
Abstract Interpretation: a methodology for the
rapid development of provably correct static
analyzers

Arnaud Venet

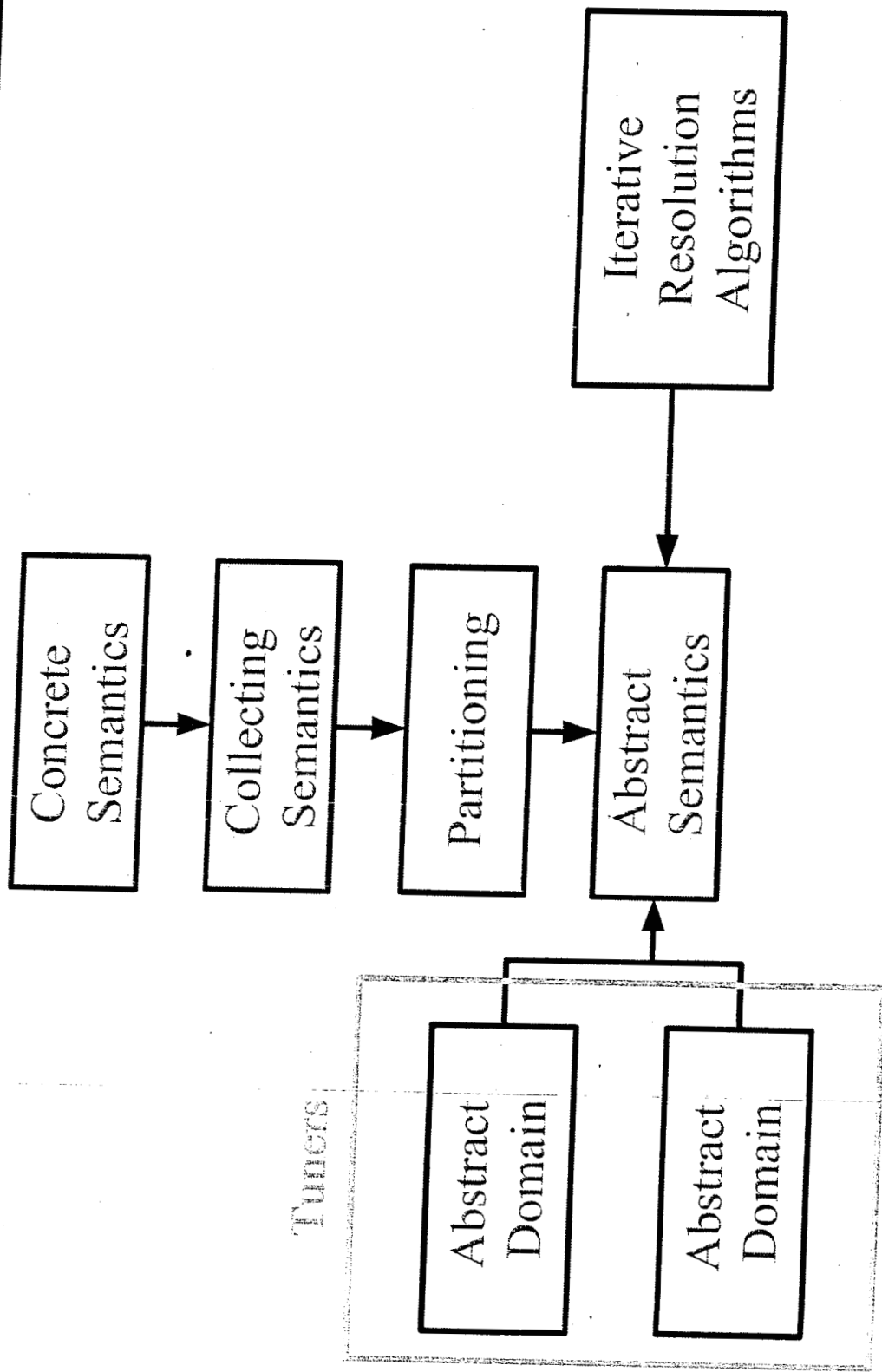
Kestrel Technology
NASA Ames Research Center

