Tuesday

Computation of justifications

Schedule for today

- Justifications
- Glass box and black box techniques
- Use of justifications for other inference services

Justifications and associated services

Justifications

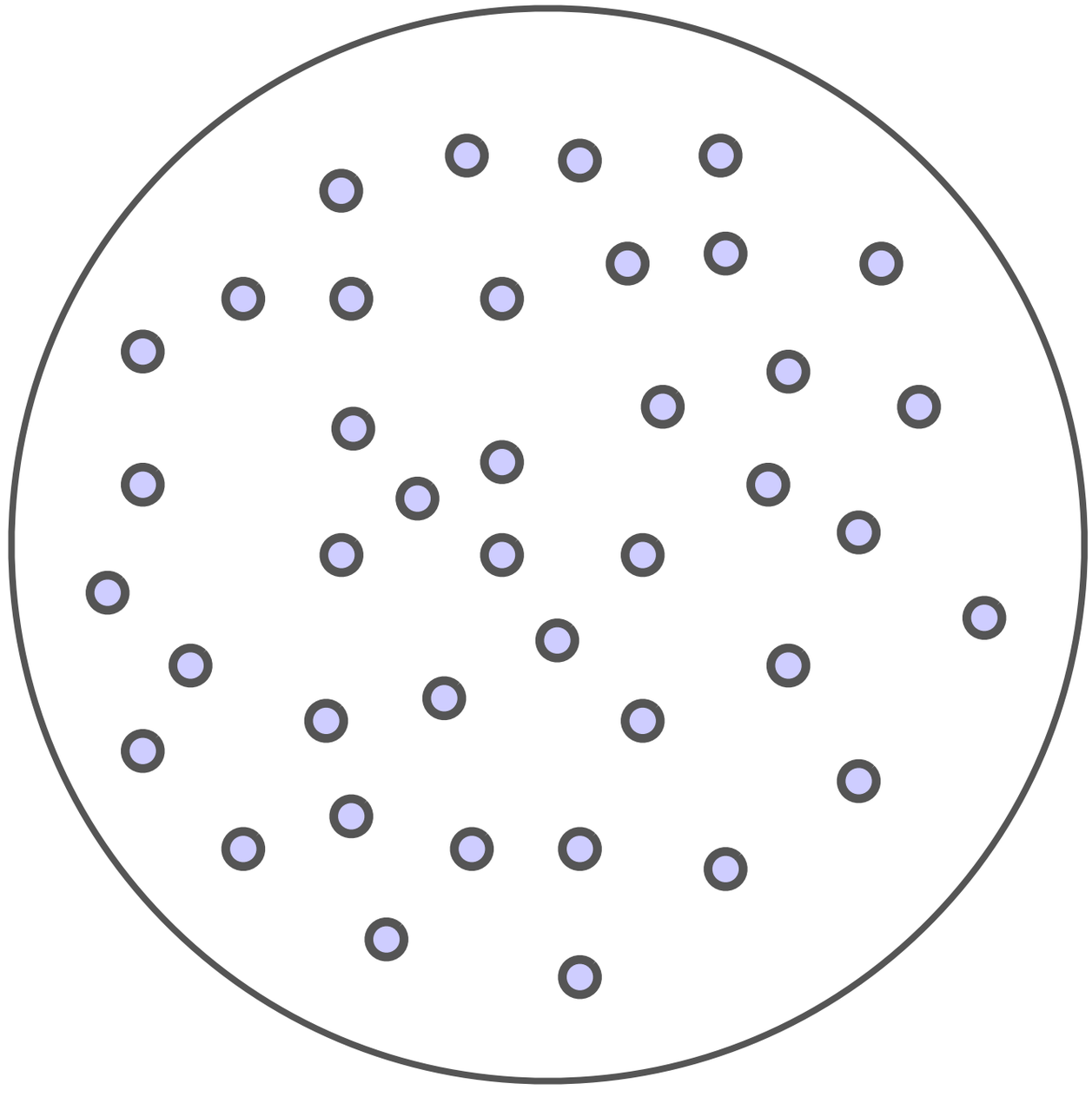
- Justifications are a kind of explanation
- Justifications are minimal subsets of an ontology that are sufficient for a given entailment to hold
- Also known as

