Reduced Product in Abstract Interpretation

František Nečas

November 14, 2023

Motivation

- A single abstract domain may not be sufficient for analysis.
- Two possible solutions:

Motivation

- A single abstract domain may not be sufficient for analysis.
- Two possible solutions:
  - Create a single more **complex/universal domain** covering more properties of the program.

Motivation

- A single abstract domain may not be sufficient for analysis.
- Two possible solutions:
  - Create a single more **complex/universal domain** covering more properties of the program.
  - Use multiple **specialized domains** in parallel and combine their results.

Motivation

- A single abstract domain may not be sufficient for analysis.
- Two possible solutions:
  - Create a single more **complex/universal domain** covering more properties of the program.
  - Use multiple **specialized domains** in parallel and combine their results.
- Creating a universal domain is a complex task. Combination of multiple simpler domains is more feasible.

Motivation

- A single abstract domain may not be sufficient for analysis.
- Two possible solutions:
  - Create a single more **complex/universal domain** covering more properties of the program.
  - Use multiple **specialized domains** in parallel and combine their results.
- Creating a universal domain is a complex task. Combination of multiple simpler domains is more feasible.
- Results of one domain can refine the results of another domain.

Reduced product

- For simplicity, let's consider only 2 domains.

Reduced product

- For simplicity, let's consider only 2 domains.
- Let $\langle A_1, \sqsubseteq_1 \rangle$ and $\langle A_2, \sqsubseteq_2 \rangle$ be abstract domains with their concretization functions γ_1 and γ_2, respectively. Their **Cartesian product** [1] is $\langle \boldsymbol{A}, \sqsubseteq \rangle$ where:
  - $\boldsymbol{A} = A_1 \times A_2$
  - $\langle p_1, p_2 \rangle \sqsubseteq \langle q_1, q_2 \rangle \iff p_1 \sqsubseteq_1 q_1 \wedge p_2 \sqsubseteq_2 q_2$
  - $\gamma_{\boldsymbol{A}}(\langle p_1, p_2 \rangle) = \gamma_1(p_1) \cap \gamma_2(p_2)$

Reduced product

- For simplicity, let's consider only 2 domains.
- Let $\langle A_1, \sqsubseteq_1 \rangle$ and $\langle A_2, \sqsubseteq_2 \rangle$ be abstract domains with their concretization functions γ_1 and γ_2, respectively. Their **Cartesian product** [1] is $\langle \boldsymbol{A}, \sqsubseteq \rangle$ where:
  - $\boldsymbol{A} = A_1 \times A_2$
  - $\langle p_1, p_2 \rangle \sqsubseteq \langle q_1, q_2 \rangle \iff p_1 \sqsubseteq_1 q_1 \wedge p_2 \sqsubseteq_2 q_2$
  - $\gamma_{\boldsymbol{A}}(\langle p_1, p_2 \rangle) = \gamma_1(p_1) \cap \gamma_2(p_2)$
- Such combined abstract domain does not provide more precise results than running the analyses with each abstract domain independently [2].

Reduced product

- For simplicity, let's consider only 2 domains.
- Let $\langle A_1, \sqsubseteq_1 \rangle$ and $\langle A_2, \sqsubseteq_2 \rangle$ be abstract domains with their concretization functions γ_1 and γ_2, respectively. Their **Cartesian product** [1] is $\langle \boldsymbol{A}, \sqsubseteq \rangle$ where:
  - $\boldsymbol{A} = A_1 \times A_2$
  - $\langle p_1, p_2 \rangle \sqsubseteq \langle q_1, q_2 \rangle \iff p_1 \sqsubseteq_1 q_1 \wedge p_2 \sqsubseteq_2 q_2$
  - $\gamma_{\boldsymbol{A}}(\langle p_1, p_2 \rangle) = \gamma_1(p_1) \cap \gamma_2(p_2)$
- Such combined abstract domain does not provide more precise results than running the analyses with each abstract domain independently [2].
- The **reduced product** is $\langle \boldsymbol{A}/_{\equiv}, \sqsubseteq \rangle$ where $P \equiv Q \iff \gamma_{\boldsymbol{A}}(P) = \gamma_{\boldsymbol{A}}(Q)$ and $\gamma_{\boldsymbol{A}}$ and $\sqsubseteq$ are extended to the equivalence classes of $\equiv$.