`arnaud@email.arc.nasa.gov`

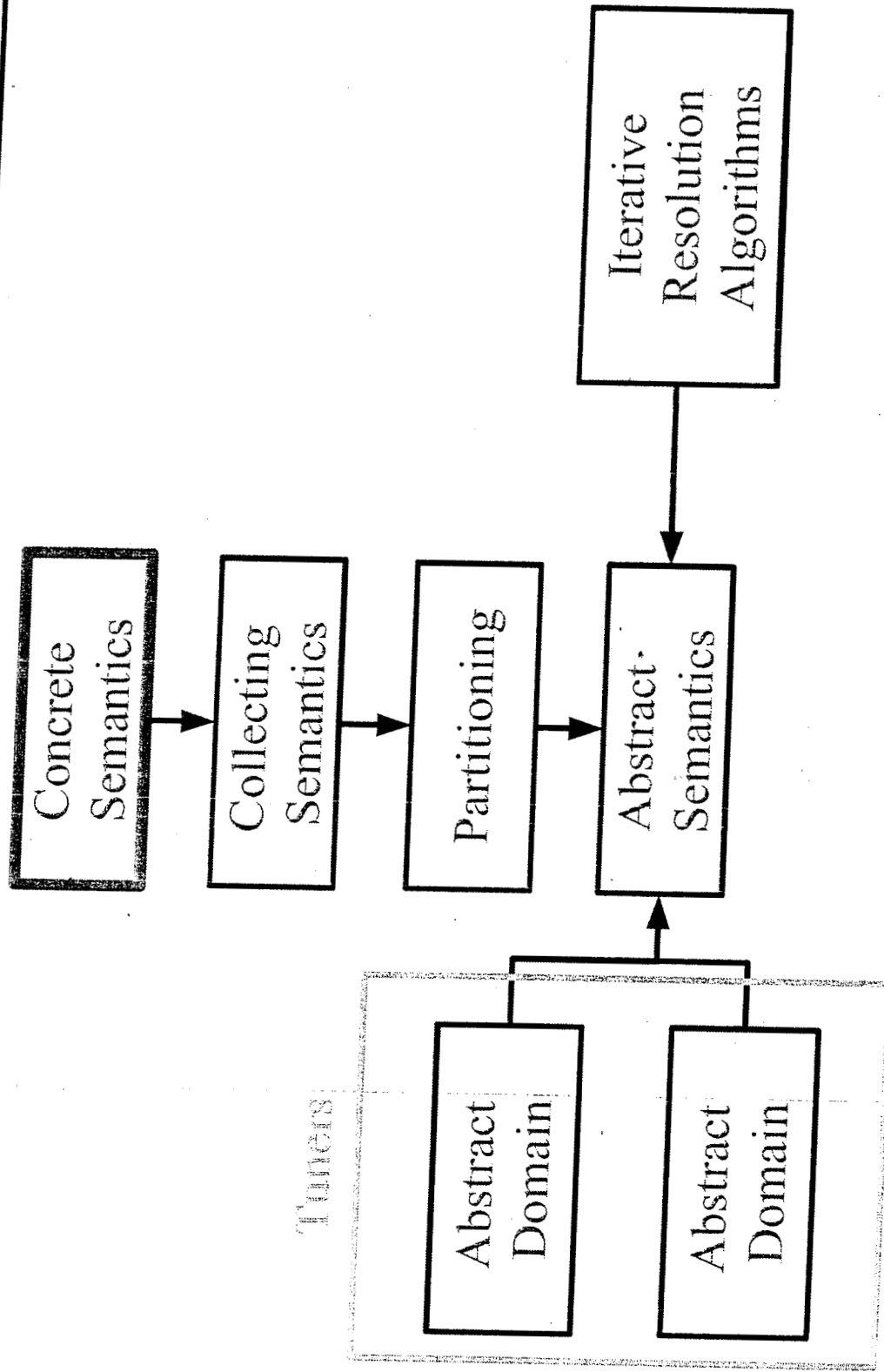
Abstract Interpretation

- + A general methodology for building static analyzers
- + Provides generic algorithms
- + Approximation and resolution are separated: the analyzers are tunable by construction
- + The soundness proof goes along with the analyzer design
- Scalability is difficult to achieve

Methodology



Methodology



Concrete semantics

Small-step operational semantics: $(\Sigma, \rightarrow)$

$S = \langle \text{program point}, \text{env} \rangle$

$S \rightarrow S'$

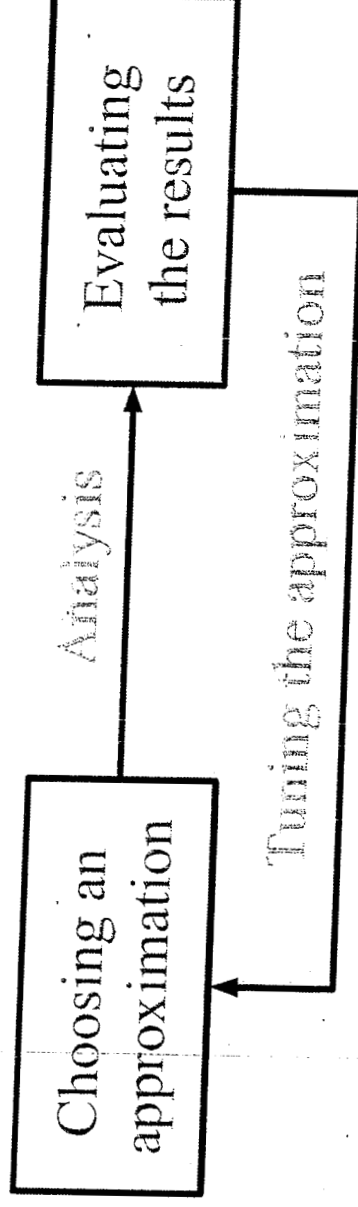
Example:

