Combining Abstract Interpreters

Mooly Sagiv
http://www.cs.tau.ac.il/~msagiv/courses/pa16.html
Tel Aviv University
640-6706

Motivation

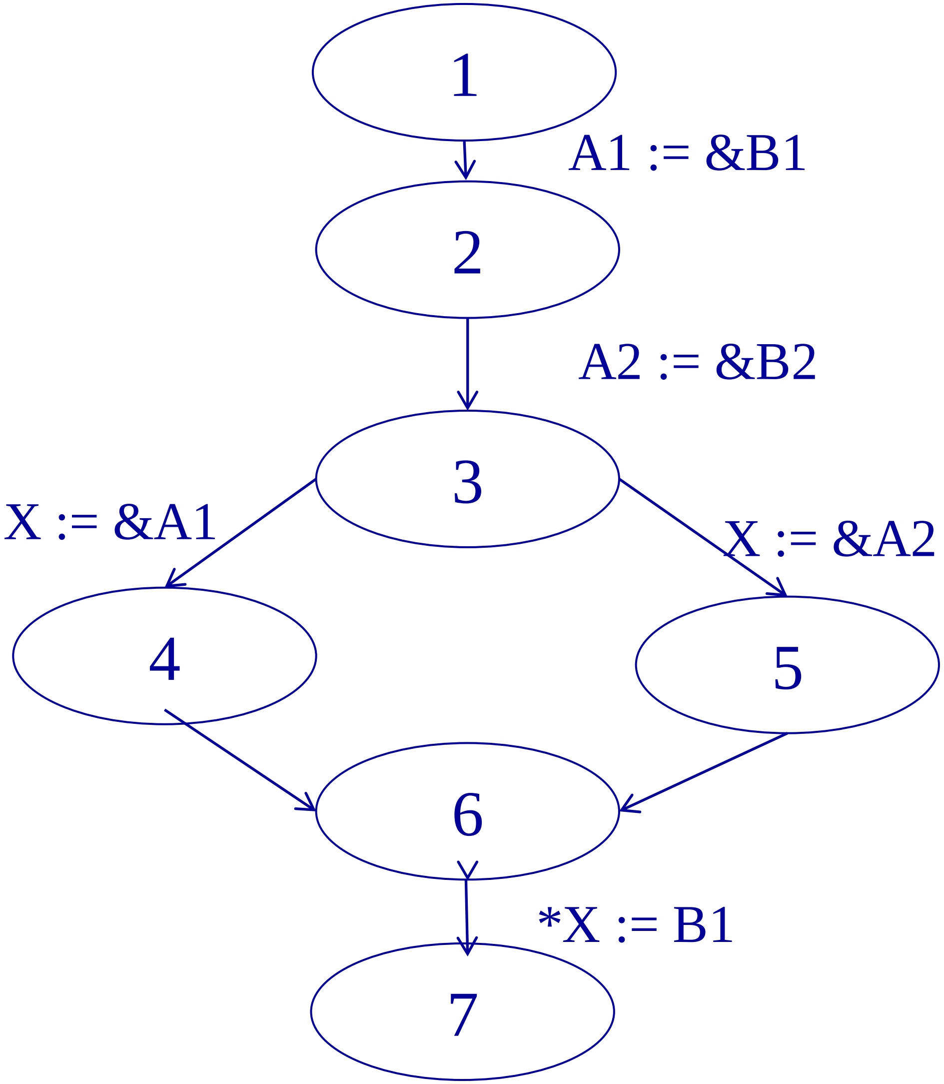
- Develop new abstract interpreters from old
- If I know how to handle programs over data type A and programs over data type B
  - How can I handle programs with variables of type A or type B
- Develop new abstract domains from old
- Develop new transformers from old

Pointer Semantics

Simple Pointer Commands

- <com> ::= X := Y   X, Y $\in$ Var
  - | X := &Y
  - | *X := Y
  - | assume X = Y
  - | assume *X = Y
  - | assert X = Y
  - | assert *X = Y
- Control Flow Graph G(N, E, s) where $E \subseteq N \times N$ is annotated with commands
  - s $\in$ N is the start node