  MUPS – *Minimal Unsatisfiability Preserving Sub-TBoxes*

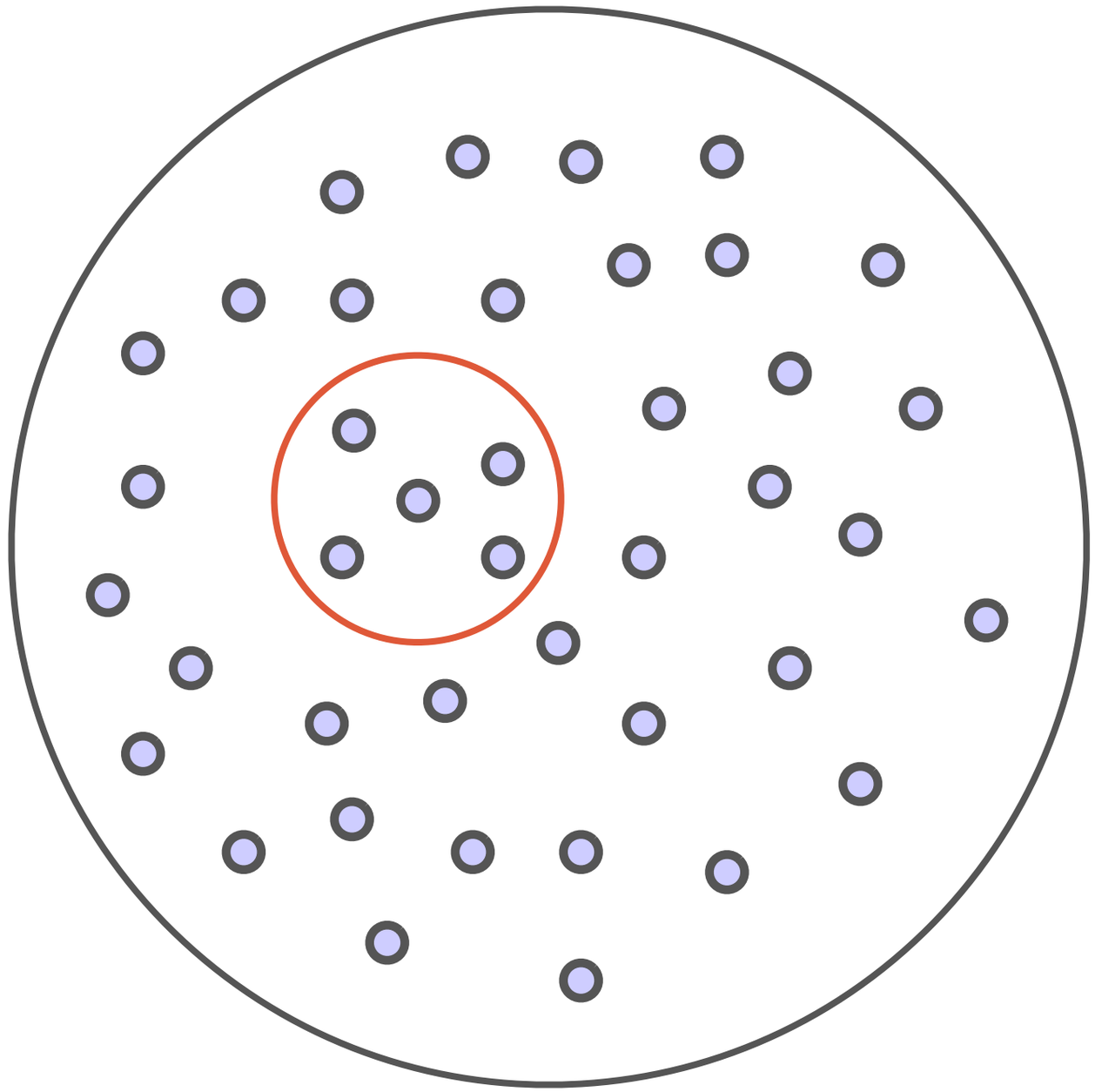
  MinAs – *Minimal Axiom sets*

  Kernels

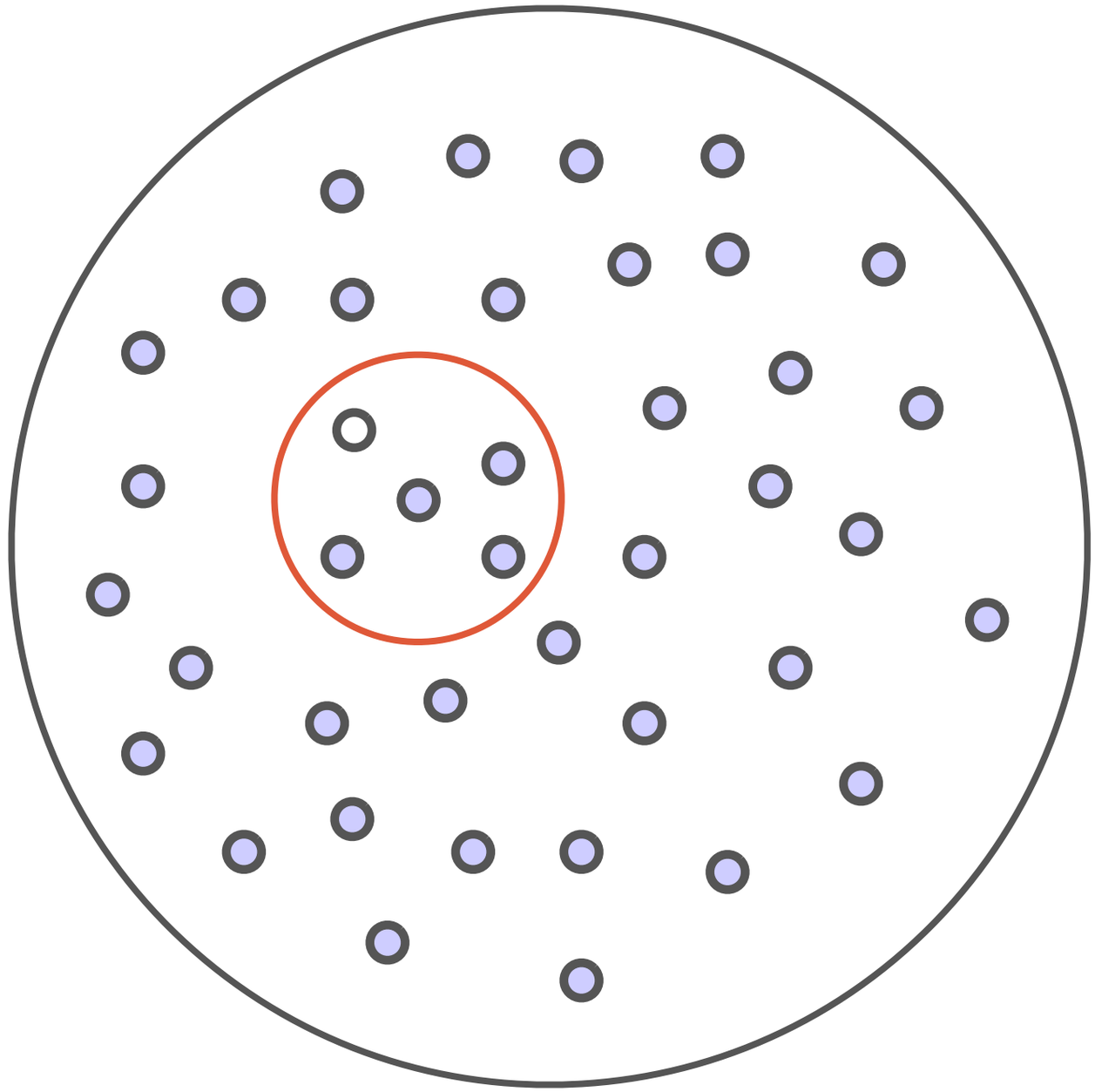
Justifications



Justifications



Justifications

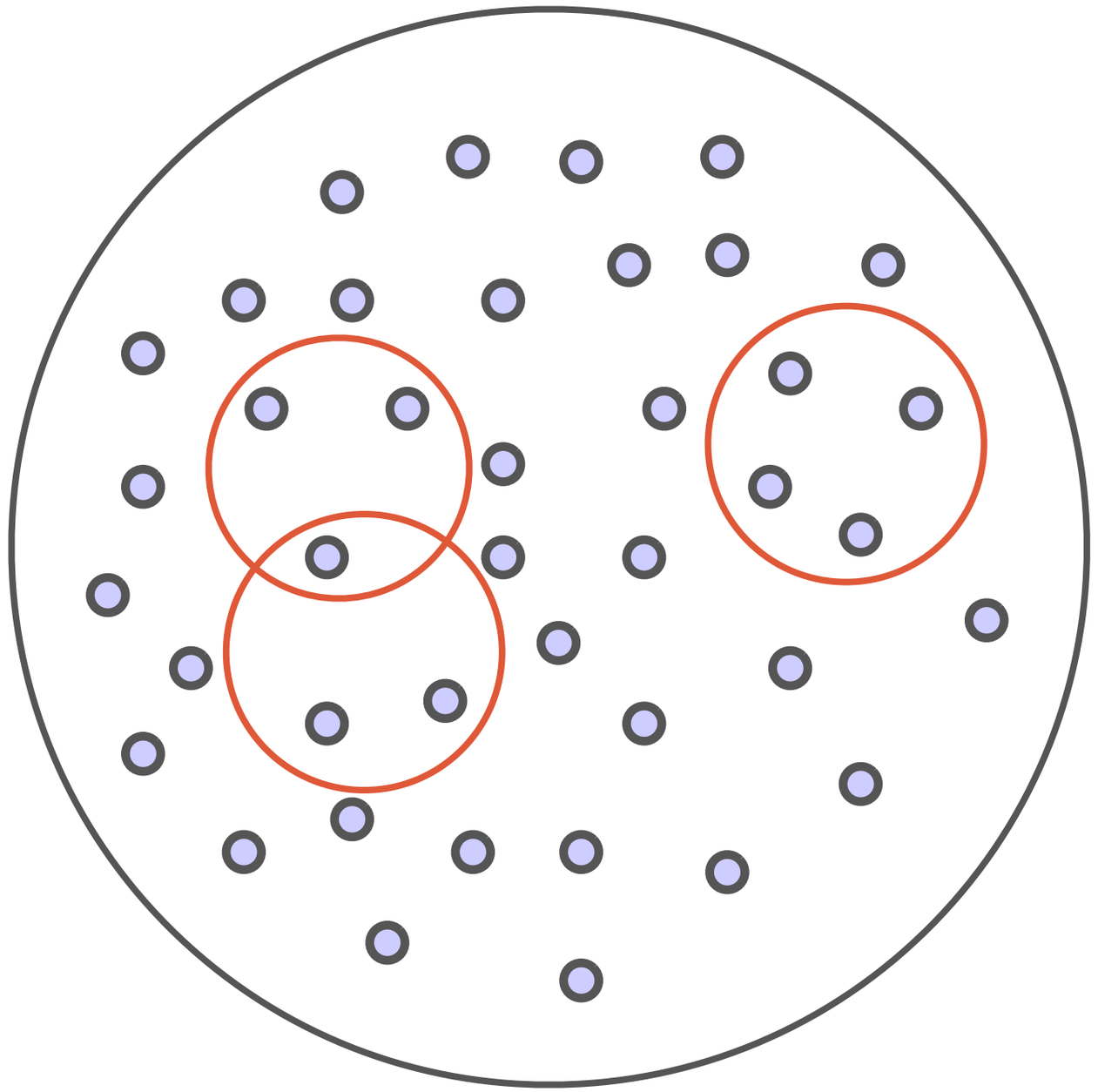


Minimality: remove an axiom ~> entailment no longer holds

MANCHESTER 1824

The University of Manchester

Justifications



In general, more than one justification for a given entailment

Justifications

Justifications

$$\mathcal{O} = \{\alpha_1, \alpha_2 \ldots \alpha_n\}$$

Justifications

$$\mathcal{O} = \{\alpha_1, \alpha_2 \ldots \alpha_n\} \qquad \mathcal{O} \models \eta$$

Justifications

$$\mathcal{O} = \{\alpha_1, \alpha_2 \ldots \alpha_n\} \qquad \mathcal{O} \models \eta$$

