Efficient Solving Techniques for Answer Set Programming

Carmine Dodaro
Department of Mathematics and Computer Science
University of Calabria

Klagenfurt am Wörthersee, 19th April 2016

Outline

Outline

Answer Set Programming (ASP)

- Declarative programming paradigm
- Based on the stable model (answer set) semantics

Idea:

1. Logic programs represent computational problems
2. Answer sets correspond to solutions
3. *Use a solver to find solutions*

Applications in several fields

Applications in several fields

Developing effective systems is a crucial research topic

ASP Computation

- Grounder
  - Eliminates variables
  - Produces an equivalent propositional theory
- Solver
  - Works on propositional theory

ASP syntax and semantics

Syntax:

$$\underbrace{a_1 \mid \ldots \mid a_n}_{\text{head}} \leftarrow \underbrace{b_1, \ldots, b_k, \sim b_{k+1}, \ldots, \sim b_m}_{\text{body}}$$

Intuitive meaning:

"The head must be true whenever the body is true."

ASP syntax and semantics

Syntax:

$$\underbrace{a_1 \mid \ldots \mid a_n}_{\text{head}} \leftarrow \underbrace{b_1, \ldots, b_k, \sim b_{k+1}, \ldots, \sim b_m}_{\text{body}}$$

Intuitive meaning:

"The head must be true whenever the body is true."

Definition (Stable models)

An interpretation I is a stable model (answer set) of a program Π if I is a minimal model of Π^I, i.e., Π where the interpretation of negative literals is fixed by I.

Example

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
$d \leftarrow \sim c$

Interpretation I_1

a, b, c

Interpretation I_2

a, b, d

Example

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

Interpretation I_2

a, b, d

Example

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
$d \leftarrow \sim c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim\!\!\cancel{d}$
~~$d \leftarrow \sim c$~~

✓ I_1 is stable

Interpretation I_2

a, b, d

Example

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
$d \leftarrow \sim c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
~~$d \leftarrow \sim c$~~

✓ I_1 is stable

Interpretation I_2

a, b, d

Reduct Π^{I_2}

~~$a \mid b \leftarrow c$~~
$a \leftarrow b$
$b \leftarrow a$
~~$c \leftarrow \sim d$~~
$d \leftarrow \sim c$

Example

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
$d \leftarrow \sim c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow \sim d$
~~$d \leftarrow \sim c$~~

✓ I_1 is stable

Interpretation I_2

a, b, d

Reduct Π^{I_2}

~~$a \mid b \leftarrow c$~~
$a \leftarrow b$
$b \leftarrow a$
~~$c \leftarrow \sim d$~~
$d \leftarrow \sim c$

✗ I_2 is not stable
✗ d is a model of Π^{I_2}

Properties of answer sets

- **Supportedness:** All atoms in answer set must be supported

Example

b $\leftarrow$
a $\leftarrow$ b, $\sim$ c

$I = \{a, b\}$ is an answer set and a and b are supported

- **Unfounded-free:** an answer set is unfounded-free

Example

b $\leftarrow$ a
a $\leftarrow$ b

$I = \{a, b\}$ is a supported model but it is not an answer set

ASP encoding of TSP

Example (Travelling Salesman Problem)

Input: A weighted, directed graph $G =< V, E, \phi >$
A vertex $s \in V$

Goal: Find the Hamiltonian cycle of minimum weight

% Guess a cycle		
$in(x,y) \mid out(x,y) \leftarrow$	$\forall (x,y) \in E$	Guess
% A vertex can be reached only once		
$\leftarrow \#count\{1 : in(x,y) \mid y \in V\} = 1$	$\forall x \in V$	Check
$\leftarrow \#count\{1 : in(x,y) \mid x \in V\} = 1$	$y \in V$	
% All vertices must be reached		
$\leftarrow not\ reached(x)$	$\forall x \in V$	Aux. Rules
$reached(y) \leftarrow in(s,y)$	$\forall y \in V$	
$reached(y) \leftarrow reached(x), in(x,y)$	$\forall (x,y) \in E$	
% Minimize the sum of distances		
$\leftarrow in(x,y) \quad [\phi(x,y)]$	$\forall (x,y) \in E$	Optimize

Outline

Solving ground ASP programs

Computational tasks and applications

1. Model generation
   - Given a ground ASP program Π, find an answer set of Π

     $\rightarrow$ [Balduccini et al., LPNMR 2001; Gebser et al, TPLP 2011]
2. Optimum answer set search
   - Given a ground ASP program Π, find an answer set of Π with the minimum cost

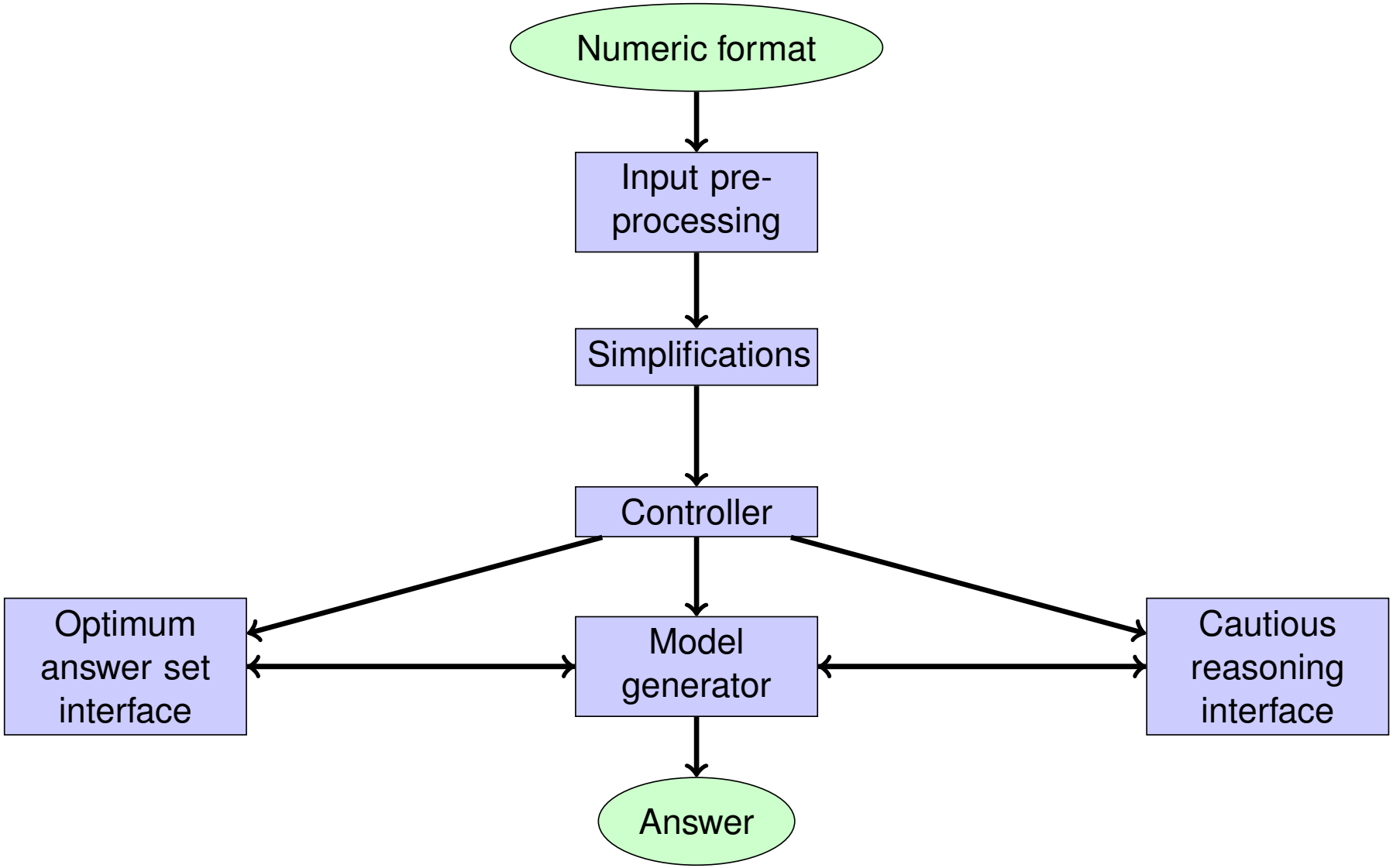
     $\rightarrow$ [Marra et al., JELIA 2014; Koponen et al., TPLP 2015]
3. Cautious reasoning
   - Given a ground ASP program Π and a ground atom a, check whether a is true in all answer sets of Π