Reduced product

- Finding the equivalence class of an abstract context can be seen as using a **reduction function** $\sigma : \boldsymbol{A} \rightarrow \boldsymbol{A}$ such that $\sigma(\langle p_1, p_2 \rangle) = \langle \alpha_1(\gamma_{\boldsymbol{A}}(\langle p_1, p_2 \rangle)), \alpha_2(\gamma_{\boldsymbol{A}}(\langle p_1, p_2 \rangle)) \rangle$

Reduced product

- Finding the equivalence class of an abstract context can be seen as using a **reduction function** $\sigma : \boldsymbol{A} \to \boldsymbol{A}$ such that $\sigma(\langle p_1, p_2\rangle) = \langle \alpha_1(\boldsymbol{\gamma_A}(\langle p_1, p_2\rangle)), \alpha_2(\boldsymbol{\gamma_A}(\langle p_1, p_2\rangle))\rangle$
- For example, in a reduced product of interval and parity domain, $\langle [1, 9], \mathit{even}\rangle \equiv \langle [2, 8], \mathit{even}\rangle$:

$$\begin{aligned}
\boldsymbol{\gamma_A}(\langle [1,9], \mathit{even}\rangle) &= \gamma_1([1,9]) \cap \gamma_2(\mathit{even}) \\
&= \{1, 2, \ldots, 9\} \cap \{0, 2, 4, \ldots\} \\
&= \{2, 4, 6, 8\} \\
&= \boldsymbol{\gamma_A}(\langle [2,8], \mathit{even}\rangle)
\end{aligned}$$

Reduced product

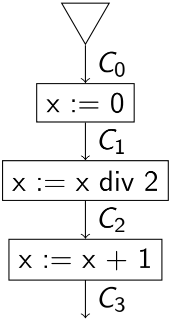
- Finding the equivalence class of an abstract context can be seen as using a **reduction function** $\sigma : \boldsymbol{A} \to \boldsymbol{A}$ such that $\sigma(\langle p_1, p_2\rangle) = \langle \alpha_1(\boldsymbol{\gamma_A}(\langle p_1, p_2\rangle)), \alpha_2(\boldsymbol{\gamma_A}(\langle p_1, p_2\rangle))\rangle$
- For example, in a reduced product of interval and parity domain, $\langle [1, 9], \mathit{even}\rangle \equiv \langle [2, 8], \mathit{even}\rangle$:

$$\begin{aligned}
\boldsymbol{\gamma_A}(\langle [1, 9], \mathit{even}\rangle) &= \gamma_1([1, 9]) \cap \gamma_2(\mathit{even}) \\
&= \{1, 2, \ldots, 9\} \cap \{0, 2, 4, \ldots\} \\
&= \{2, 4, 6, 8\} \\
&= \boldsymbol{\gamma_A}(\langle [2, 8], \mathit{even}\rangle)
\end{aligned}$$

- In practice, analyzers compute an over-approximation of the reduction using some rules (concretization is not feasible).
- Typically, messages are exchanged between domains, each domain implements refinement based on a received message. The message format varies (e.g. various logics).

Full example [3]

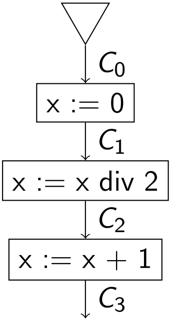
- Consider the parity and sign domains.



- $A_1 = \{\perp, odd, even, \top\}$
- $A_2 = \{\perp, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

Full example [3]

- Consider the parity and sign domains.

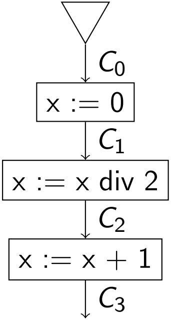


- $A_1 = \{\perp, \mathit{odd}, \mathit{even}, \top\}$
- $A_2 = \{\perp, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1		
C_2		
C_3		

Full example [3]

- Consider the parity and sign domains.

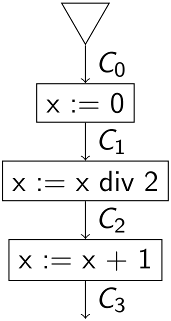


- $A_1 = \{\bot, odd, even, \top\}$
- $A_2 = \{\bot, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	
C_2		
C_3		

Full example [3]

- Consider the parity and sign domains.

