratic
  - Program examples: at most linear

Conclusion

- Our Contributions: Solved some challenges in [Dai+, CAV'13]
    - Challenge 1: Sharpness
    - Challenge 2: Numerical Error
- Our method works only for fairly simple examples:
    - Geometric examples: at most quadratic
    - Program examples: at most linear