     $\rightarrow$ [Arenas et al., TPLP 2003; Eiter, LPNMR 2005]

Computational tasks: complexity

Computational Task	**Normal Programs**	**Disjunctive Programs**
Model Generation	NP-complete	Σ_2^P-complete
Optimum Answer Set Search	Δ_2^P-complete	Δ_3^P-complete
Cautious Reasoning	coNP-complete	Π_2^P-complete

Architecture of an ASP solver



Input preprocessing and simplifications

Preprocessing of the input program

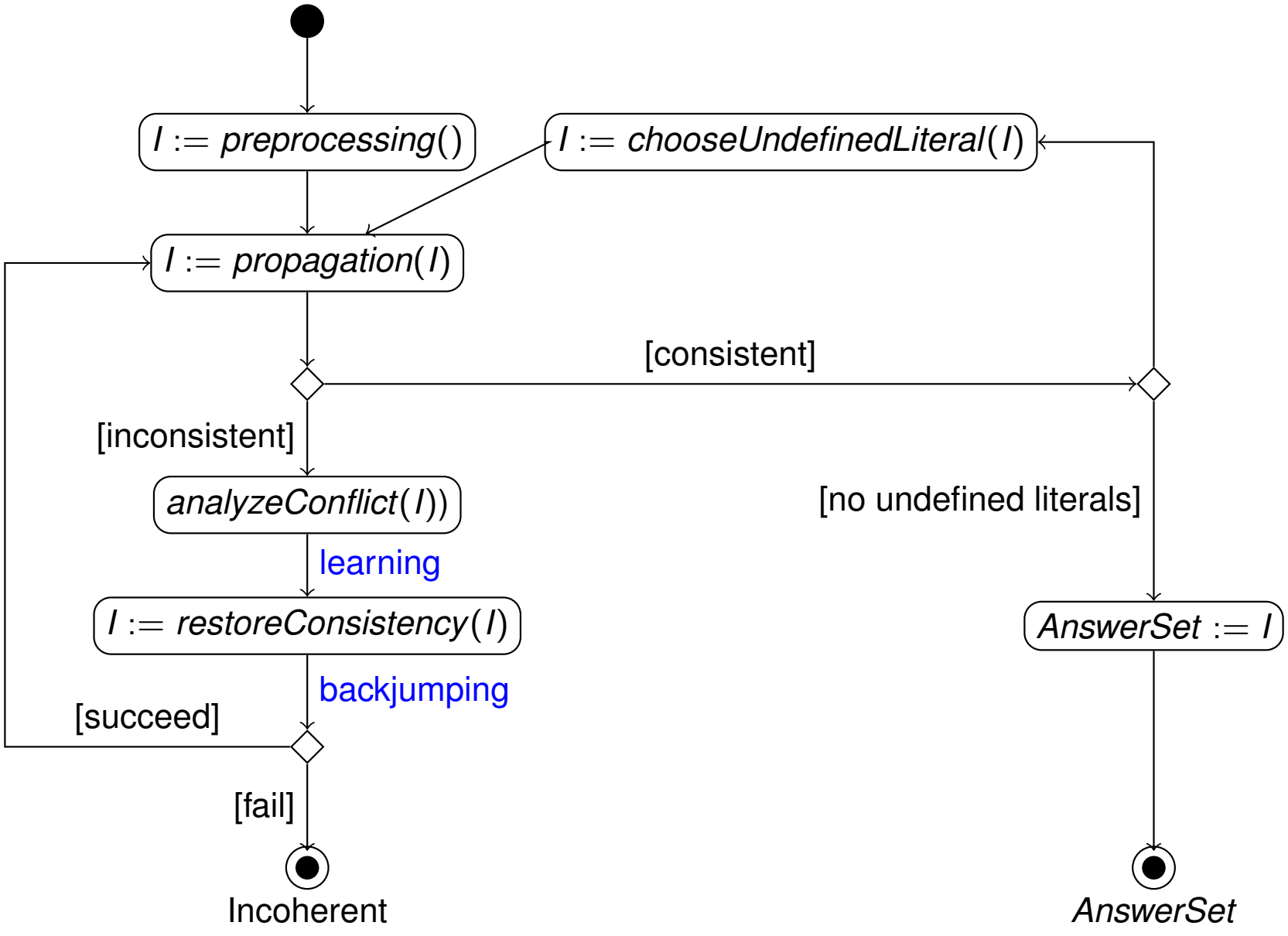
- Deletion of duplicate rules
  - -> Even more than 80% in some benchmarks
- Deterministic inferences
  - -> Deletion of satisfied rules
- Clark's completion
  - -> Constraints for discarding unsupported models

Simplifications

- In the style of SATELITE [Eén and Biere, SAT 2005]
  - -> Subsumption, self-subsumption, literals elimination

Outline

Model Generator



Propagation

Derivation Rules

1. Unit propagation (from SAT)
2. Aggregates propagation (from Pseudo-Boolean)
3. Unfounded-free propagation (ASP specific)

Unit and Aggregate propagation

- Infer a literal if it is the only one which can satisfy a rule

Example (Unit propagation)

a $\leftarrow$ b, c.

If *b* and *c* are true then *a* must be true

- Uses aggregates for further inferences

Example (Aggregate propagation)

$\leftarrow$ #sum{1 : d; 2 : e; 1 : f} >=2

If *d* is true then *e* and *f* must be false

Unfounded-free propagation

- All atoms in an unfounded set are inferred as false

Example (Unfounded set)

a $\leftarrow$ b
b $\leftarrow$ a

$\{a, b\}$ is an unfounded set, thus a and b are inferred as false

Unfounded-free check

- HCF programs
  - Unfounded-free check can be done in polynomial time
  - Algorithm based on source pointers [Simons et al., Artif. Intell. 2002]
- non-HCF programs
  - Unfounded-free check is co-NP complete
  - Algorithm based on calls to a SAT oracle [Koch et al., Artif. Intell. 2003]
    - Input: an input program Π and an interpretation I
    - Output: a formula φ such that I is unfounded-free if and only if φ is unsatisfiable

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Interpretation I_2

a, b, d

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

Interpretation I_2

a, b, d

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

$\varphi(\Pi, I_1)$

$a \vee b \vee \neg c$
$a \vee \neg b$
$b \vee \neg a$
c
$\neg a \vee \neg b \vee \neg c$