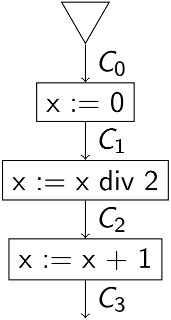


- $A_1 = \{\perp, odd, even, \top\}$
- $A_2 = \{\perp, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	
C_2	$\langle \top, 0 \rangle$	
C_3		

Full example [3]

- Consider the parity and sign domains.

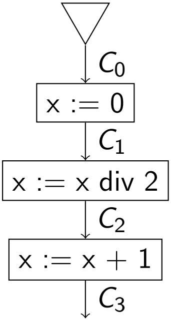


- $A_1 = \{\bot, odd, even, \top\}$
- $A_2 = \{\bot, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	
C_2	$\langle \top, 0 \rangle$	
C_3	$\langle \top, \geq 0 \rangle$	

Full example [3]

- Consider the parity and sign domains.

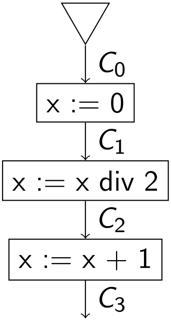


- $A_1 = \{\perp, odd, even, \top\}$
- $A_2 = \{\perp, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	$\langle even, 0 \rangle$
C_2	$\langle \top, 0 \rangle$	
C_3	$\langle \top, \geq 0 \rangle$	

Full example [3]

- Consider the parity and sign domains.

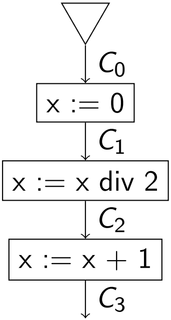


- $A_1 = \{\bot, odd, even, \top\}$
- $A_2 = \{\bot, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	$\langle even, 0 \rangle$
C_2	$\langle \top, 0 \rangle$	$\langle \top, 0 \rangle$
C_3	$\langle \top, \geq 0 \rangle$	

Full example [3]

- Consider the parity and sign domains.

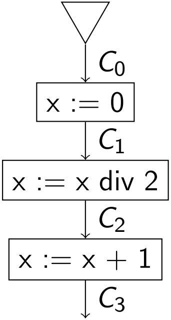


- $A_1 = \{\bot, odd, even, \top\}$
- $A_2 = \{\bot, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	$\langle even, 0 \rangle$
C_2	$\langle \top, 0 \rangle$	$\langle \top, 0 \rangle \equiv \langle even, 0 \rangle$
C_3	$\langle \top, \geq 0 \rangle$	

Full example [3]

- Consider the parity and sign domains.

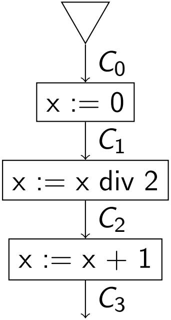


- $A_1 = \{\perp, odd, even, \top\}$
- $A_2 = \{\perp, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	$\langle even, 0 \rangle$
C_2	$\langle \top, 0 \rangle$	$\langle \top, 0 \rangle \equiv \langle even, 0 \rangle$
C_3	$\langle \top, \geq 0 \rangle$	$\langle odd, \geq 0 \rangle$

Full example [3]

- Consider the parity and sign domains.



- $A_1 = \{\bot, odd, even, \top\}$
- $A_2 = \{\bot, \geq 0, 0, \leq 0, \top\}$
- Let's consider $\boldsymbol{A} = A_1 \times A_2$

	Product	Reduced Product
C_0	$\langle \top, \top \rangle$	$\langle \top, \top \rangle$
C_1	$\langle even, 0 \rangle$	$\langle even, 0 \rangle$
C_2	$\langle \top, 0 \rangle$	$\langle \top, 0 \rangle \equiv \langle even, 0 \rangle$
C_3	$\langle \top, \geq 0 \rangle$	$\langle odd, \geq 0 \rangle$

- Notice that we obtain more information in C_3:
  - $\gamma_{\boldsymbol{A}}(\langle \top, \geq 0 \rangle) = \{0, 1, 2, \ldots\}$
  - $\gamma_{\boldsymbol{A}}(\langle odd, \geq 0 \rangle) = \{1, 3, 5, \ldots\}$
- This was a simple sequential example but such reductions can have a positive effect on widening and narrowing as well.

Combining more domains

- The product can naturally be extended to 3 or more domains.

Combining more domains

- The product can naturally be extended to 3 or more domains.
- However, adding a new domain requires **redesigning the reduction** [1].