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

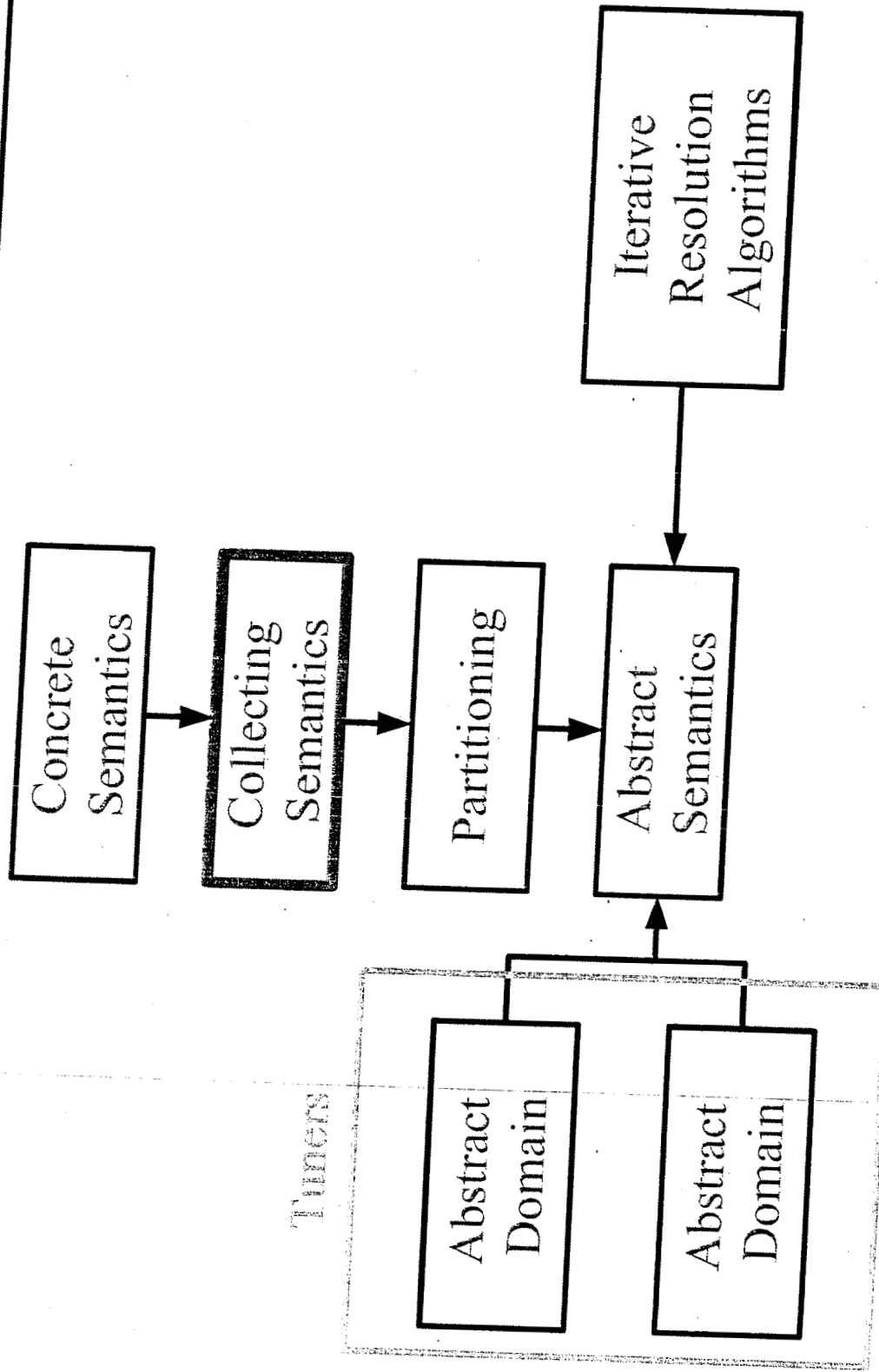
$$\begin{aligned} \langle 1, n \Rightarrow \Omega \rangle &\rightarrow \langle 2, n \Rightarrow 0 \rangle \rightarrow \langle 3, n \Rightarrow 0 \rangle \rightarrow \langle 4, n \Rightarrow 1 \rangle \\ &\rightarrow \langle 2, n \Rightarrow 1 \rangle \rightarrow \dots \rightarrow \langle 5, n \Rightarrow 1000 \rangle \end{aligned}$$

Static analysis in real life

- Undecidable problem: automatic program verification $\Rightarrow$ **loops**
- Approximation for decidability: false positives
- Tradeoff precision/efficiency
- The approximation should be tunable:



Methodology



Collecting semantics

The first abstraction step. It defines the observable behaviors of programs:

- Sets of states (e.g. range of variables)
- Sets of finite traces (e.g. computational dependencies)
- Sets of finite and infinite traces (e.g. termination properties)

State properties

The set of descendants of the initial state s_0 :