Interpretation I_2

a, b, d

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

$\varphi(\Pi, I_1)$

$a \vee b \vee \neg c$
$a \vee \neg b$
$b \vee \neg a$
c
$\neg a \vee \neg b \vee \neg c$

✓ Unsatisfiable

Interpretation I_2

a, b, d

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

$\varphi(\Pi, I_1)$

$a \vee b \vee \neg c$
$a \vee \neg b$
$b \vee \neg a$
c
$\neg a \vee \neg b \vee \neg c$

✓ Unsatisfiable

Interpretation I_2

a, b, d

Reduct Π^{I_2}

~~$a \mid b \leftarrow c$~~
$a \leftarrow b$
$b \leftarrow a$
~~$c \leftarrow {\sim}d$~~
$d \leftarrow {\sim}c$

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

$\varphi(\Pi, I_1)$

$a \vee b \vee \neg c$
$a \vee \neg b$
$b \vee \neg a$
c
$\neg a \vee \neg b \vee \neg c$

✓ Unsatisfiable

Interpretation I_2

a, b, d

Reduct Π^{I_2}

~~$a \mid b \leftarrow c$~~
$a \leftarrow b$
$b \leftarrow a$
~~$c \leftarrow {\sim}d$~~
$d \leftarrow {\sim}c$

$\varphi(\Pi, I_2)$

$a \vee \neg b$
$b \vee \neg a$
d
$\neg a \vee \neg b \vee \neg d$

Example

$$\varphi(\Pi, I) := \{(H(r) \cap I) \cup \{\neg b \mid b \in B^+(r)\} \mid r \in \Pi^I\} \cup \{\{\neg p \mid p \in I\}\}$$

Program Π

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
$d \leftarrow {\sim}c$

Interpretation I_1

a, b, c

Reduct Π^{I_1}

$a \mid b \leftarrow c$
$a \leftarrow b$
$b \leftarrow a$
$c \leftarrow {\sim}d$
~~$d \leftarrow {\sim}c$~~

$\varphi(\Pi, I_1)$

$a \vee b \vee \neg c$
$a \vee \neg b$
$b \vee \neg a$
c
$\neg a \vee \neg b \vee \neg c$

✓ Unsatisfiable

Interpretation I_2

a, b, d

Reduct Π^{I_2}

~~$a \mid b \leftarrow c$~~
$a \leftarrow b$
$b \leftarrow a$
~~$c \leftarrow {\sim}d$~~
$d \leftarrow {\sim}c$

$\varphi(\Pi, I_2)$

$a \vee \neg b$
$b \vee \neg a$
d
$\neg a \vee \neg b \vee \neg d$

✗ $\{d\}$ is a model

Heuristics and learning

Learning

- Detect the reason of a conflict
- Learn constraints using 1-UIP schema

Deletion Policy

- Exponentially many constraints $\rightarrow$ forget something
- Less “useful” constraints are removed

Search Restarts

- Avoid unfruitful branches by restarting the search
- Based on some heuristic sequence

Branching Heuristics

- Look back MINISAT heuristic

Optimum answer set search

- **Find the answer set with the minimum cost**
  - **Input:** a propositional program Π
  - **Output:** an optimum answer set of Π
- Based on MaxSAT algorithms
  - Model-guided
  - Core-guided

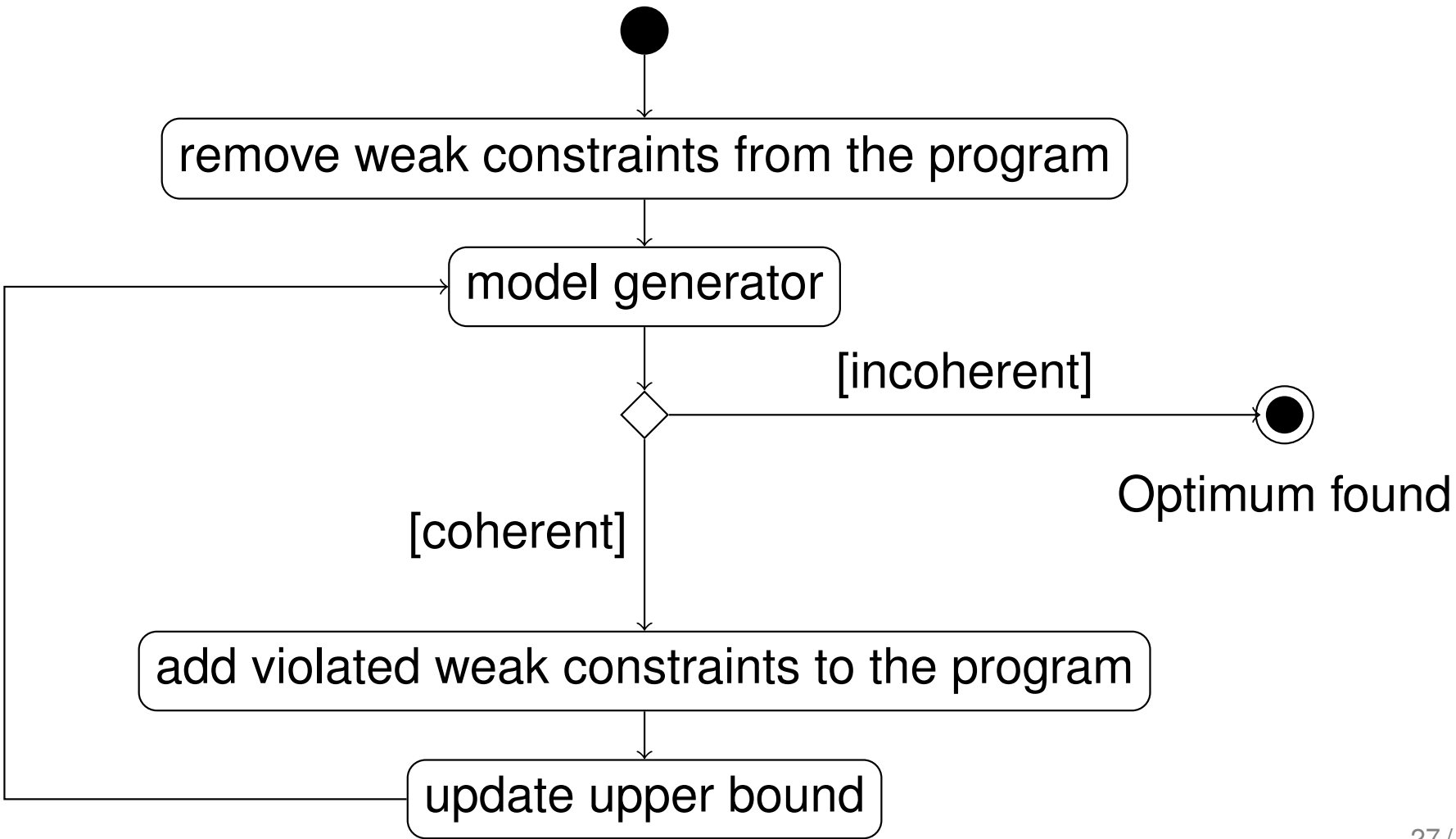
Outline

Optimum answer set search

- Model-guided algorithms: OPT, BASIC and MGD
  - \+ Easy to implement
  - \+ Work well on particular domains
  - \+ Produce feasible solutions during the search
  - \- Poor performances on industrial instances
- Core-guided algorithms: PMRES and OLL
  - \+ Good performances on industrial instances
  - \- Do not produce feasible solutions (in general)
  - \- The implementation is usually nontrivial