Combining more domains

- The product can naturally be extended to 3 or more domains.
- However, adding a new domain requires **redesigning the reduction** [1].
- Oftentimes, only **pairwise reductions** are applied. This is easier to implement at the cost of potentially less precise results.

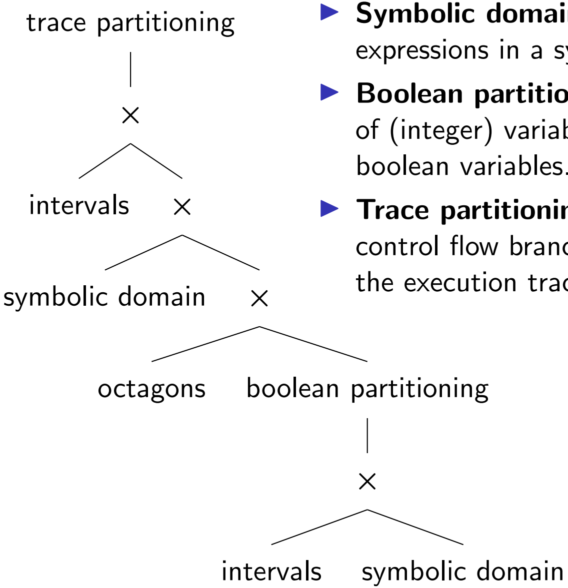
Combining more domains

- The product can naturally be extended to 3 or more domains.
- However, adding a new domain requires **redesigning the reduction** [1].
- Oftentimes, only **pairwise reductions** are applied. This is easier to implement at the cost of potentially less precise results.
- Refinement in one domain can facilitate further refinements. Therefore, the pairwise reductions are applied until a fixpoint is reached [4].

Combining more domains

- The product can naturally be extended to 3 or more domains.
- However, adding a new domain requires **redesigning the reduction** [1].
- Oftentimes, only **pairwise reductions** are applied. This is easier to implement at the cost of potentially less precise results.
- Refinement in one domain can facilitate further refinements. Therefore, the pairwise reductions are applied until a fixpoint is reached [4].
- Alternatively, reductions can be applied in a fixed order, e.g. Astrée [5].

Astrée example hierarchy [5]



- **Symbolic domain** propagates assigned expressions in a symbolic way [6].
- **Boolean partitioning** relates the values of (integer) variables to the values of boolean variables.
- **Trace partitioning** tracks history of control flow branches and values along the execution trace.

References I

[1] P. Cousot, R. Cousot, and L. Mauborgne, "The reduced product of abstract domains and the combination of decision procedures,", Mar. 2011, pp. 456–472, ISBN: 978-3-642-19804-5. DOI: 10.1007/978-3-642-19805-2_31.

[2] P. Cousot and R. Cousot, "Systematic design of program analysis frameworks," in *Proceedings of the 6th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, ser. POPL '79, San Antonio, Texas: Association for Computing Machinery, 1979, pp. 269–282, ISBN: 9781450373579. DOI: 10.1145/567752.567778. [Online]. Available: https://doi.org/10.1145/567752.567778.

References II

[3] M. Codish, A. Mulkers, M. Bruynooghe, M. G. de la Banda, and M. Hermenegildo, "Improving abstract interpretations by combining domains," in *Proceedings of the 1993 ACM SIGPLAN Symposium on Partial Evaluation and Semantics-Based Program Manipulation*, ser. PEPM '93, Copenhagen, Denmark: Association for Computing Machinery, 1993, pp. 194–205, ISBN: 0897915941. DOI: 10.1145/154630.154650. [Online]. Available: https://doi.org/10.1145/154630.154650.

[4] J. Bertrane, P. Cousot, R. Cousot, *et al.*, "Static analysis and verification of aerospace software by abstract interpretation," *American Institue of Aeronautics and Astronautics (AIAA) Infotech@Aerospace 2010*, vol. 2, Apr. 2010. DOI: 10.2514/6.2010-3385.

References III

[5] P. Cousot, R. Cousot, J. Feret, *et al.*, "Combination of abstractions in the astrée static analyzer," in *Advances in Computer Science - ASIAN 2006. Secure Software and Related Issues*, M. Okada and I. Satoh, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 272–300, ISBN: 978-3-540-77505-8.

[6] A. Miné, "Symbolic methods to enhance the precision of numerical abstract domains," in *Verification, Model Checking, and Abstract Interpretation*, E. A. Emerson and K. S. Namjoshi, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 348–363, ISBN: 978-3-540-31622-0.