$$S = \{s \mid s_0 \rightarrow \dots \rightarrow s\}$$

Theorem: $F : (\wp(\Sigma), \subseteq) \rightarrow (\wp(\Sigma), \subseteq)$

$$F(S) = \{s_0\} \cup \{s' \mid \exists s \in S: s \rightarrow s'\}$$

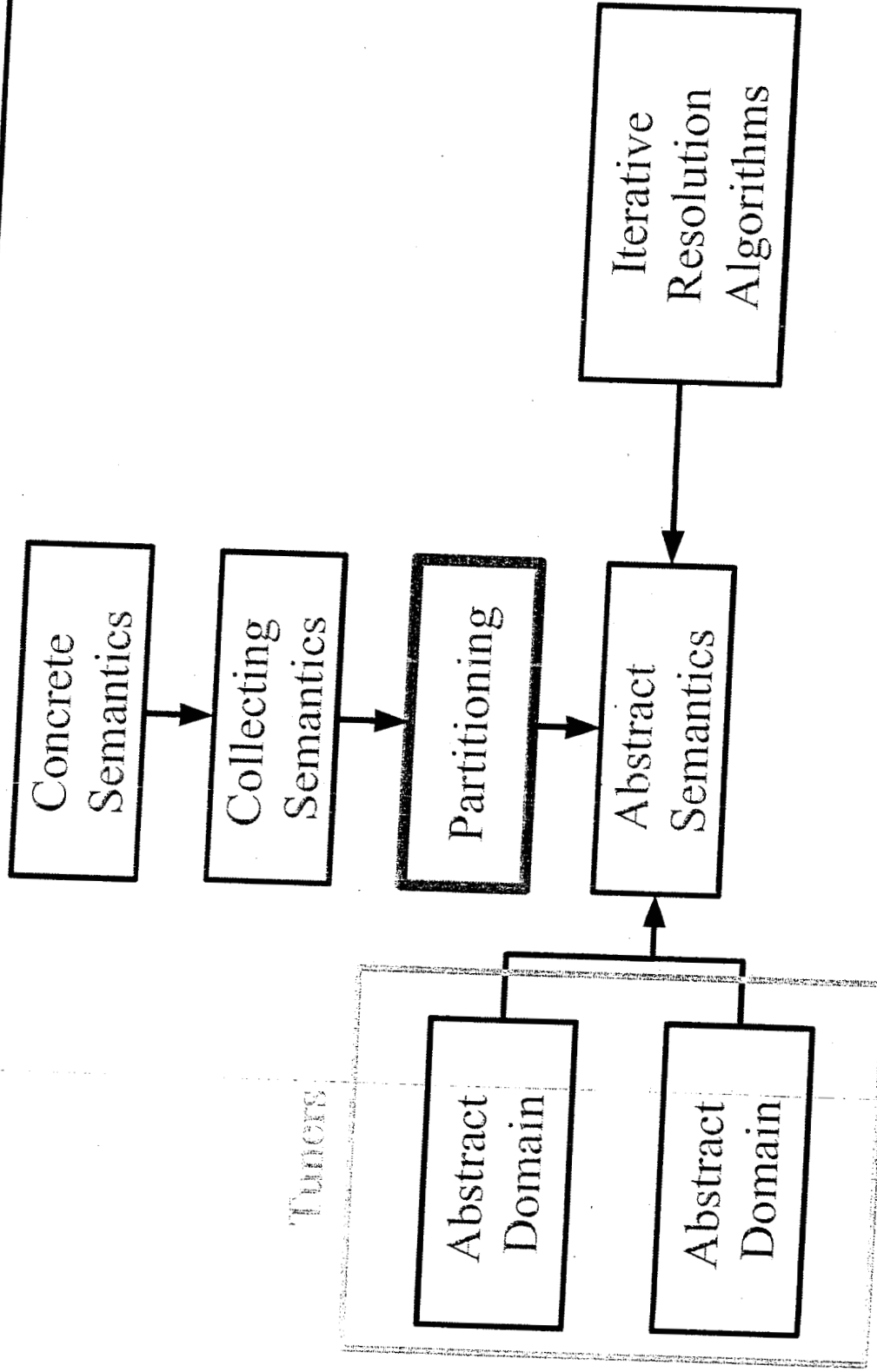
$$S = \text{lfp } F$$

Example

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

$$S = \{ \langle 1, n \Rightarrow \Omega \rangle, \langle 2, n \Rightarrow 0 \rangle, \langle 3, n \Rightarrow 0 \rangle, \langle 4, n \Rightarrow 1 \rangle, \langle 2, n \Rightarrow 1 \rangle, \dots, \langle 5, n \Rightarrow 1000 \rangle \}$$

Methodology



Partitioning

We partition the set Σ of states w.r.t. program points:

- $\Sigma = \Sigma_1 \oplus \Sigma_2 \oplus \dots \oplus \Sigma_n$
- $F(S_1, \dots, S_n)_i = \{s' \in S_i \mid \exists j \exists s \in S_j : s \rightarrow s'\}$
- Control-flow graph: $(P, \rightarrow)$
- $F(S_1, \dots, S_n)_i = \{\langle i, \epsilon' \rangle \mid \exists j \rightarrow i : \langle j, \epsilon \rangle \rightarrow \langle i, \epsilon' \rangle\}$

Semantic equations

- $i \rightarrow j$: operation **op**
- Notation: E_i = set of environments at program point i
- $[\mathbf{op}]_e$ = semantics of **op**
- System of semantic equations:

$$E_i = U \{ [\mathbf{op}]E_j \mid j \rightarrow i : \mathbf{op} \}$$

- Solution of the system = $\mathbf{S} = \text{lfp } F$

Example

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

$$E_1 = \{n \Rightarrow \Omega\}$$

$$E_2 = [n = 0]E_1 \cup E_4$$

$$E_3 = E_2 \cap]-\infty, 999]$$

$$E_4 = [n = n + 1]E_3$$

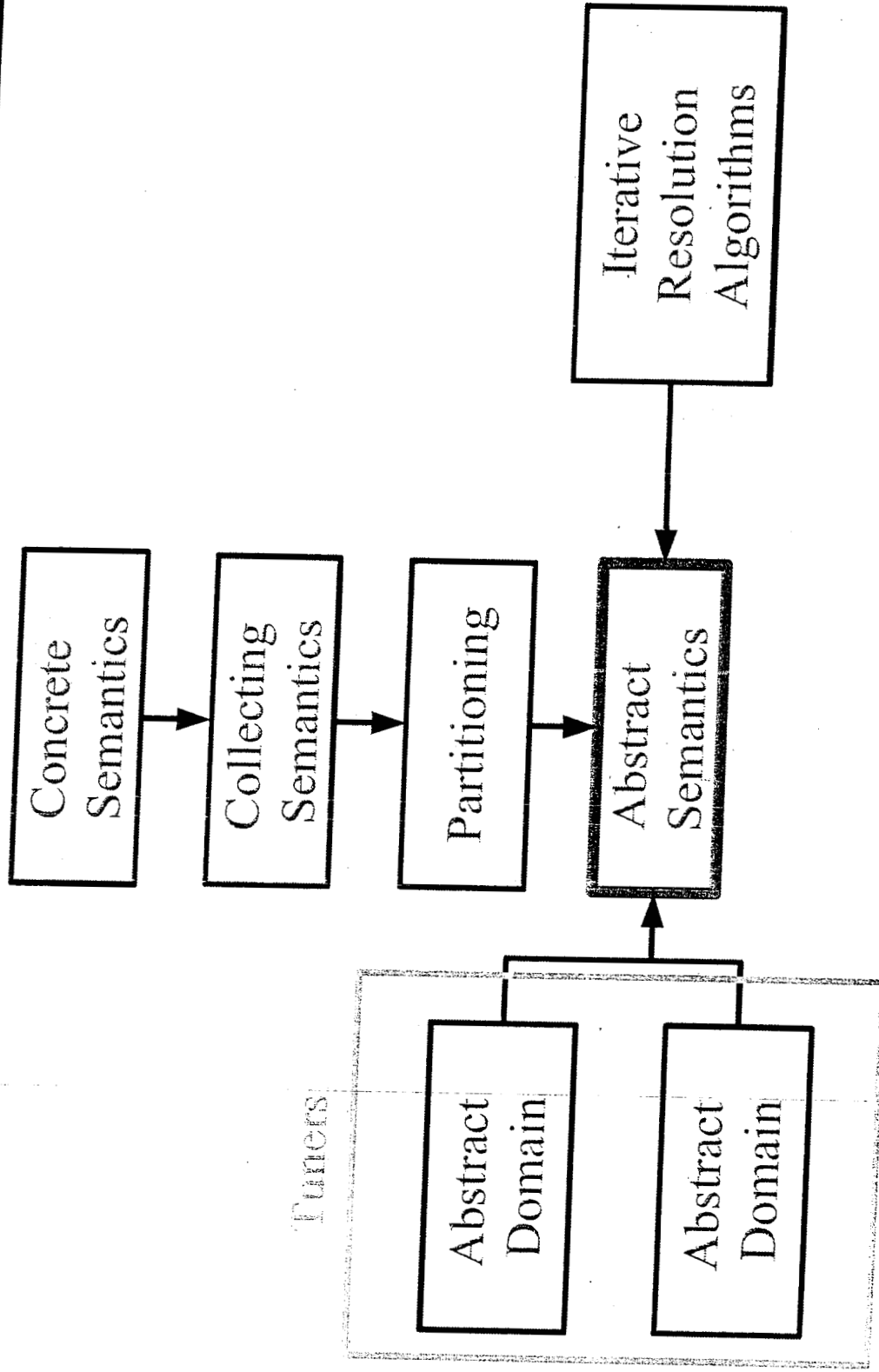
$$E_5 = E_2 \cap [1000, +\infty[$$

Other kinds of partitioning

In the case of collecting semantics of traces:

- Partitioning w.r.t. procedure calls: **context sensitivity**
- Partitioning w.r.t. executions paths in a procedure: **path sensitivity**
- Dynamic partitioning (Bourdoncle)

Methodology



Approximation

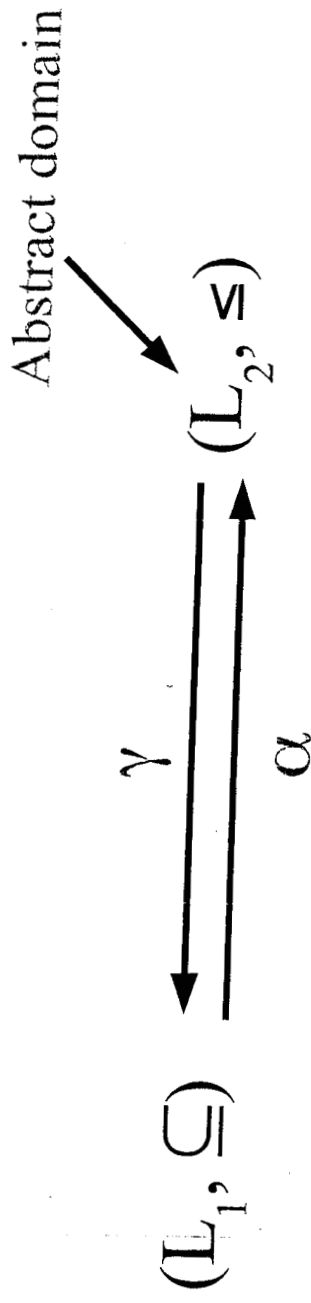
Problem: Compute a sound approximation $S^\#$ of S

$$S \subseteq S^\#$$

Solution: Galois connections

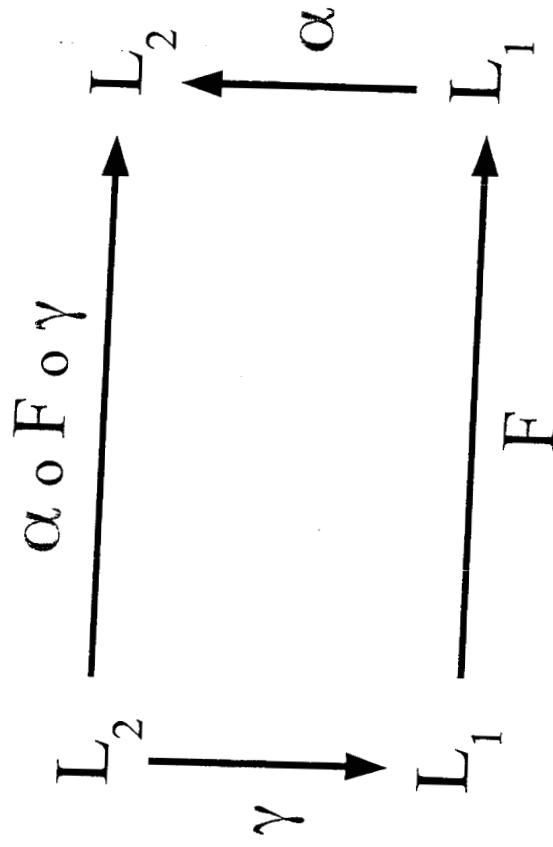
Galois connection

L_1, L_2 two lattices



- $\forall x \forall y : \alpha(x) \leq y \Leftrightarrow x \subseteq \gamma(y)$
- $\forall x \forall y : x \subseteq \gamma \circ \alpha(x) \ \& \ \alpha \circ \gamma(y) \leq y$