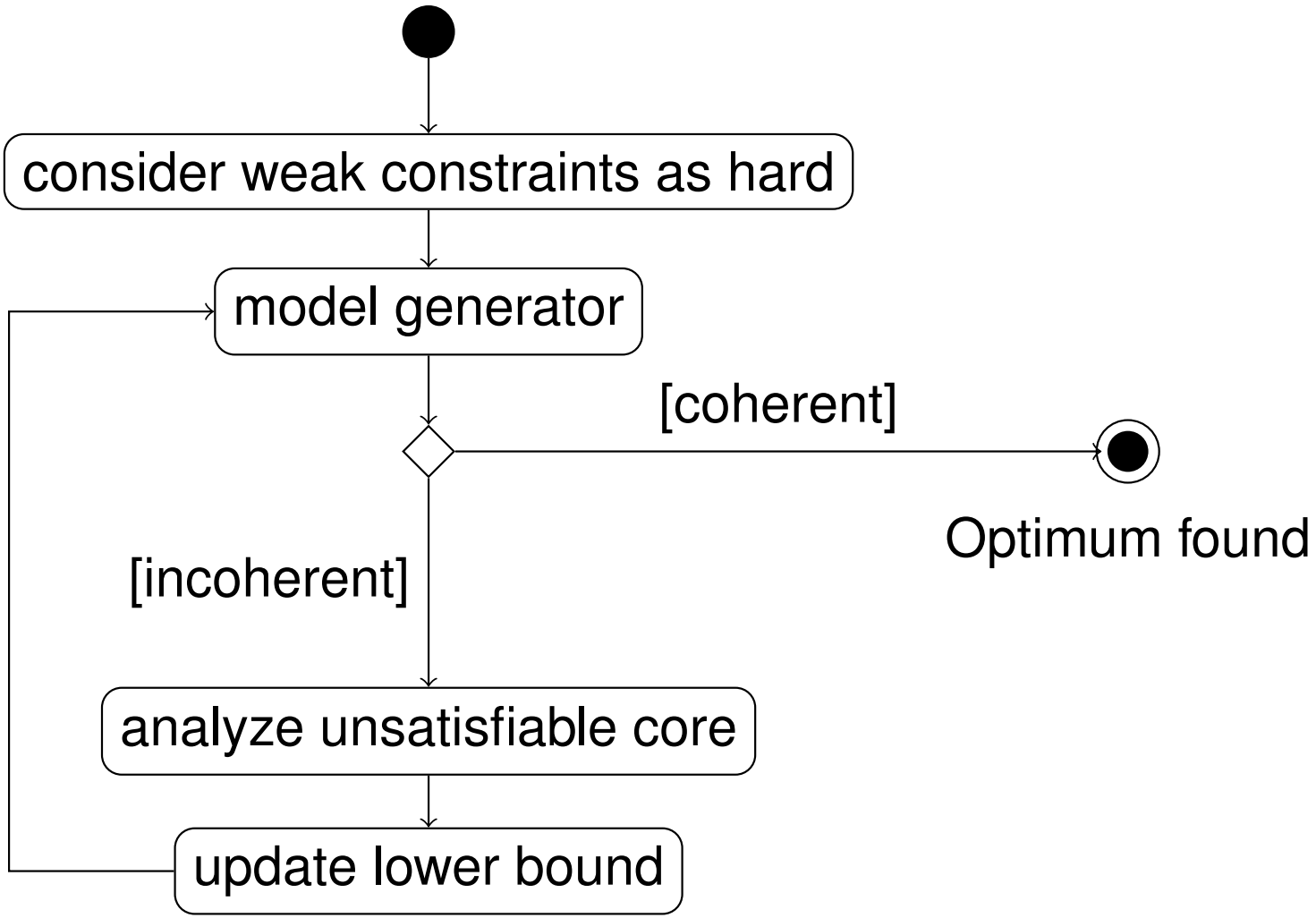
Model-guided algorithms

I need a solution! Give me any answer set



Core-guided algorithms

I feel lucky! Try to satisfy all weak constraints



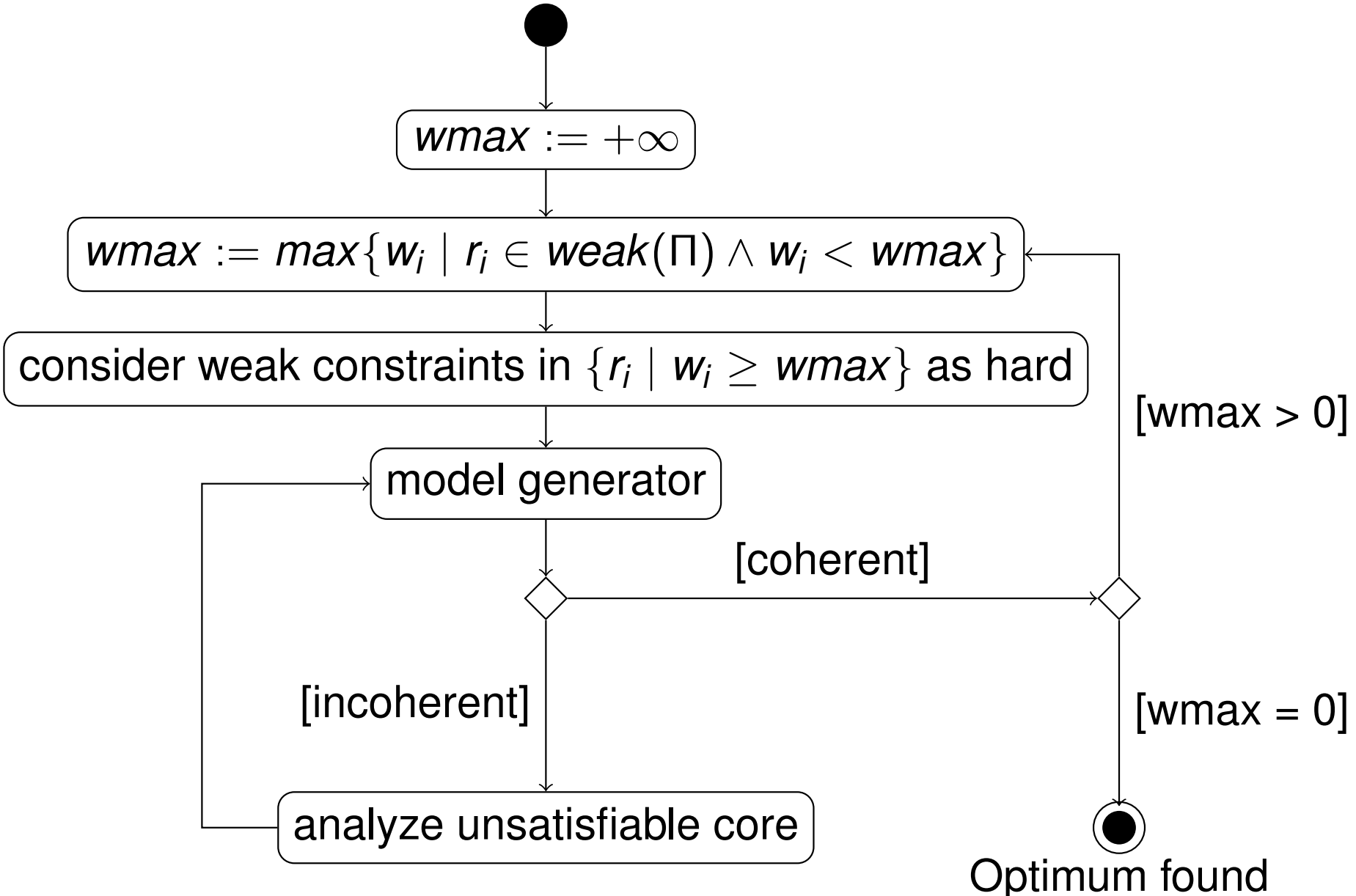
Stratification

1. Generic technique

   $\rightarrow$ can be applied to PMRES, OLL, ...