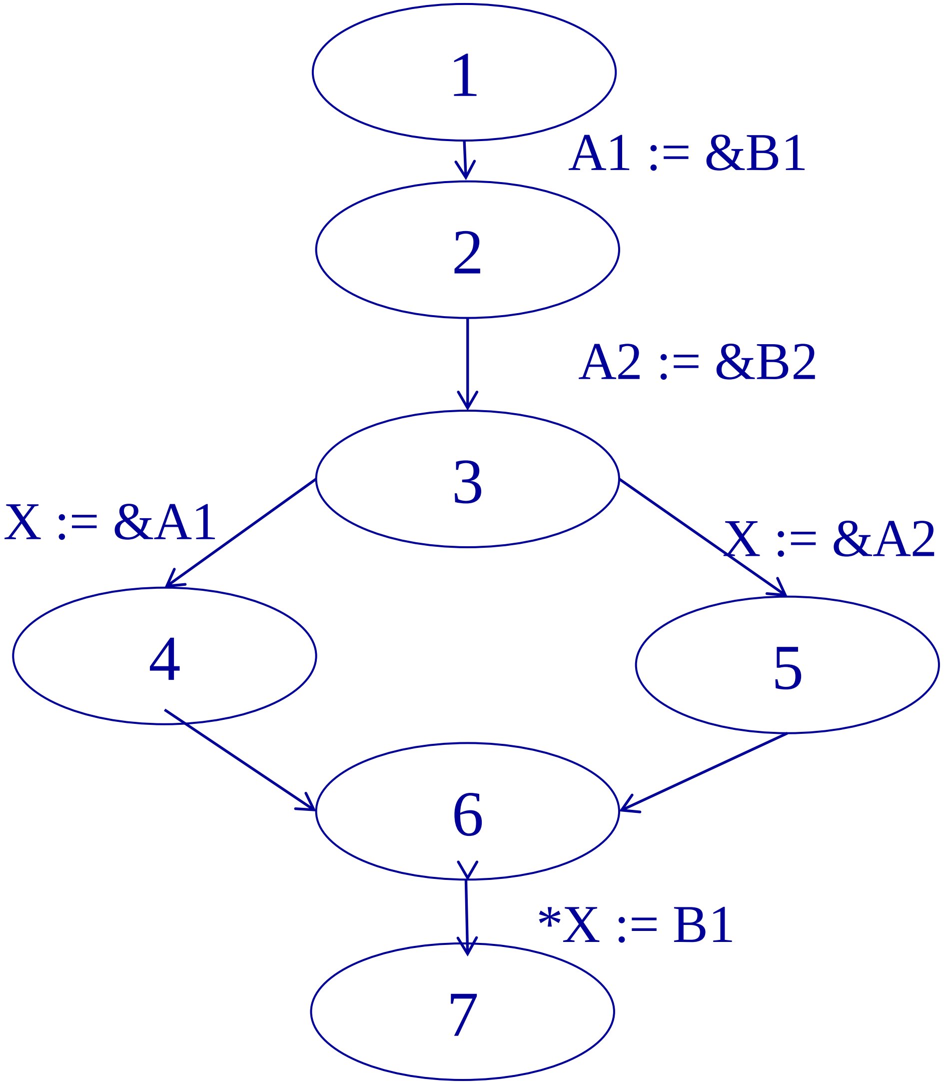
Example Program



Disjunctive Completion

- Given an abstract domain D
  - Construct an abstract domain D' such that join does not lose information
    - » How can this be formulated?
- Examples:
  - Signs = {∅, {+}, {-}. {0}, {0, +}, {0, -}, Z}
  - Points-to
- Size of the new domin
- Height of the new abstract domain
  - Finite lattices
  - Infinite lattice
- Constructing transformers

Example Program



Cartesian Product

- Given domains
  $D_1 = <D_1, \sqsubseteq^1, \sqcup^1, \sqcap^1, \perp^1, \top^1)$ and
  $D_2 = <D_2, \sqsubseteq^2, \sqcup^2, \sqcap^2, \perp^2, \top^2)$
- Construct a domain $D = <D_1 \times D_2, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$
- Galois connection
  – Is it a Galois insertion
- Widening
- Transformers

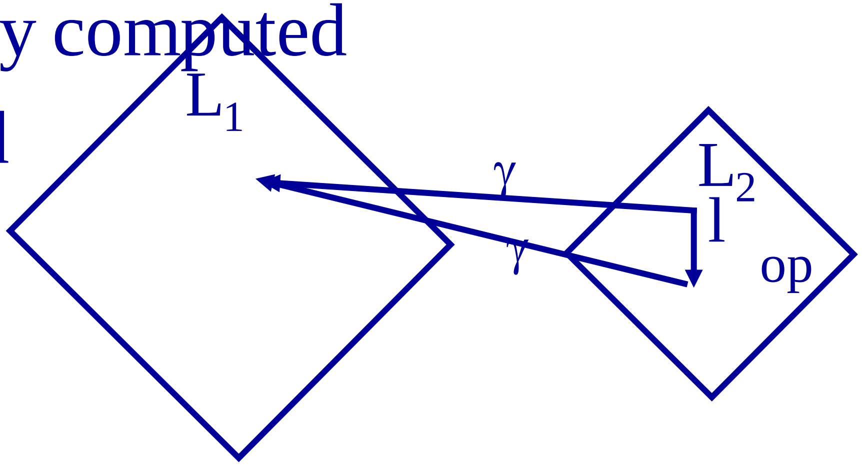
```
for (i=0; i < arr.length ; i++)
    arr[i] = 0
```