Justifications

$$\mathcal{O} = \{\alpha_1, \alpha_2 \ldots \alpha_n\} \qquad \mathcal{O} \models \eta$$

$$J \subseteq \mathcal{O} \qquad J \models \eta$$

Justifications

$$\mathcal{O} = \{\alpha_1, \alpha_2 \ldots \alpha_n\} \qquad \mathcal{O} \models \eta$$

$$J \subseteq \mathcal{O} \qquad J \models \eta$$

$$\forall J' \subset J \qquad J' \not\models \eta$$

Example

Example

$$\mathcal{O} = \{$$

$$\}$$

Example

$$
\begin{aligned}
\mathcal{O} = \{1: \quad & C \sqsubseteq B \sqcap D \\
2: \quad & B \sqsubseteq \exists R F \\
3: \quad & D \sqsubseteq \forall R \neg F \\
4: \quad & C \sqsubseteq G \\
5: \quad & C \sqsubseteq H \sqcap \neg G\}
\end{aligned}
$$

Example

$$\begin{aligned}\mathcal{O} = \{1: \quad & C \sqsubseteq B \sqcap D \\ 2: \quad & B \sqsubseteq \exists R F \\ 3: \quad & D \sqsubseteq \forall R \neg F \\ 4: \quad & C \sqsubseteq G \\ 5: \quad & C \sqsubseteq H \sqcap \neg G\}\end{aligned}$$

$$\mathcal{O} \models C \equiv \bot$$

Example

$$\mathcal{O} = \{ 1: \quad C \sqsubseteq B \sqcap D$$
$$2: \quad B \sqsubseteq \exists R F$$
$$3: \quad D \sqsubseteq \forall R \neg F$$
$$4: \quad C \sqsubseteq G$$
$$5: \quad C \sqsubseteq H \sqcap \neg G \}$$

$$\mathcal{O} \models C \equiv \bot$$

$$J = \{1, 2, 3\}$$

Multiple justifications

$$
\begin{aligned}
\mathcal{O} = \{\ 1: \quad & C \sqsubseteq B \sqcap D \\
2: \quad & B \sqsubseteq \exists R F \\
3: \quad & D \sqsubseteq \forall R \neg F \\
4: \quad & C \sqsubseteq G \\
5: \quad & C \sqsubseteq H \sqcap \neg G\ \}
\end{aligned}
$$

Multiple justifications

$$\mathcal{O} = \{ \begin{array}{ll} 1: & C \sqsubseteq B \sqcap D \\ 2: & B \sqsubseteq \exists R F \\ 3: & D \sqsubseteq \forall R \neg F \\ 4: & C \sqsubseteq G \\ 5: & C \sqsubseteq H \sqcap \neg G \end{array} \}$$

Multiple justifications

$$\mathcal{O} = \{ \begin{array}{ll} 1: & C \sqsubseteq B \sqcap D \\ 2: & B \sqsubseteq \exists R F \\ 3: & D \sqsubseteq \forall R \neg F \\ 4: & C \sqsubseteq G \\ 5: & C \sqsubseteq H \sqcap \neg G \end{array} \}$$

Multiple justifications

- Entailments can have multiple justifications.
- If an entailment has multiple justifications, they may overlap.
- Removing one axiom from the justification breaks the justification: the entailment is no longer supported by the remaining axioms. This is a repair.

Repairs

- For a repair, at least one axiom from every justification needs to be removed.
- ~> For a repair plan, all just's are needed
- Examples:

  MYGRID ontology (ontology of web services):
  on average **9** justifications per entailment

  PIZZA ontology: for some entailments **>100** just's

  ➡ Indication of redundancy?

  ➡ Maintenance nightmare?

User Issues

- Small numbers of small justificaitons
  - Easy and pleasant to inspect
- Small numbers of large justificaitons
  - Better than alternatives
- Large numbers of justifications
  - Pretty hopeless with current mechanisms

User Issues

- Small numbers of small justificaitons
  - Easy and pleasant to inspect
- Small numbers of large justificaitons
  - Better than alternatives
- Large numbers of justifications
  - Pretty hopeless with current mechanisms

How to manage these issues?

Remember the Task

- We've detected unsatisfiable classes
  - 144! Bugs!
    - Few justs per UC and we're toast
- In manual debugging
  - We had a notion of a key or root class
  - The source of unsatisfiablity

Root/derived unsat. classes

- A class whose satisfiability depends on another class is a derived unsatisfiable class.
- All other unsatisfiable classes are root unsatisfiable classes.

They cannot be fixed by making other classes satisfiable.

Root/derived unsat. classes

- A class whose satisfiability depends on another class is a derived unsatisfiable class.
- All other unsatisfiable classes are root unsatisfiable classes.

> Root unsatisfiable classes should be examined and fixed first.

They cannot be fixed by making other classes satisfiable.

Root/derived unsat. classes (UCs)

More precisely, using justifications:

- A class is a derived UC if it has a justification that is a strict superset of a justification for some other UC.
- Any other UC is a root UC.

	Koala subClassOf Nothing
1)	Koala subClassOf isHardWorking value false
2)	isHardWorking domain Person
3)	Koala subClassOf Marsupials
4)	Marsupials disjointWith Person

Explanation 1 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	hasDegree domain Person	1	
3)	Koala subClassOf Marsupials	2	
4)	Marsupials disjointWith Person	2	

Explanation 2 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	Koala subClassOf isHardWorking value false	1	
3)	isHardWorking domain Person	1	
4)	Koala subClassOf Marsupials	2	
5)	Marsupials disjointWith Person	2	

	Koala subClassOf Nothing
1)	Koala subClassOf isHardWorking value false
2)	isHardWorking domain Person
3)	Koala subClassOf Marsupials
4)	Marsupials disjointWith Person

Explanation 1 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	hasDegree domain Person	1	
3)	Koala subClassOf Marsupials	2	
4)	Marsupials disjointWith Person	2	

Explanation 2 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	Koala subClassOf isHardWorking value false	1	
3)	isHardWorking domain Person	1	
4)	Koala subClassOf Marsupials	2	
5)	Marsupials disjointWith Person	2	

	Koala subClassOf Nothing
1)	Koala subClassOf isHardWorking value false
2)	isHardWorking domain Person
3)	Koala subClassOf Marsupials
4)	Marsupials disjointWith Person

Explanation 1 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	hasDegree domain Person	1	
3)	Koala subClassOf Marsupials	2	
4)	Marsupials disjointWith Person	2	

Explanation 2 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	Koala subClassOf isHardWorking value false	1	
3)	isHardWorking domain Person	1	
4)	Koala subClassOf Marsupials	2	
5)	Marsupials disjointWith Person	2	

	Koala subClassOf Nothing
1)	Koala subClassOf isHardWorking value false
2)	isHardWorking domain Person
3)	Koala subClassOf Marsupials
4)	Marsupials disjointWith Person

Explanation 1 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	hasDegree domain Person	1	
3)	Koala subClassOf Marsupials	2	
4)	Marsupials disjointWith Person	2	

Explanation 2 (Entailment 1) Display laconic explanation Dump lemmas

	KoalaWithPhD subClassOf Nothing		
1)	KoalaWithPhD equivalentTo Koala and hasDegree value PhD	2	
2)	Koala subClassOf isHardWorking value false	1	
3)	isHardWorking domain Person	1	
4)	Koala subClassOf Marsupials	2	
5)	Marsupials disjointWith Person	2	

Focusing on Roots reduces the cognitive burden

Suggestive

- Root focusing is useful
  - If you are debugging UCs
  - There are lots of them
    - They are related
- Other metrics can help
  - "Power" of an axiom
- Perhaps patterns as well as overlap?

Suggestive

- Root focusing is useful
  - If you are debugging UCs
  - There are lots of them
    - They are related
- Other metrics can help
  - "Power" of an axiom
- Perhaps patterns as well as overlap?

Imposed methodology helpful!