2. Force the ASP solver to concentrate on weak constraints with higher weights

Stratification



Outline

Cautious reasoning

- Formally, an atom a is a cautious consequence of a program Π if a belongs to all stable models of Π

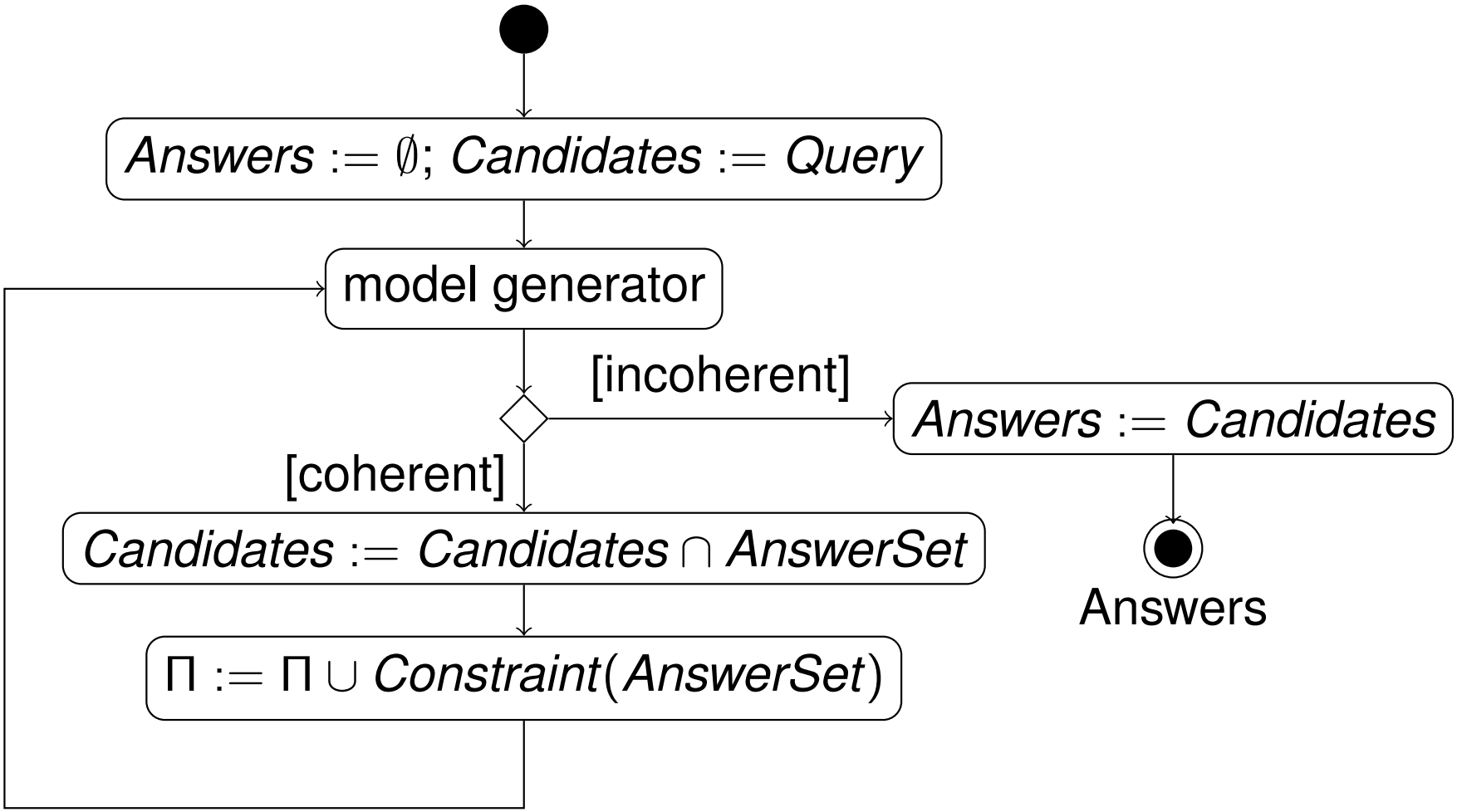
Compute cautious consequences

- **Input:** a propositional program Π
- **Output:** all cautious consequences of Π

Cautious reasoning: algorithms

- Enumeration of models (DLV)
- Overestimate reduction (CLASP)
- Iterative coherence testing (WASP)

Cautious reasoning by enumeration of models



Example

Program Π

$a \leftarrow not\ b \quad b \leftarrow not\ a \quad$ % either a or b
$c \leftarrow a \quad c \leftarrow b$
$d \leftarrow not\ e \quad e \leftarrow not\ d \quad$ % either d or e

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

$\leftarrow a, c, e$ % added after step 2

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{a, c, e\}$	$\emptyset$	$\{a, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

$\leftarrow a, c, e$ % added after step 2

$\leftarrow b, c, d$ % added after step 3

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{a, c, e\}$	$\emptyset$	$\{a, c\}$
3	$\{b, c, d\}$	$\emptyset$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

$\leftarrow a, c, e$ % added after step 2

$\leftarrow b, c, d$ % added after step 3

$\leftarrow b, c, e$ % added after step 4

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{a, c, e\}$	$\emptyset$	$\{a, c\}$
3	$\{b, c, d\}$	$\emptyset$	$\{c\}$
4	$\{b, c, e\}$	$\emptyset$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

$\leftarrow a, c, e$ % added after step 2

$\leftarrow b, c, d$ % added after step 3

$\leftarrow b, c, e$ % added after step 4

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{a, c, e\}$	$\emptyset$	$\{a, c\}$
3	$\{b, c, d\}$	$\emptyset$	$\{c\}$
4	$\{b, c, e\}$	$\emptyset$	$\{c\}$
5	*Incoherent*	$\{c\}$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ % either d or e