Fixpoint approximation



Theorem:

$$\text{lfp } F \subseteq \gamma (\text{lfp } \alpha \circ F \circ \gamma)$$

Abstracting the collecting semantics

- Find a Galois connection:

$$(\wp(\Sigma), \subseteq) \begin{array}{c} \xleftarrow{\gamma} \\ \xrightarrow{\alpha} \end{array} (\Sigma^{\#}, \leq)$$

- Find a function: $\alpha \circ F \circ \gamma \leq F^{\#}$

Partitioning $\Leftrightarrow$ Abstract sets of environments

Abstract algebra

- Notation: E the set of all environments
- Galois connection:

$$(\wp(E), \subseteq) \begin{array}{c} \xleftarrow{\gamma} \\ \xrightarrow{\alpha} \end{array} (E^{\#}, \leq)$$

- $\cup, \cap$ approximated by $\cup^{\#}, \cap^{\#}$
- $[\mathbf{op}]$ approximated by $[\mathbf{op}]^{\#}$

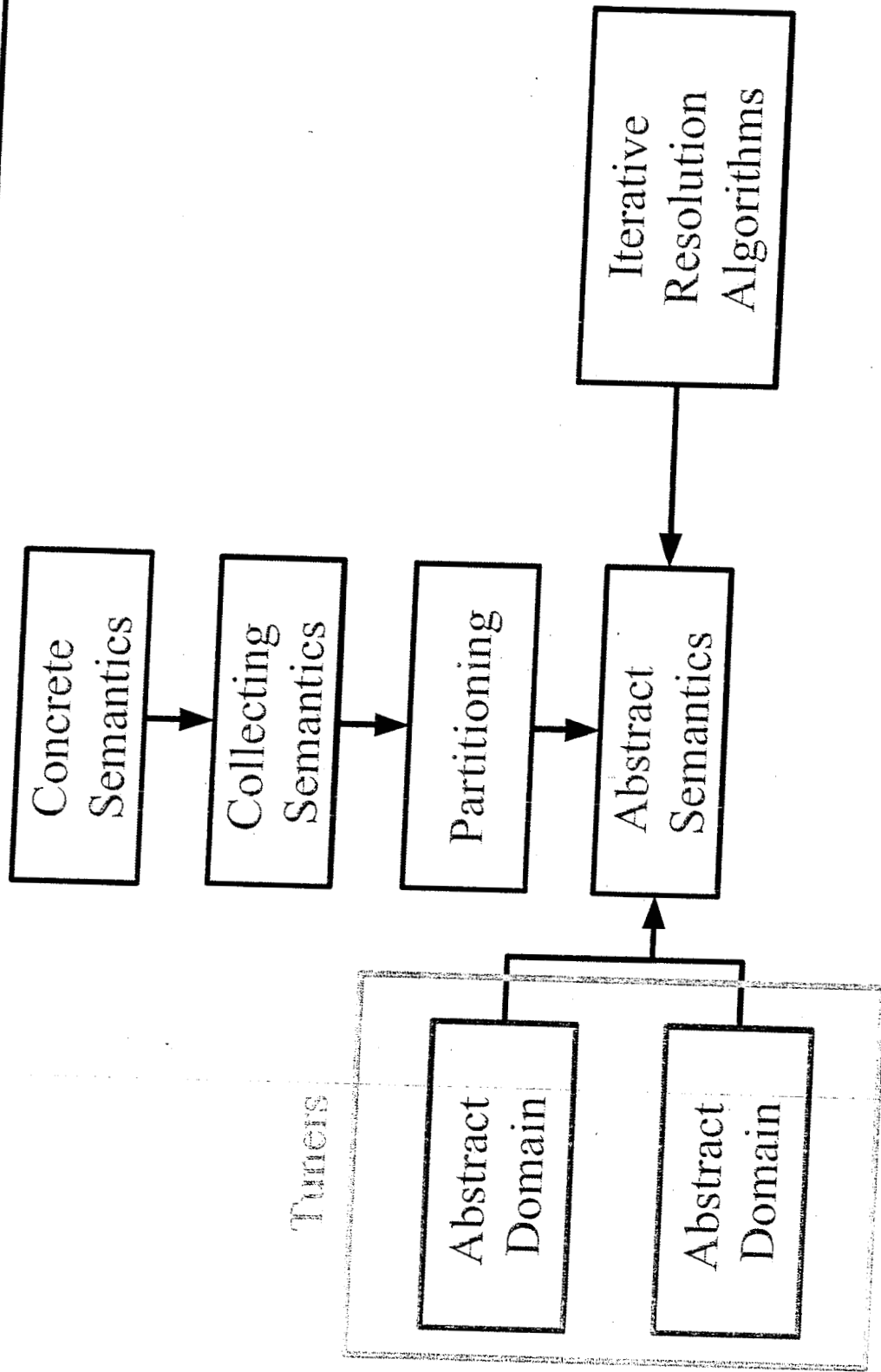
$$\alpha \circ [\mathbf{op}] \circ \gamma \leq [\mathbf{op}]^{\#}$$

Abstract semantic equations

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

$$\begin{aligned} E_1^\# &= \alpha(\{n \Rightarrow \Omega\}) \\ E_2^\# &= [n = 0]^\# E_1^\# \cup^\# E_4^\# \\ E_3^\# &= E_2^\# \cap^\# \alpha([-\infty, 999]) \\ E_4^\# &= [n = n + 1]^\# E_3^\# \\ E_5^\# &= E_2^\# \cap^\# \alpha([1000, +\infty[) \end{aligned}$$

Methodology



Abstract domains

Environment: $x \Rightarrow v, y \Rightarrow w, \dots$

Various kinds of approximations:

- Intervals (nonrelational):

$$x \Rightarrow [a, b], y \Rightarrow [a', b'], \dots$$

- Polyhedra (relational):

$$x + y - 2z \leq 10, \dots$$

- Difference-bound matrices (weakly relational):

$$y - x \leq 5, z - y \leq 10, \dots$$

Example: intervals

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

• Iteration 1: $E_2^\# = [0, 0]$

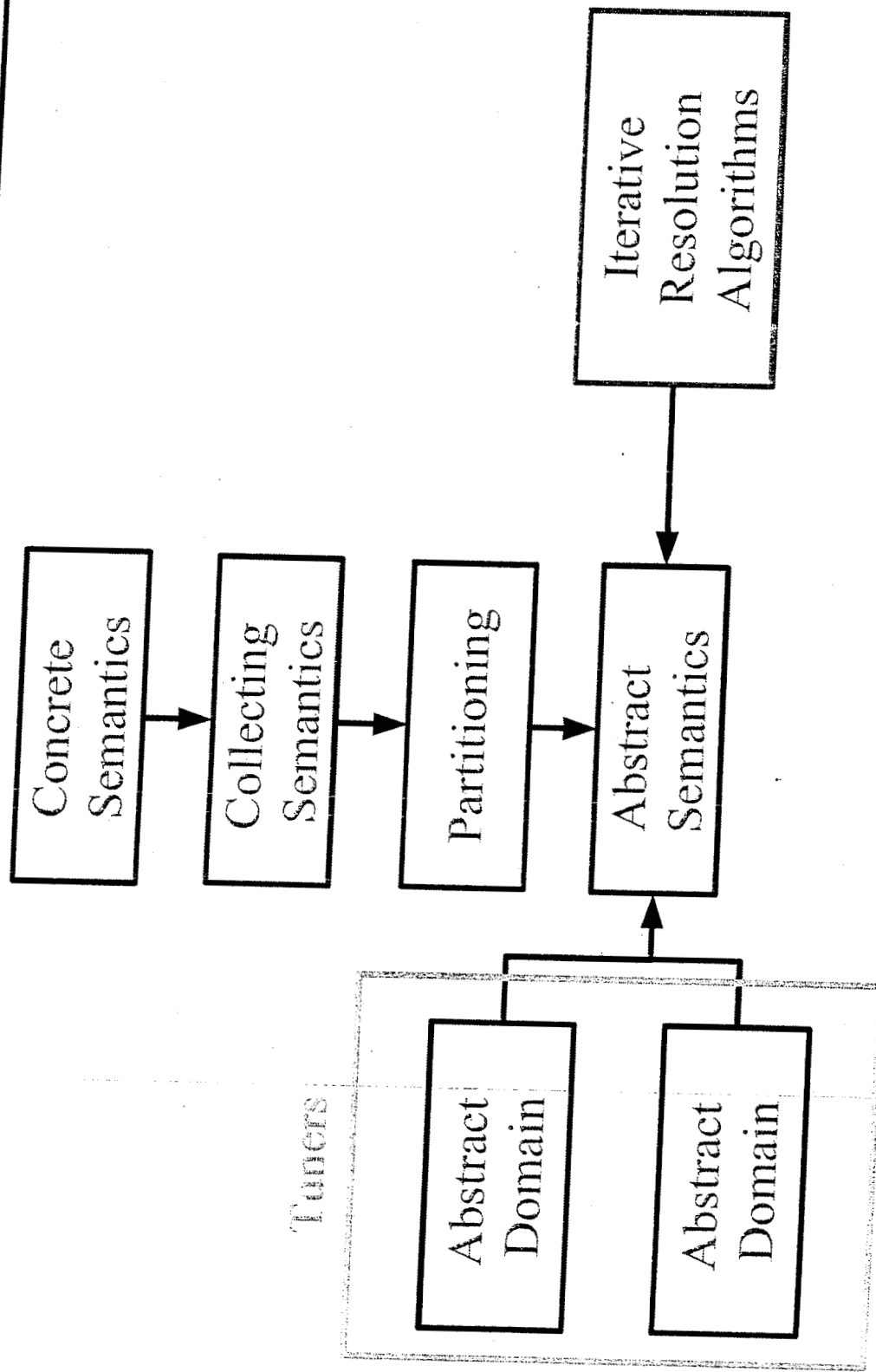
• Iteration 2: $E_2^\# = [0, 1]$

• Iteration 3: $E_2^\# = [0, 2]$

• Iteration 4: $E_2^\# = [0, 3]$

• ...

Methodology



Widening operator

Lattice $(L, \leq)$: $\nabla : L \times L \rightarrow L$

- Abstract union operator:

$$\forall x \forall y : x \leq x \nabla y \ \& \ y \leq x \nabla y$$

- Enforces convergence: $(x)_{n \geq 0}$

$$\begin{cases} y_0 &= x_0 \\ y_{n+1} &= y_n \nabla x_{n+1} \end{cases}$$

$(y)_{n \geq 0}$ is ultimately stationary

Widening of intervals

$$[a, b] \nabla [a', b']$$

- If $a \leq a'$ then a else $-\infty$
- If $b' \leq b$ then b else $+\infty$

↳ Open unstable bounds (jump over the fixpoint)

Iteration with widening

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit
```

$$(E_2^{\#})_{n+1} = (E_2^{\#})_n \nabla ([n = 0]^{\#}(E_1^{\#})_n \cup^{\#}(E_4^{\#})_n)$$

Iteration 1 (union): $E_2^{\#} = [0, 0]$

Iteration 2 (union): $E_2^{\#} = [0, 1]$

Iteration 3 (widening): $E_2^{\#} = [0, +\infty] \Rightarrow \text{stable}$

Imprecision at loop exit