Derived unsat. classes

- Partial derived UCs: derived UCs for which there is some justification that is not a strict superset of a justification for another UC
- Pure derived UCs: all other derived UCs

RECIPE

Derived unsat. classes

- Partial derived UCs: derived UCs for which there is some justification that is not a strict superset of a justification for another UC
- Pure derived UCs: all other derived UCs

RECIPE

```
While root UCs exist {
    Fix root UCs
    // this may turn some partial derived UCs into root UCs
    Recompute root UCs
}
```

Example

$$
\begin{array}{ll}
\alpha_1 & A \sqsubseteq E \sqcap \neg E \\
\alpha_2 & B \sqsubseteq \exists R.A \\
\alpha_3 & B \sqsubseteq \forall R.\neg A \\
\alpha_4 & C \sqsubseteq B \\
\alpha_5 & \neg C \equiv \forall S.\bot \\
\alpha_6 & \top \sqsubseteq \exists S.\top \\
\alpha_7 & D \sqsubseteq B
\end{array}
$$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$
$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$ *root*

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$
$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$ *root*

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$ *partial*

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$
$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$ *root*

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$ *partial*

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$ *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$ *root*

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$ *partial*

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$ *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

pure

Example

α_1	$A \sqsubseteq E \sqcap \neg E$
α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

A unsat: $\{\alpha_1\}$ *root*

B unsat: $\{\alpha_1, \alpha_2\}$ $\{\alpha_2, \alpha_3\}$ *partial*

C unsat: $\{\alpha_1, \alpha_2, \alpha_4\}$ *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_1, \alpha_2, \alpha_7\}$ $\{\alpha_2, \alpha_3, \alpha_7\}$

pure

Repair 1: delete α_1

Example

α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

B unsat: $\{\alpha_2, \alpha_3\}$

C unsat:

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

B unsat: $\{\alpha_2, \alpha_3\}$ *root*

C unsat:

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

B unsat: $\{\alpha_2, \alpha_3\}$ *root*

C unsat: *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_2, \alpha_3, \alpha_7\}$

Example

α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

B unsat: $\{\alpha_2, \alpha_3\}$ *root*

C unsat: *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_2, \alpha_3, \alpha_7\}$

pure

Example

α_2	$B \sqsubseteq \exists R.A$
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

B unsat: $\{\alpha_2, \alpha_3\}$ *root*

C unsat: *partial*

$\{\alpha_2, \alpha_3, \alpha_4\}$ $\{\alpha_5, \alpha_6\}$

D unsat: $\{\alpha_2, \alpha_3, \alpha_7\}$

pure

Repair 2: delete α_2

Example

α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

C unsat:

$\{\alpha_5, \alpha_6\}$

Example

α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

C unsat: *root*

$\{\alpha_5, \alpha_6\}$

Example

α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_5	$\neg C \equiv \forall S.\bot$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

C unsat: *root*

$\{\alpha_5, \alpha_6\}$

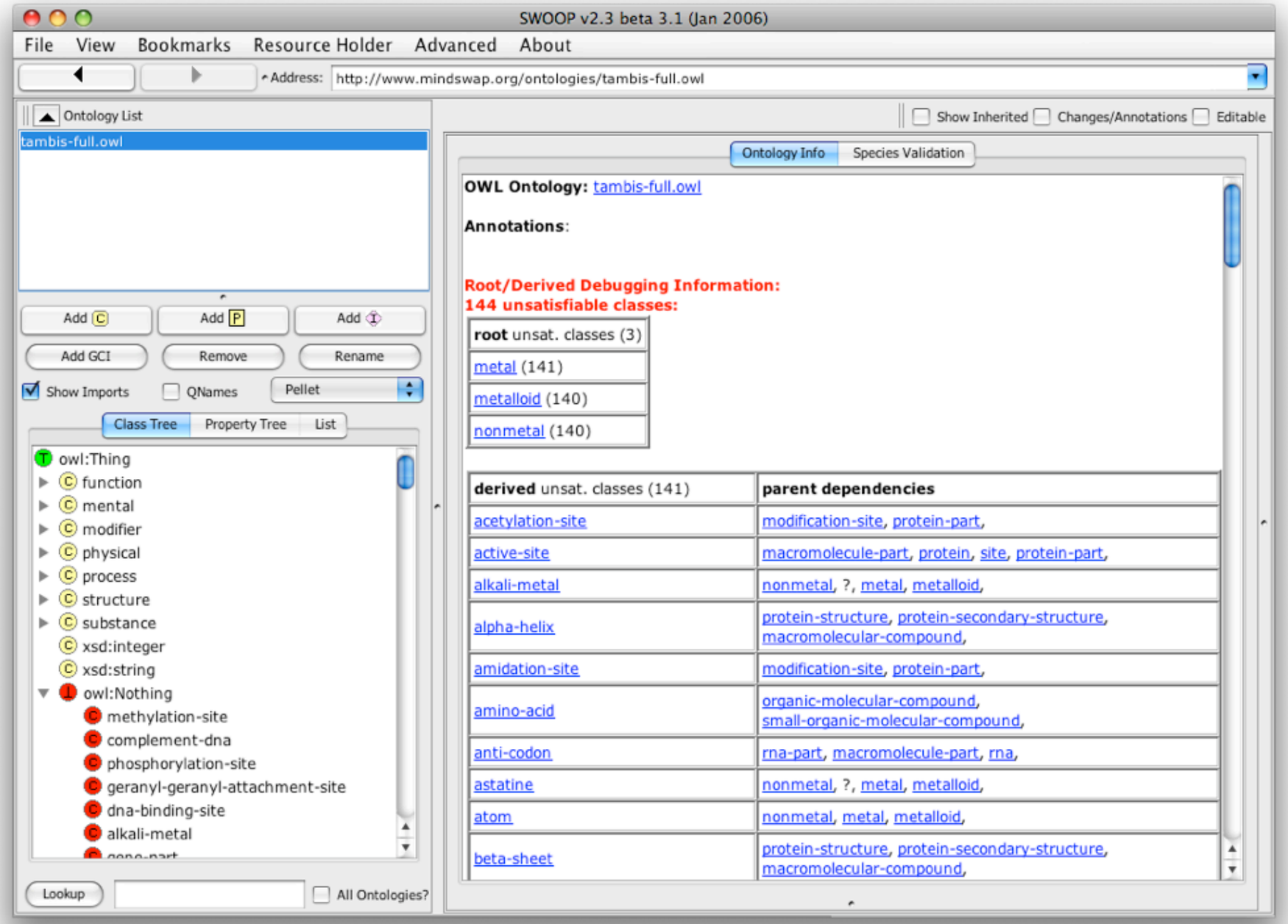
Repair 3: delete α_5

Example

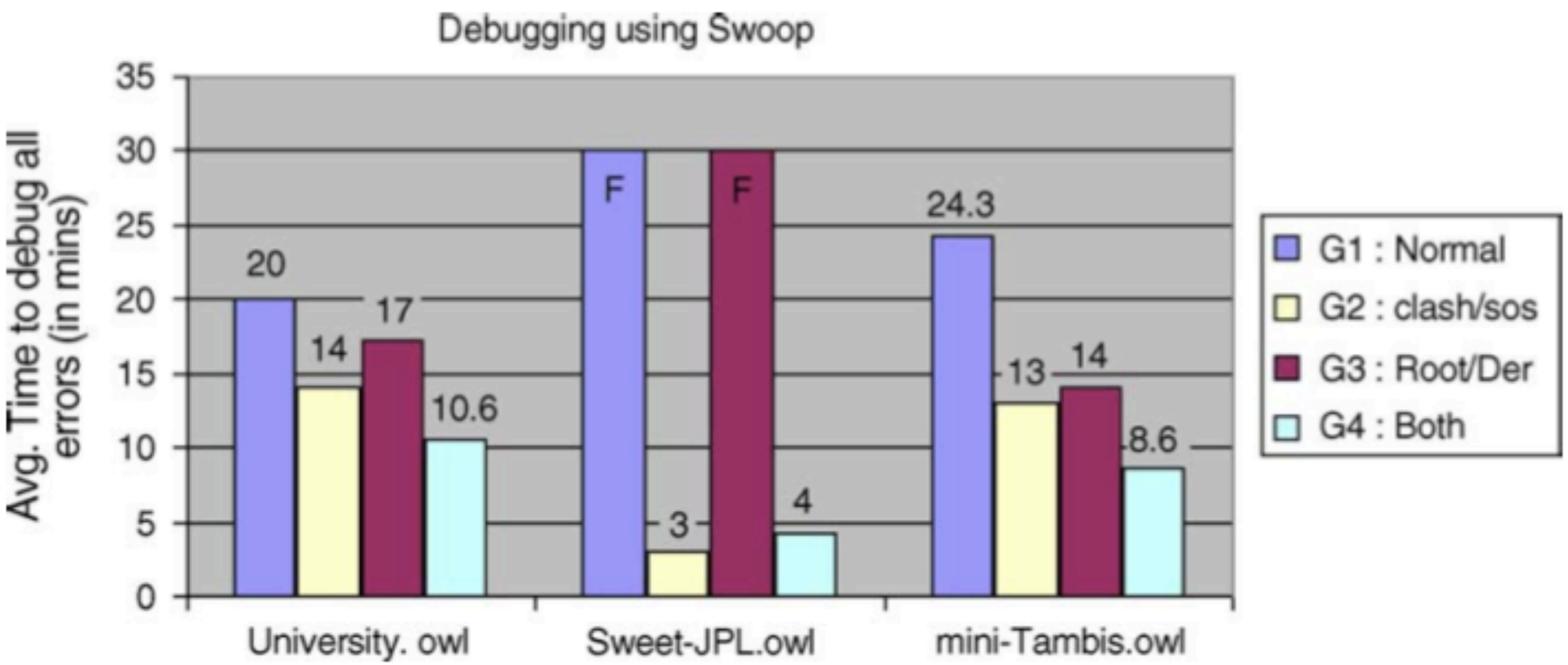
α_3	$B \sqsubseteq \forall R.\neg A$
α_4	$C \sqsubseteq B$
α_6	$\top \sqsubseteq \exists S.\top$
α_7	$D \sqsubseteq B$