$\leftarrow a, c, d$ % added after step 1

$\leftarrow a, c, e$ % added after step 2

$\leftarrow b, c, d$ % added after step 3

$\leftarrow b, c, e$ % added after step 4

Query Q

a, b, c

Execution

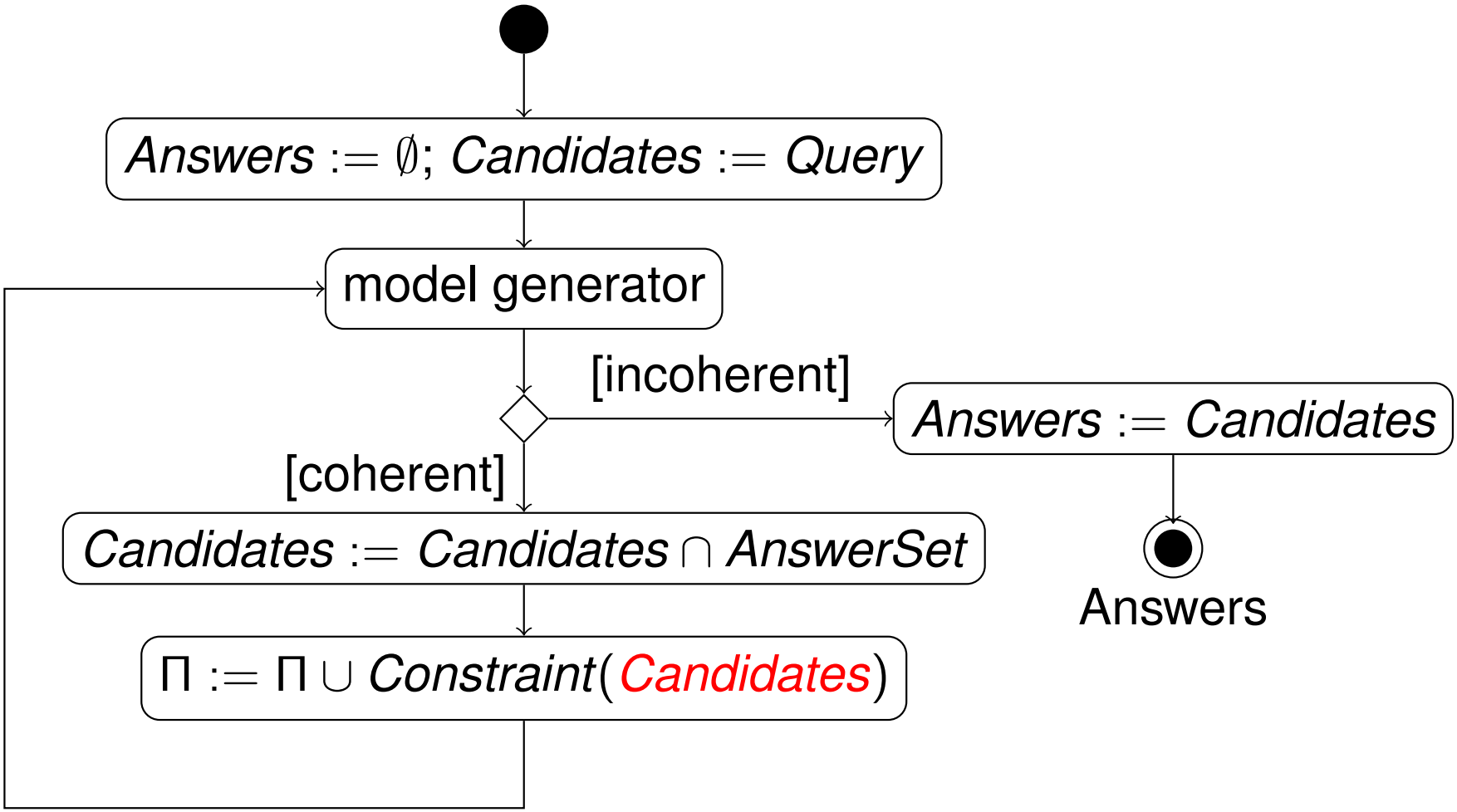
Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{a, c, e\}$	$\emptyset$	$\{a, c\}$
3	$\{b, c, d\}$	$\emptyset$	$\{c\}$
4	$\{b, c, e\}$	$\emptyset$	$\{c\}$
5	*Incoherent*	$\{c\}$	$\{c\}$

Cautious reasoning by enumeration of models

Cautious reasoning by enumeration of models

+ Easy to be implemented
- Redundant computation
- Poor performances
- Does not produce answers during the computation

Cautious reasoning by overestimate reduction



Example

Program Π

$a \leftarrow \mathit{not}\ b \quad b \leftarrow \mathit{not}\ a \quad$ % either a or b
$c \leftarrow a \quad c \leftarrow b$
$d \leftarrow \mathit{not}\ e \quad e \leftarrow \mathit{not}\ d \quad$ % either d or e

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

$\leftarrow a, c$ $\quad$ % added after step 1

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$

Example

Program Π

$a \leftarrow not\ b \quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a \quad c \leftarrow b$

$d \leftarrow not\ e \quad e \leftarrow not\ d$ % either d or e

$\leftarrow c$ % added after step 2

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{b, c, d\}$	$\emptyset$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b \quad b \leftarrow not\ a$ % either a or b
$c \leftarrow a \quad c \leftarrow b$
$d \leftarrow not\ e \quad e \leftarrow not\ d$ % either d or e

$\leftarrow c$ % added after step 2

Query Q

a, b, c

Execution

Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{b, c, d\}$	$\emptyset$	$\{c\}$
3	*Incoherent*	$\{c\}$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b \quad b \leftarrow not\ a$ % either a or b

$c \leftarrow a \quad c \leftarrow b$

$d \leftarrow not\ e \quad e \leftarrow not\ d$ % either d or e

$\leftarrow c$ % added after step 2

Query Q

a, b, c

Execution

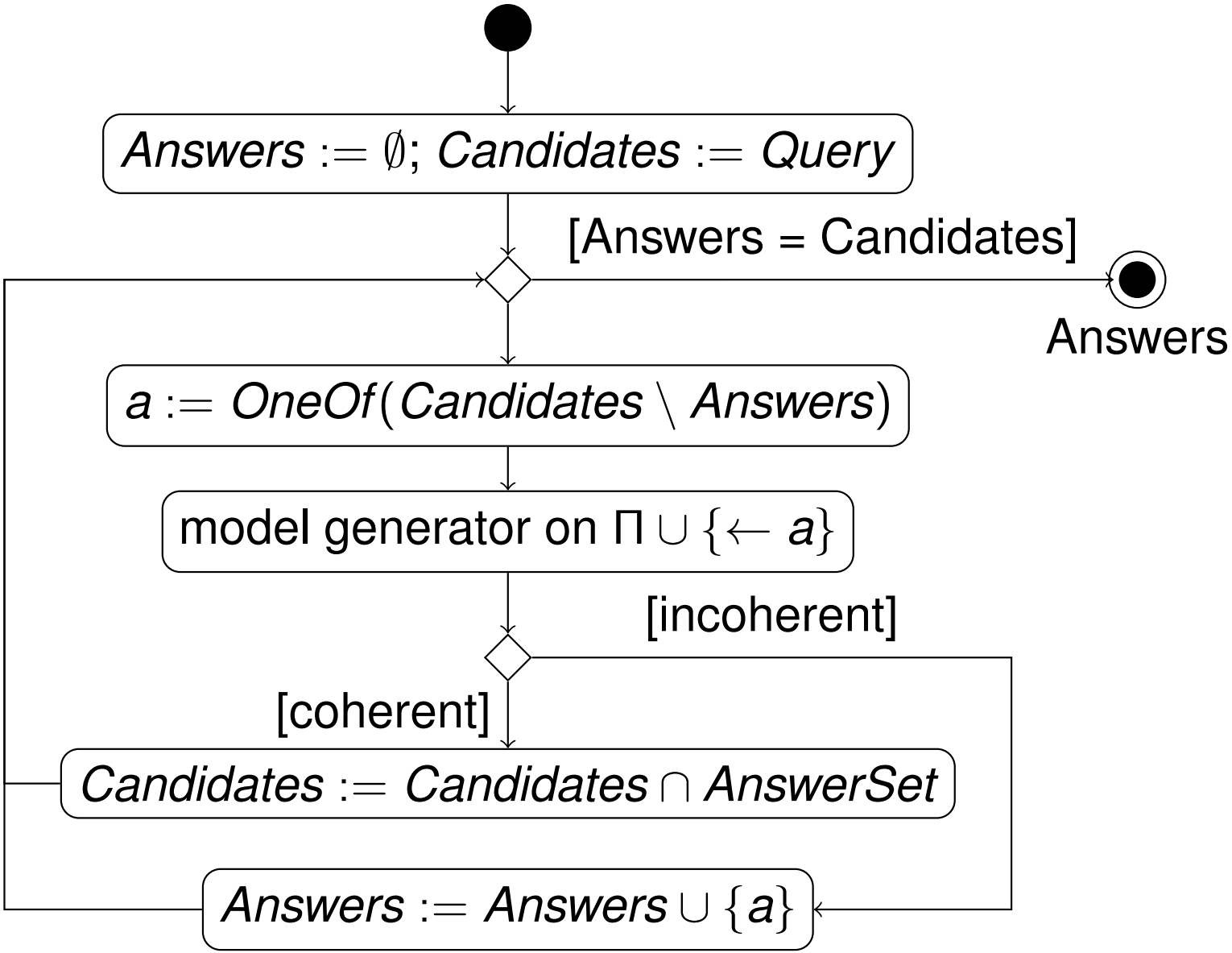
Step	Stable model	Underestimate	Overestimate
0		$\emptyset$	$\{a, b, c\}$
1	$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	$\{b, c, d\}$	$\emptyset$	$\{c\}$
3	*Incoherent*	$\{c\}$	$\{c\}$