Reduced Product

- Cartesian product does not utilize the interaction between the domains
- How can each analysis in the abstract composition benefits from the information brought by the other analyses
- Two solutions
  - Employ a theorem prover
  - Employ semantic reduction

Semantic Reduction

- Improve the precision of the analysis by recovering properties of the program semantics
- A Galois connection $(L_1, \alpha, \gamma, L_2)$
- An operation $op: L_2 \rightarrow L_2$ is a semantic reduction
  - $\forall l \in L_2 \; op(l) \sqsubseteq l$
  - $\gamma(op(l)) = \gamma(l)$
- The most precise semantic reduction can be defined but not-necessarily computed
- Can be applied before and after basic operations



Example

- D1= Intervals
- D2 = Parity
- Example Reduction:
  - Update lower/upper bound

Granger Product

- A general heuristics for approximating semantic reduction
- $D_1 = <D_1, \sqsubseteq^1, \sqcup^1, \sqcap^1, \perp^1, \top^1)$
  $D_2 = <D_2, \sqsubseteq^2, \sqcup^2, \sqcap^2, \perp^2, \top^2)$
  $D = < D_1 \times D_2, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$
- Define operations: $\rho_1: D_1 \times D_2 \rightarrow D_1$ and $\rho_2: D_1 \times D_2 \rightarrow D_2$ such that
  - $\rho_1 (d_1, d_2) \sqsubseteq^1 d_1$ and $\gamma(\rho_1 (d_1, d_2), d_2) = \gamma(d_1, d_2)$
  - $\rho_2 (d_1, d_2) \sqsubseteq^2 d_2$ and $\gamma(d_1, \rho_2 (d_1, d_2)) = \gamma(d_1, d_2)$
- Compute the semantic reduction iteratively
  - $<a0, b0> = <a, b>$
  - $<a_{n+1}, b_{n+1}> = <\rho_1 (a_n, b_n), \rho_2(a_n, b_n) >$

A Simple Example [Intervals+Inqualities]

```
if (I <= 0)
   arr = new Int[1]
else
   arr = new Int[I]
for (i=0; i < I ; i++)
    arr[i] = 0;
```

A Complex Example [Intervals+Inqualities]

```
if (I <= 2)
   arr = new Int[1]
else
   arr = new Int[I]
for (i=3; i < I ; i++)
    arr[i] = 0;
```

Reduced Cardinal Power

- Combine the two domains in a way which keeps correlations between the individual elements
- The element a $\rightarrow$b means that if the state obeys ‘a’ it also obeys ‘b’

Reduced Cardinal Power

- Given domains
  $D_1 = \langle D_1, \sqsubseteq^1, \sqcup^1, \sqcap^1, \perp^1, \top^1)$ and
  $D_2 = \langle D_2, \sqsubseteq^2, \sqcup^2, \sqcap^2, \perp^2, \top^2)$
- Construct a domain $D = \langle D_1 \rightarrow D_2, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$
- Galois connection
- Transformers

A Complex Example [Intervals+Inqualities]

```
if (I <= 2)
   arr = new Int[1]
else
   arr = new Int[I]
for (i=3; i < I ; i++)
    arr[i] = 0;
```

A Complex Example [Booleans+Signs]

```
x := 100;  b := true;
while b do {
      x := x – 1;
      b := (x > 0);
}
```

Practical Applications of Reduced Cardinal Power

- Astree Branch Correlations
- Interprodecural analysis [Next lesson]
- Shape Analysis [Later]

Other Combinations

- Open Product
- Logical Product

Bibliography

- Cousot & Cousot POPL 1979
- Bruno Blachet, Introduction to Abstract Interpretation
- A. Cortesi, G. Contanini, P. Ferrrara: A survey o Product Operator in Abstract Interpretations
- Roberto Giacobazzi, Francesco Ranzato: Optimal Domains for Disjunctive Abstract Intepretation. Sci. Comput. Program. 32(1-3): 177-210
- Sumit Gulewani, Ashish Tiwari: Combining abstract interpreters PLDI’06

Summary

- Many ways to combine abstractions
- Simplifies the design and implementation of static analyzers
- Improve our understanding of abstractions