Repair finished! ☺

Root/Derived Pinpointing



Combining Services



Concise Format | Abstract Syntax | Graphical View | Natural Language | RDF/XML | Turtle

OWL-Class: OceanCrustLayer

Unsatisfiable concept

Reason: Any member of OceanCrustLayer has more than one value for the functional property hasDimension

Axioms causing the problem:

1) (OceanCrustLayer ⊑ OceanRegion)
2) |_(OceanRegion ⊑ TopographicalRegion)
3) |_(TopographicalRegion ⊑ EarthRegion)
4) |_(EarthRegion ⊑ Region)
5) |_(Region ⊑ GeometricalObject_2D)
6) |_(GeometricalObject_2D ⊑ (∃hasDimension . {"2"^^<xsd:integer>}))
7) |_Functional Property (hasDimension)
8) (OceanCrustLayer ⊑ CrustLayer)
9) |_(CrustLayer ⊑ LithosphereLayer)
10) |_(LithosphereLayer ⊑ SolidEarthLayer)
11) |_(SolidEarthLayer ⊑ Layer)
12) |_(Layer ⊑ GeometricalObject_3D)
13) |_(GeometricalObject_3D ⊑ (∃hasDimension . {"3"^^<xsd:integer>}))

Equivalent to: (Add)
owl:Nothing

Subclass of: (Add)
CrustLayer (Delete)
OceanRegion (Delete)

Lessons

- Service and Tool support is necessary
  - Justifications are core
  - Other services may be useful
- Justifications are computable
  - That's for tomorrow
- Interaction mechanisms matter
  - But do not dominate

Glass box and black box techniques

Entailments to Unsatisfiable Expressions

$$\mathcal{O} \models C \sqsubseteq D$$

Entailments to Unsatisfiable Expressions

$$\mathcal{O} \models C \sqsubseteq D$$

$$\mathcal{O} \models C \sqcap \neg D \equiv \bot$$

Entailments to Unsatisfiable Expressions

$$\mathcal{O} \models C \sqsubseteq D$$

$$\mathcal{O} \models \boxed{C \sqcap \neg D} \equiv \bot$$

Computing Justifications

Input:
Ontology, O
Entailment, η

Find all justifications
Model diagnosis techniques
(Reiter - Hitting Sets)

Find one justification
Sub-procedure
(**Black-box** or **Glass-box**
implementation techniques)

Output:
A set of justifications for η in O

Find all justifications
Pure **Glass-box**
(Note shown practical for even weak logics)

Computing Justifications

- Implementations of a service for computing justifications can be split into two main categories:
  - Glass-box
  - Black-box

Glass-box

- Glass-box techniques are specific to a particular reasoner
- For an existing reasoner, implementing glass box tracing requires a thorough and non-trivial modification of the reasoner internals
- Examples: Pellet, CEL, DION

Glass Box

- DL reasoners typically* use tableau algorithms:
  - Proof by refutation:
    To prove $\mathcal{O} \models \eta$, we check consistency of $\mathcal{O} \cup \{\neg\eta\}$
  - Build a completion graph;
    Ontology is inconsistent if complection graph is closed, i.e., every branch has a clash (contradiction)
- Tableau Tracing
  - Track the axioms responsible for each modification to the completion graph
  - Identify clash-causing axioms

**Changing, but 3 out of 4 top reasoners do*

MANCHESTER 1824

The University of Manchester

Tableau Tracing

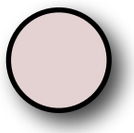
$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

MeatPizza

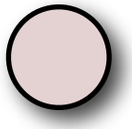


The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

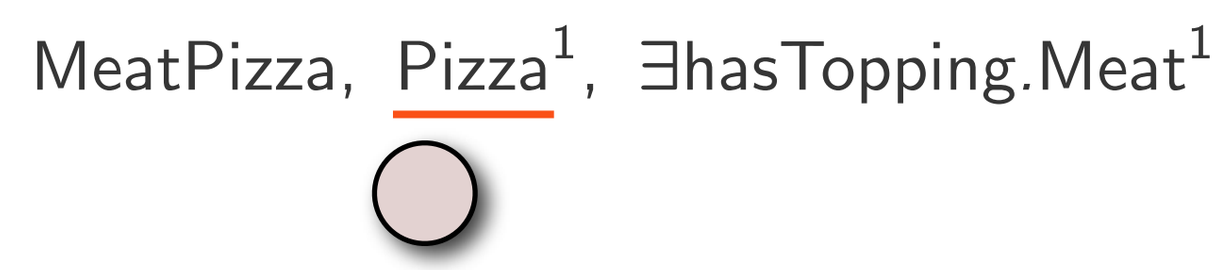
$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1}$



The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$



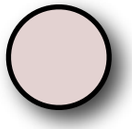
MANCHESTER 1824

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \underline{\exists \mathsf{hasTopping}.\mathsf{Meat}^{1}},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$$

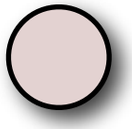


Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$

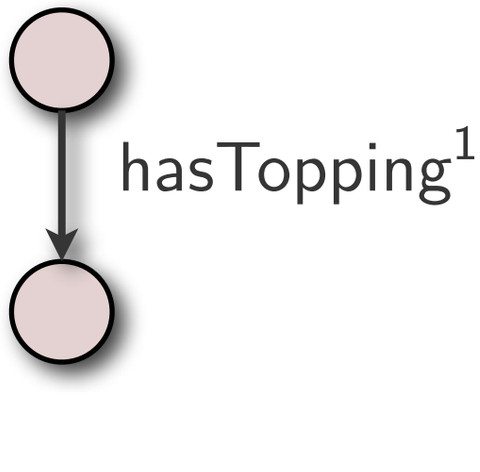
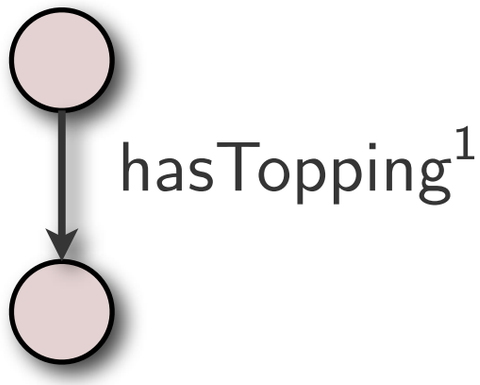


Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



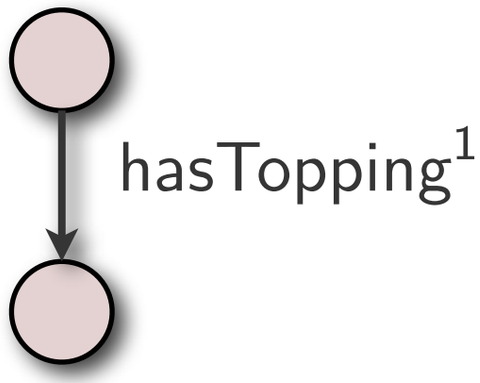
Meat^{1}

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



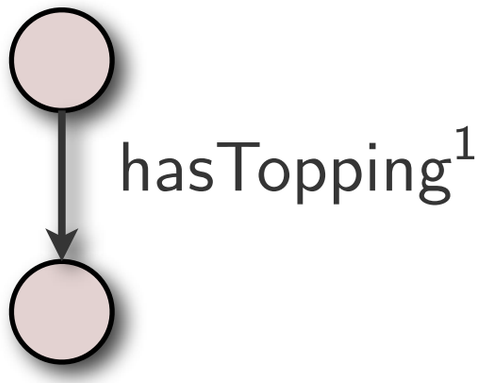
$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg \mathsf{Vegetable}^{1,3}$

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



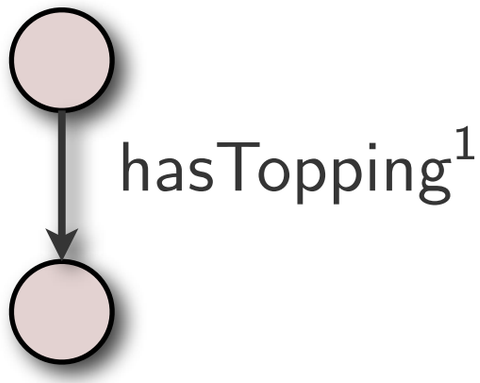
$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg \mathsf{Vegetable}^{1,3}$

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg \mathsf{Vegetable}^{1,3},\ \mathsf{Cheese}^{1,2}$

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

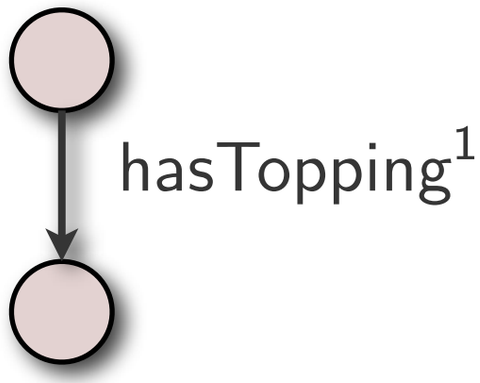
$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$

$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg \mathsf{Vegetable}^{1,3},\ \underline{\mathsf{Cheese}}^{1,2}$

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$



$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg \mathsf{Vegetable}^{1,3},\ \mathsf{Cheese}^{1,2},\ \neg \mathsf{Meat}^{1,2,4}$

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$

$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg\mathsf{Vegetable}^{1,3},\ \mathsf{Cheese}^{1,2},\ \neg\mathsf{Meat}^{1,2,4}$

Clash! ~> Justification = {1,2,4}

The University of Manchester

Tableau Tracing

$$
\begin{array}{rrcl}
1 & \mathsf{MeatPizza} & \sqsubseteq & \mathsf{Pizza} \sqcap \exists \mathsf{hasTopping}.\mathsf{Meat} \\
2 & \mathsf{Pizza} & \sqsubseteq & \forall \mathsf{hasTopping}.\mathsf{Cheese} \\
3 & \mathsf{Meat} & \sqsubseteq & \mathsf{Food} \sqcap \neg \mathsf{Vegetable} \\
4 & \mathsf{Cheese} & \sqsubseteq & \neg \mathsf{Meat}
\end{array}
$$

$\mathsf{MeatPizza},\ \mathsf{Pizza}^{1},\ \exists \mathsf{hasTopping}.\mathsf{Meat}^{1},\ \forall \mathsf{hasTopping}.\mathsf{Cheese}^{1,2}$

$\mathsf{Meat}^{1},\ \mathsf{Food}^{1,3},\ \neg\mathsf{Vegetable}^{1,3},\ \mathsf{Cheese}^{1,2},\ \neg\mathsf{Meat}^{1,2,4}$

Clash! ~> Justification = {1,2,4}

MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 1:

Internal modifications done by the reasoner

- Normalisation—involves rewriting terms
  e.g., $\neg\exists R.C \rightsquigarrow \forall R.\neg C$
- Absorption—involves combining axioms
  e.g., $C \sqcap D \sqsubseteq E, \quad C \sqsubseteq F \rightsquigarrow C \sqsubseteq F \sqcap (E \sqcup \neg D)$

MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 1:

Internal modifications done by the reasoner

- **Normalisation**—involves rewriting terms
  e.g., $\neg\exists R.C \rightsquigarrow \forall R.\neg C$
- **Absorption**—involves combining axioms
  e.g., $C \sqcap D \sqsubseteq E, \quad C \sqsubseteq F \rightsquigarrow C \sqsubseteq F \sqcap (E \sqcup \neg D)$

Solution: trace through preprocessing

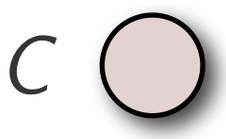
MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event
e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$



MANCHESTER 1824

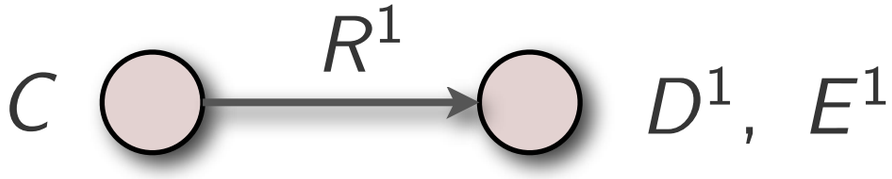
The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event

e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$



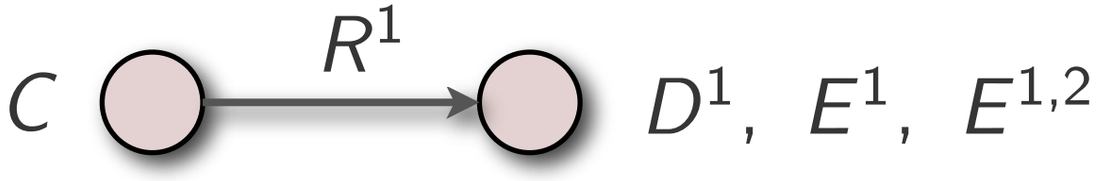
MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event
e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$



MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event

e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$

$D^1,\ E^1,\ E^{1,2},\ (\neg E)^{1,3}$

MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event
e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$

Solution: modify trace to be a set of axiom sets

MANCHESTER 1824

The University of Manchester

Tableau Tracing

Key challenge 2:

More than one reason for an event
e.g., (1) $C \sqsubseteq \exists R.(D \sqcap E)$ (2) $C \sqsubseteq \forall R.E$ (3) $D \sqsubseteq \neg E$

Solution: modify trace to be a set of axiom sets

Tableau Tracing

Additional challenge 1:

Determine whether a clash depends on a choice/merge

e.g., (1) $C \sqsubseteq \exists R.E$ (2) $C \sqsubseteq \exists R.D$

(3) $C \sqsubseteq \leqslant 1R$ (4) $C \sqsubseteq \forall R.\neg D$

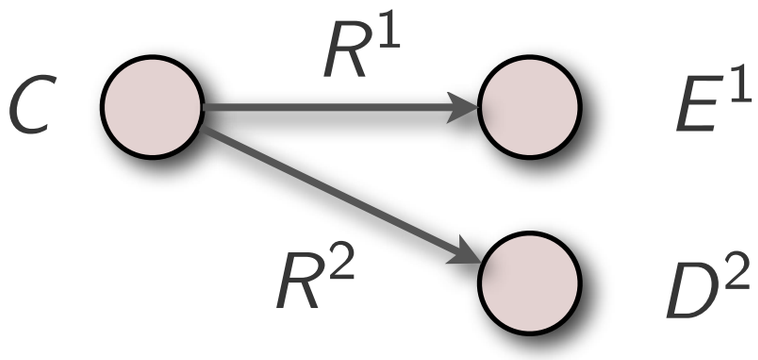


Tableau Tracing

Additional challenge 1:

Determine whether a clash depends on a choice/merge

e.g., (1) $C \sqsubseteq \exists R.E$ (2) $C \sqsubseteq \exists R.D$

(3) $C \sqsubseteq \leqslant 1R$ (4) $C \sqsubseteq \forall R.\neg D$

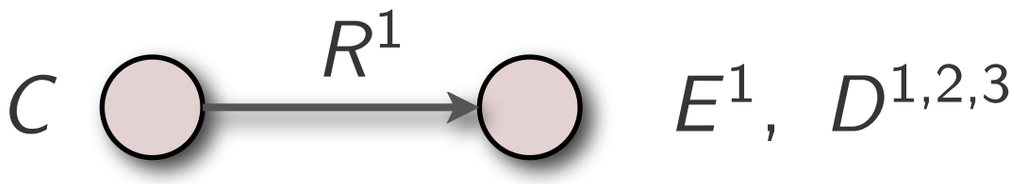


Tableau Tracing

Additional challenge 1:

Determine whether a clash depends on a choice/merge

e.g., (1) $C \sqsubseteq \exists R.E$ (2) $C \sqsubseteq \exists R.D$

(3) $C \sqsubseteq \leqslant 1 R$ (4) $C \sqsubseteq \forall R.\neg D$

Tableau Tracing

Additional challenge 1:

Determine whether a clash depends on a choice/merge

e.g., (1) $C \sqsubseteq \exists R.E$ (2) $C \sqsubseteq \exists R.D$

(3) $C \sqsubseteq \leqslant 1R$ (4) $C \sqsubseteq \forall R.\neg D$

Solution: order rules ~> delay choices; insert choice points

Tableau Tracing

Additional challenge 1:

Determine whether a clash depends on a choice/merge

e.g., (1) $C \sqsubseteq \exists R.E$ (2) $C \sqsubseteq \exists R.D$

(3) $C \sqsubseteq \leqslant 1R$ (4) $C \sqsubseteq \forall R.\neg D$

Solution: order rules ~> delay choices; insert choice points

Tableau Tracing

Additional challenge 2:

Determine which axioms are responsible for the choice (esp. due to merges)

e.g., (1) $C \sqsubseteq \exists R.D$ (2) $C \sqsubseteq \exists R.E$

(3) $C \sqsubseteq \exists R.\neg D$ (4) $C \sqsubseteq \leqslant 1R$

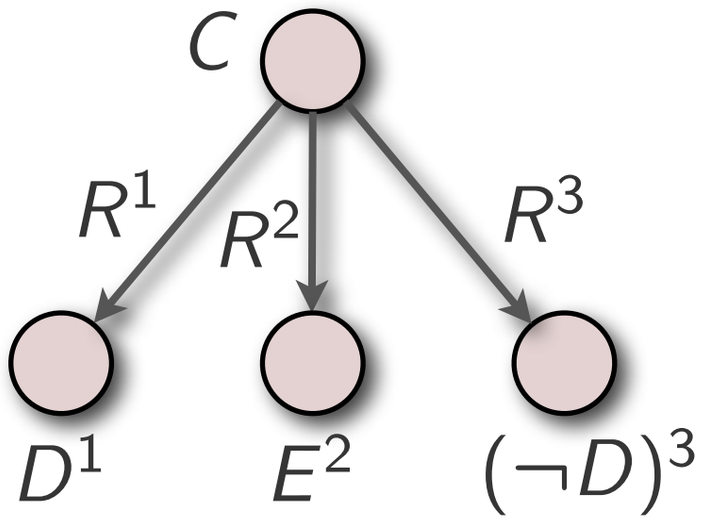


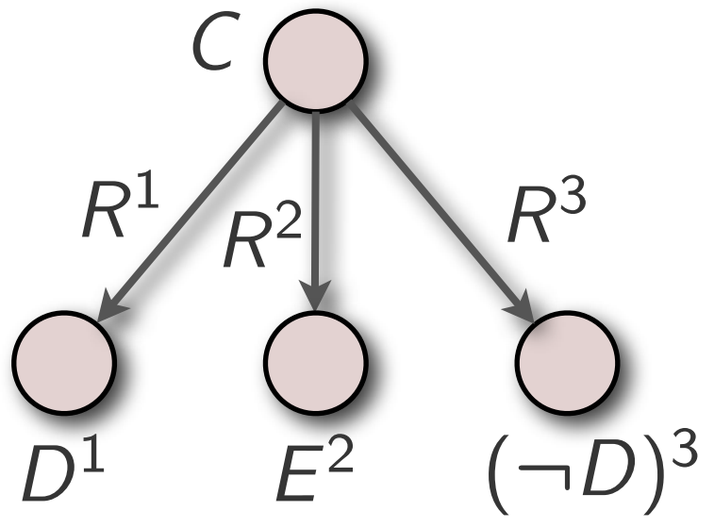
Tableau Tracing

Additional challenge 2:

Determine which axioms are responsible for the choice (esp. due to merges)

e.g., (1) $C \sqsubseteq \exists R.D$ (2) $C \sqsubseteq \exists R.E$

(3) $C \sqsubseteq \exists R.\neg D$ (4) $C \sqsubseteq \leqslant 1R$

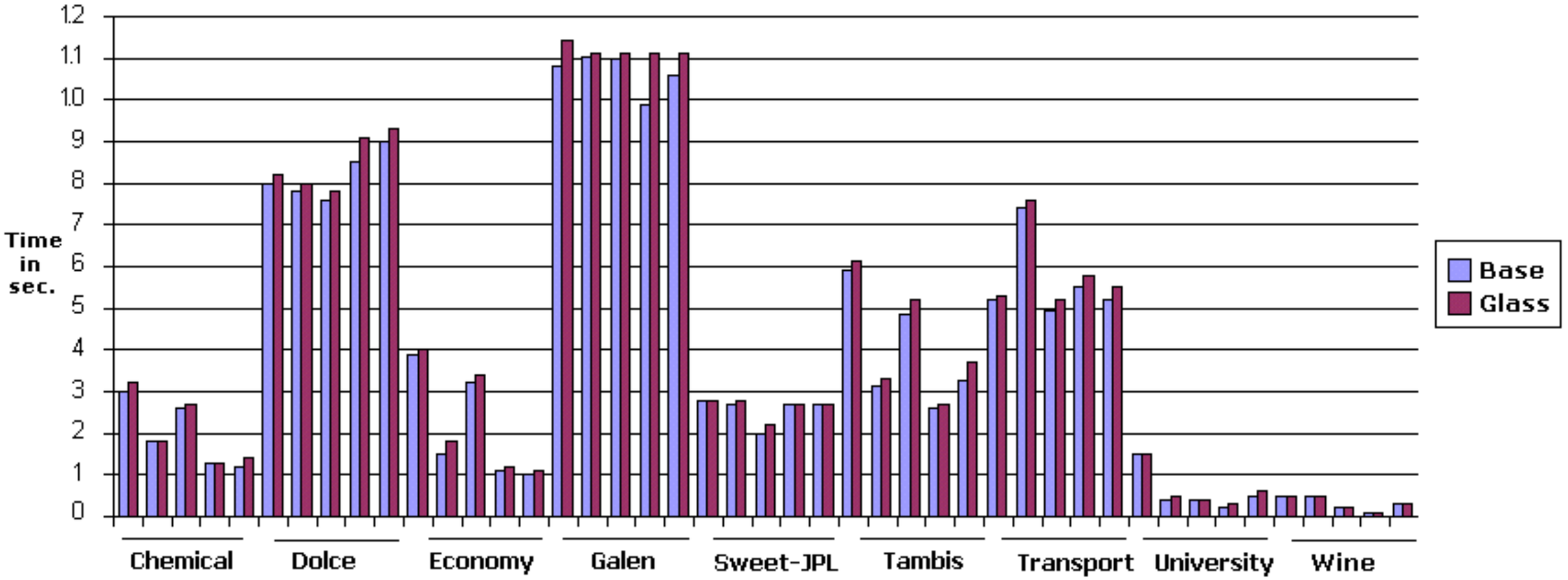


Solution: record choice context (cardinality, edges)

Tracing for One

- Pros
  1. "Side effect" -- fast/always on
  2. Works when the reasoner does
- Cons
  1. Still non-trivial, reasoner and procedure dependent
     - E.g., no exact tracing for OWL
     - Not always faster!
  2. Even when faster, doesn't dominate
  3. Reasoners get faster over time
     - Or are better for different problems

Glass Box Evaluation



Often, but not always, negligible

Tracing for All

- Radical change to algorithms
  - Termination problems
    - ALC + General Tboxes (maybe)
    - EL++ (recently)
  - Optimizations turned off
    - Total re-engineering
    - Non-determinism even harder!
- Complicates already complex implementations
  - For no obvious benefit
  - Pure black box works!