Cautious reasoning by overestimate reduction

+ Force each stable model to remove at least one candidate from the overestimate

- Does not produce answers during the computation

Cautious reasoning by iterative coherence testing



Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

Query Q

a, b, c

Execution

Step	OneOf	Stable model	Underestimate	Overestimate
0			$\emptyset$	$\{a, b, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

Query Q

a, b, c

Execution

Step	OneOf	Stable model	Underestimate	Overestimate
0			$\emptyset$	$\{a, b, c\}$
1		$\{a, c, d\}$	$\emptyset$	$\{a, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

Query Q

a, b, c

Execution

Step	OneOf	Stable model	Underestimate	Overestimate
0			$\emptyset$	$\{a, b, c\}$
1		$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	c	*Incoherent*	$\{c\}$	$\{a, c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

Query Q

a, b, c

Execution

Step	OneOf	Stable model	Underestimate	Overestimate
0			$\emptyset$	$\{a, b, c\}$
1		$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	c	*Incoherent*	$\{c\}$	$\{a, c\}$
3	a	$\{b, c, d\}$	$\{c\}$	$\{c\}$

Example

Program Π

$a \leftarrow not\ b$ $\quad b \leftarrow not\ a$ $\quad$ % either a or b

$c \leftarrow a$ $\quad c \leftarrow b$

$d \leftarrow not\ e$ $\quad e \leftarrow not\ d$ $\quad$ % either d or e

Query Q

a, b, c

Execution

Step	OneOf	Stable model	Underestimate	Overestimate
0			$\emptyset$	$\{a, b, c\}$
1		$\{a, c, d\}$	$\emptyset$	$\{a, c\}$
2	c	*Incoherent*	$\{c\}$	$\{a, c\}$
3	a	$\{b, c, d\}$	$\{c\}$	$\{c\}$

Cautious reasoning by iterative coherence testing

+ Produce sound answers during the computation
+ Good performances
- The function OneOf is crucial for the performances

Anytime variants

- Often termination cannot be achieved in reasonable time
- Anytime algorithms are crucial for such cases to produce some sound answers

It is important to be anytime

The problem is Π_2^P-complete

$\Downarrow$

There are a few instances that are Π_2^P-hard

Anytime variants

- Often termination cannot be achieved in reasonable time
- Anytime algorithms are crucial for such cases to produce some sound answers

It is important to be anytime

The problem is Π_2^P-complete

$\Downarrow$

There are a few instances that are Π_2^P-hard

Good news!

Any algorithm for cautious reasoning can be anytime

Anytime variants

- Often termination cannot be achieved in reasonable time
- Anytime algorithms are crucial for such cases to produce some sound answers

It is important to be anytime

The problem is Π_2^P-complete

$\Downarrow$

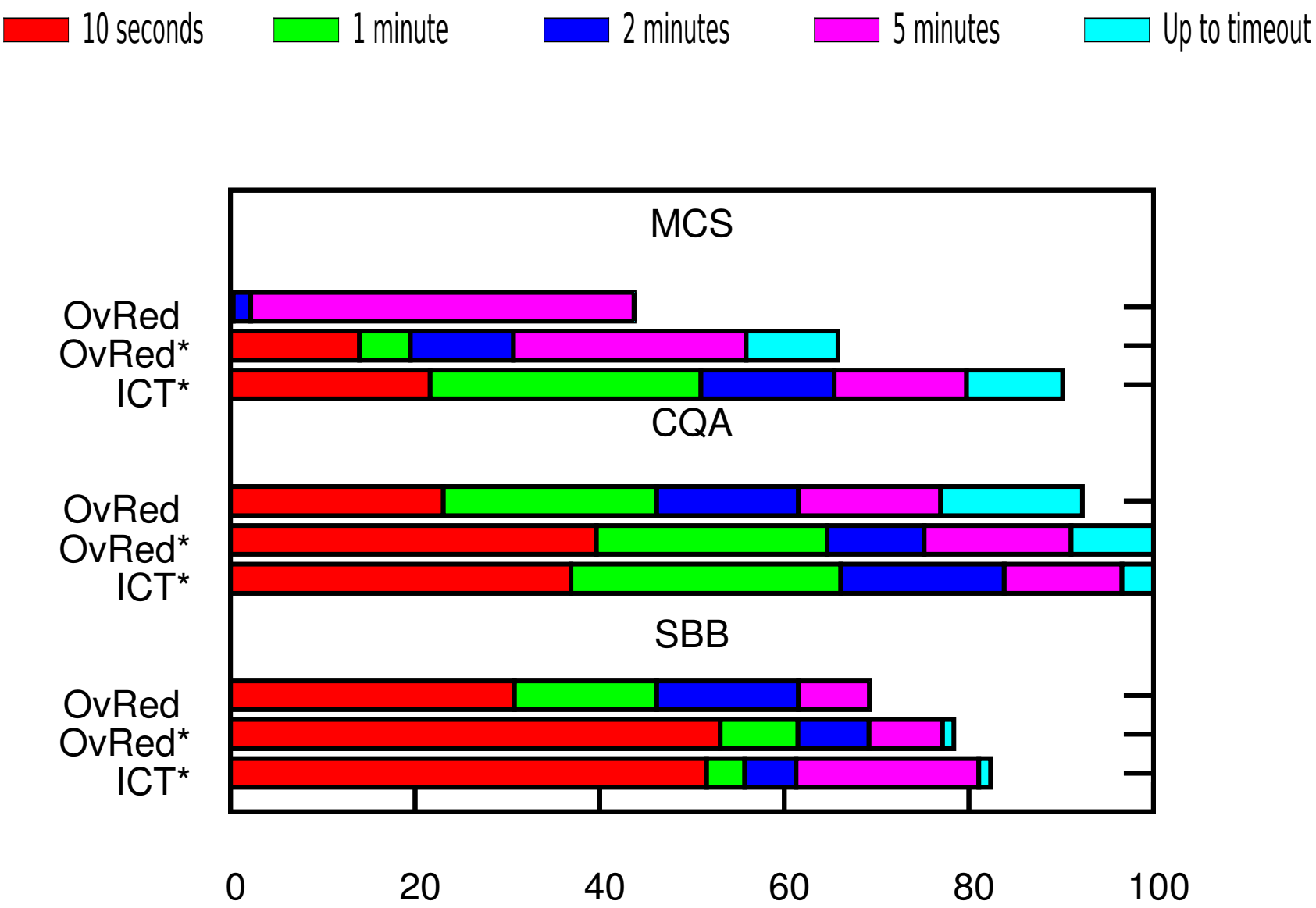
There are a few instances that are Π_2^P-hard

Good news!