```
1:  n = 0;  
2:  while n < 1000 do  
3:      n = n + 1;  
4:  end  
5:  exit t[n] = 0;
```

- $E_5^{\#} = [1000, +\infty[$
- The information is present in the equations

Narrowing operator

Lattice $(L, \leq)$: $\Delta : L \times L \rightarrow L$

- Abstract intersection operator:

$$\forall x \forall y : x \cap y \leq x \Delta y$$

- Enforces convergence: $(x)_{n \geq 0}$

$$\begin{cases} y_0 &= x_0 \\ y_{n+1} &= y_n \Delta x_{n+1} \end{cases}$$

$(y)_{n \geq 0}$ is ultimately stationary

Narrowing of intervals

$$[a, b] \Delta [a', b']$$

- If $a = -\infty$ then a' else a
- If $b = +\infty$ then b' else b

➡ Refine open bounds

Iteration with narrowing

```
1:  n = 0;  
2:  while n < 1000 do  
3:    n = n + 1;  
4:  end  
5:  exit t[n] = 0;
```

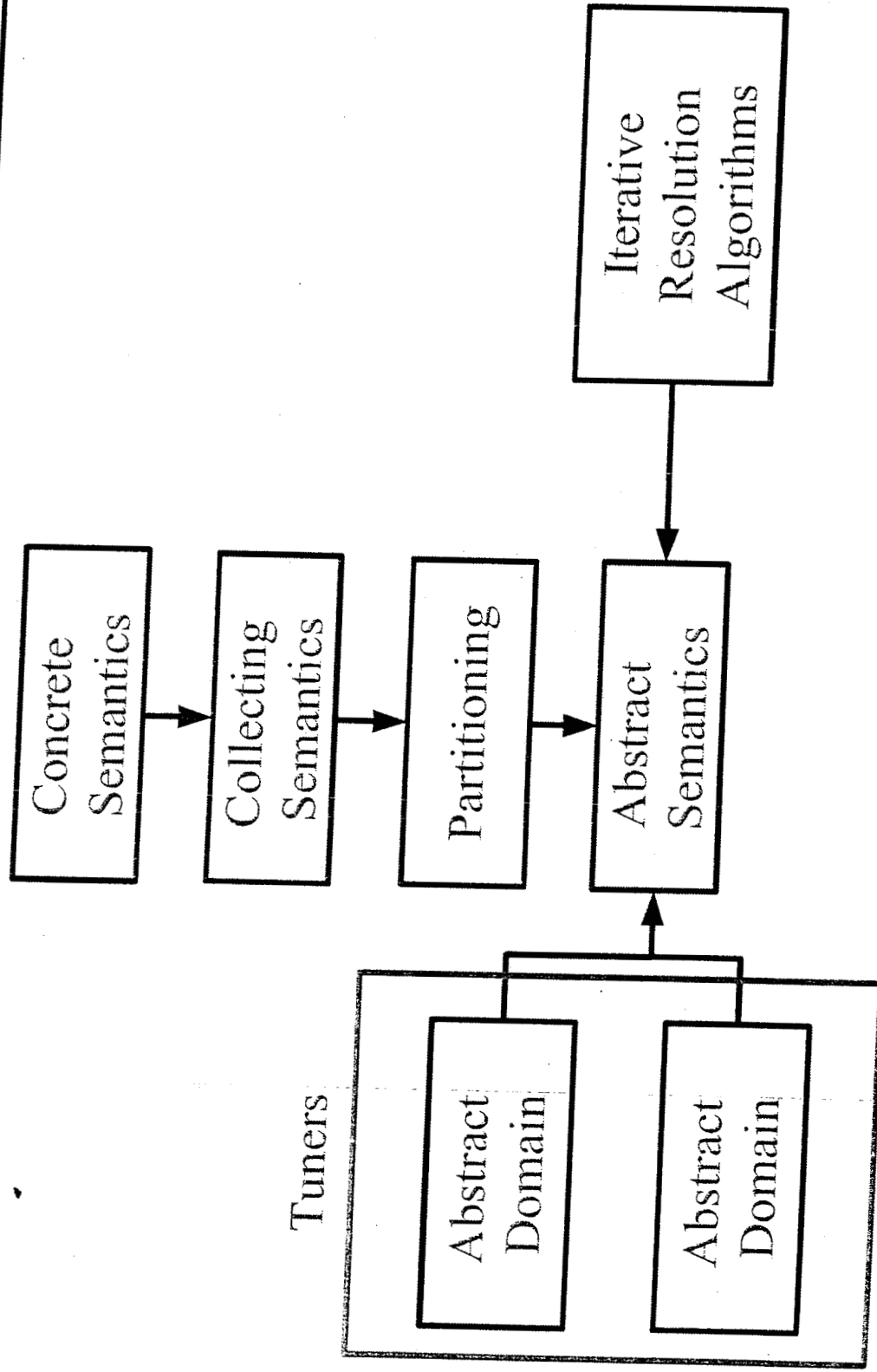
$$(E_2^{\#})_{n+1} = (E_2^{\#})_n \Delta ([n = 0]^{\#}(E_1^{\#})_n \cup^{\#}(E_4^{\#})_n)$$

Beginning of iteration: $E_2^{\#} = [0, +\infty[$

Iteration 1: $E_2^{\#} = [0, 1000] \Rightarrow$ stable

Consequence: $E_5^{\#} = [1000, 1000]$

Methodology



Tuning the abstract domains

```
1:  n = 0;  
2:  k = 0;  
3:  while n < 1000 do  
4:      n = n + 1;  
5:      k = k + 1;  
6:  end  
7:  exit
```

- Intervals:

$$E_4^\# = \langle n \Rightarrow [0, 1000], k \Rightarrow [0, +\infty[\rangle$$

- Convex polyhedra or DBMs:

$$E_4^\# = \langle 0 \leq n \leq 1000, 0 \leq k \leq 1000, n - k = 0 \rangle$$