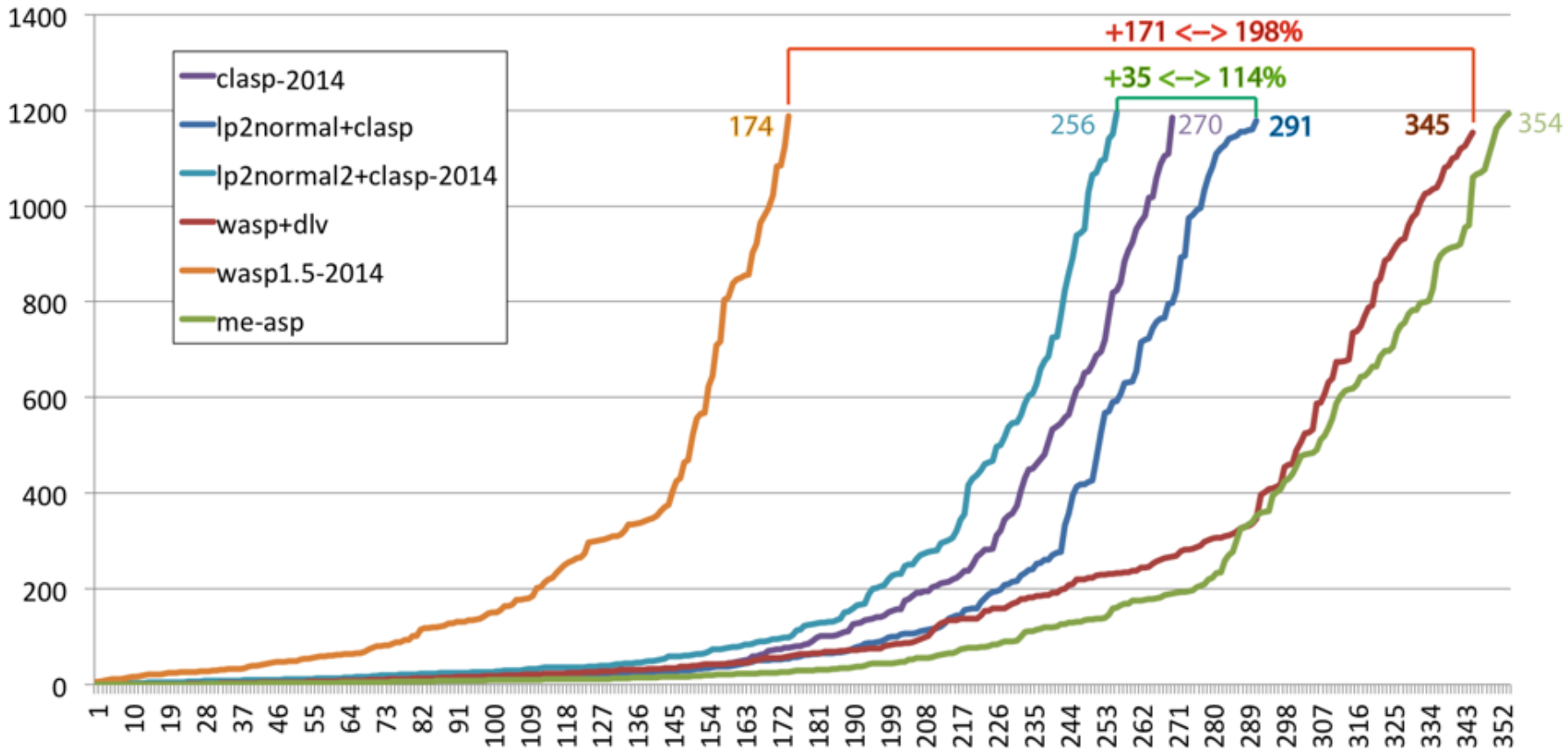
Any algorithm for cautious reasoning can be anytime

- Just check for new sound answers after each restart

Cautious reasoning



State of the art



Outline

Conclusion (1)

ASP solving require solutions for several computational tasks

- Model generation
  - Preprocessing, CDCL-like algorithm
- Optimum answer set search
  - Model and core-guided algorithms
- Cautious reasoning
  - Framework of anytime algorithms

Conclusion (2)

- ASP systems
  - CLASP (http://potassco.sourceforge.net/)
  - CMODELS (http://www.cs.utexas.edu/users/tag/cmodels/)
  - DLV (http://www.dlvsystem.com/)
  - IDP (https://dtai.cs.kuleuven.be/software/idp)
  - LP2SAT (http://research.ics.aalto.fi/software/asp/download/)
  - MEASP (https://www.mat.unical.it/ricca/me-asp/)
  - WASP (http://alviano.github.io/wasp/)

Thank you!

Bibliography (1)

[Arenas et al., TPLP 2003] M. Arenas, L. E. Bertossi, J. Chomicki. *Answer sets for consistent query answering in inconsistent databases*. TPLP 3(4-5): 393-424 (2003).

[Balduccini et al., LPNMR 2001] M. Balduccini, M. Gelfond, R. Watson, M. Nogueira. *The USA-Advisor: A Case Study in Answer Set Planning*. LPNMR 2001: 439-442.

[Eén and Biere, SAT 2005] N. Eén, A. Biere. *Effective Preprocessing in SAT Through Variable and Clause Elimination*. SAT 2005: 61-75.

[Eiter, LPNMR 2005] T. Eiter. *Data Integration and Answer Set Programming*. LPNMR 2005: 13-25.

[Gebser et al., TPLP 2011] M. Gebser, T. Schaub, S. Thiele, P. Veber. *Detecting inconsistencies in large biological networks with answer set programming*. TPLP 11(2-3): 323-360 (2011).

Bibliography (2)

[Koch et al., Artif. Intell. 2003] C. Koch, N. Leone, G. Pfeifer. *Enhancing disjunctive logic programming systems by SAT checkers*. Artif. Intell. 151(1-2): 177-212 (2003).

[Koponen et al., TPLP 2015] L. Koponen, E. Oikarinen, T. Janhunen, L. Säilä. *Optimizing phylogenetic supertrees using answer set programming*. TPLP 15(4-5): 604-619 (2015).

[Marra et al., JELIA 2014] G. Marra, F. Ricca, G. Terracina, D. Ursino. *Exploiting Answer Set Programming for Handling Information Diffusion in a Multi-Social-Network Scenario*. JELIA 2014: 618-627.

[Simons et al., Artif. Intell. 2002] P. Simons, I. Niemelä, T. Soininen. *Extending and implementing the stable model semantics*. Artif. Intell. 138(1-2): 181-234 (2002).