Black-box

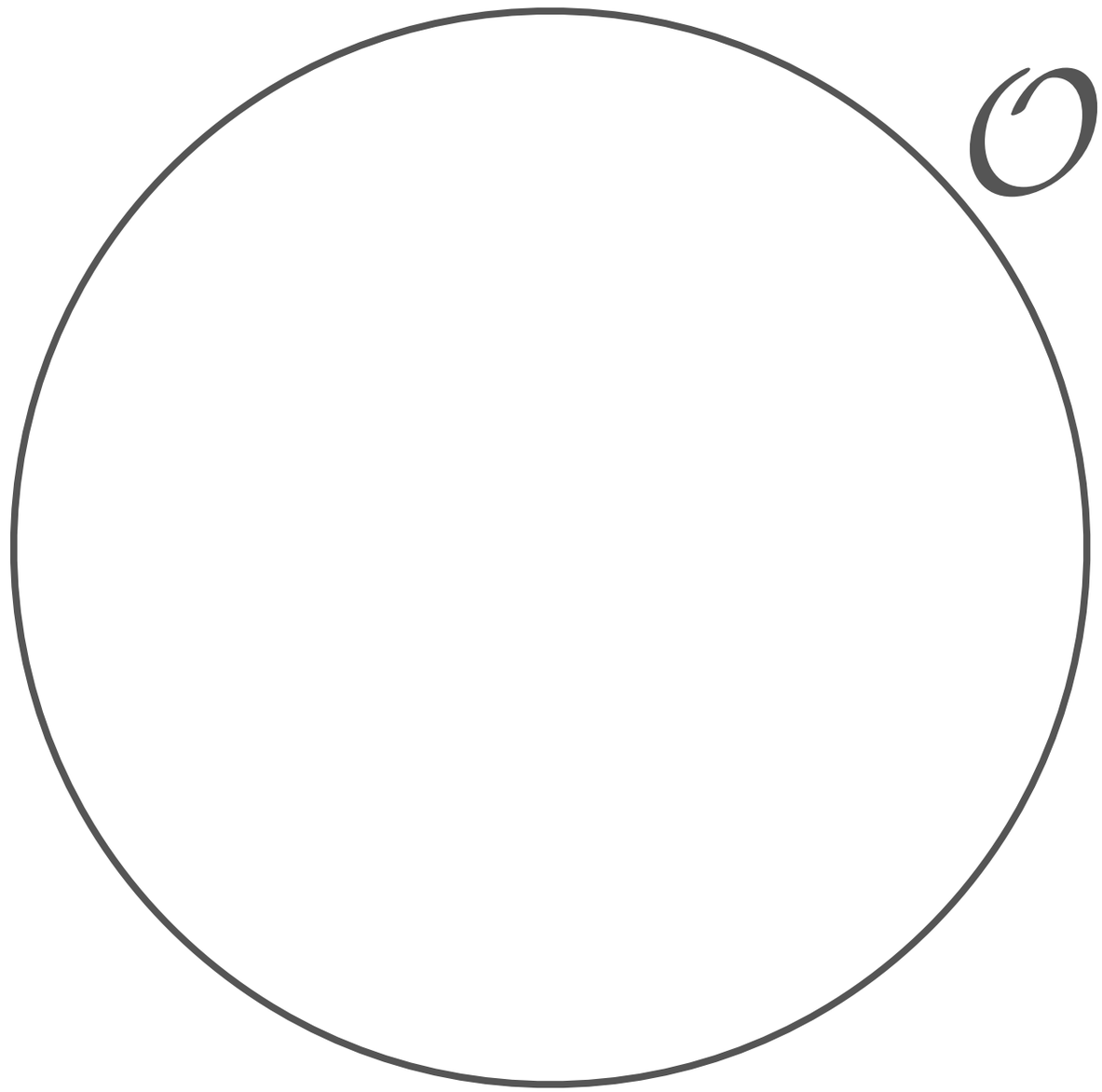
- Does not depend on a particular reasoner
- All that we require is that we can ask the reasoner whether a class expression is satisfiable - i.e. satisfiability checking

Black-box

- Typically uses an expand-contract strategy
  - Create an empty ontology
  - Expand until expression is unsatisfiable
    - Prune until the expression is satisfiable
- Various heuristics for each
  - Expand: e.g., syntactically related first, modules
  - Contract: e.g., windows, divide and conquer

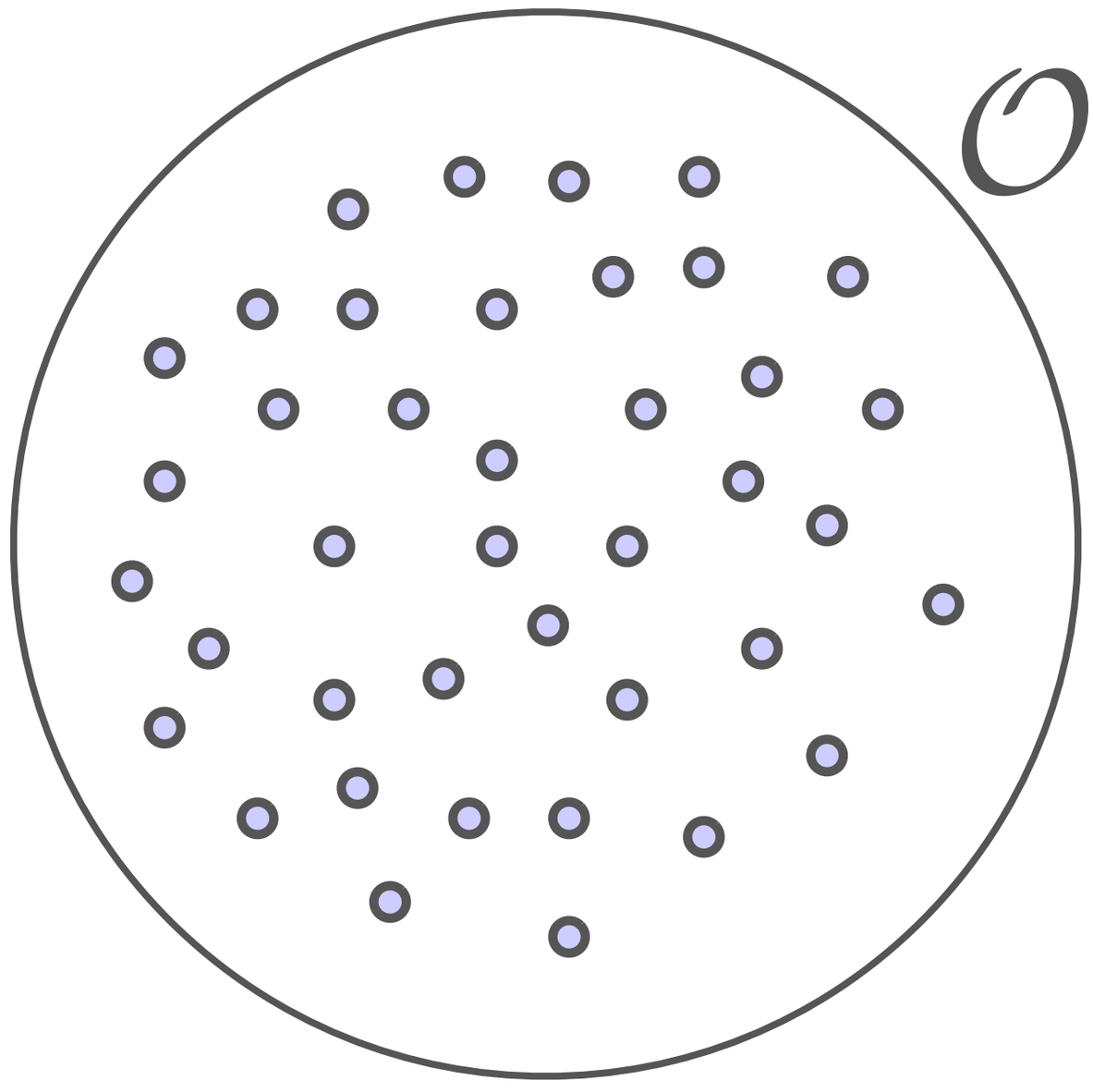
Black-Box

$C \sqsubseteq D$



Black-Box

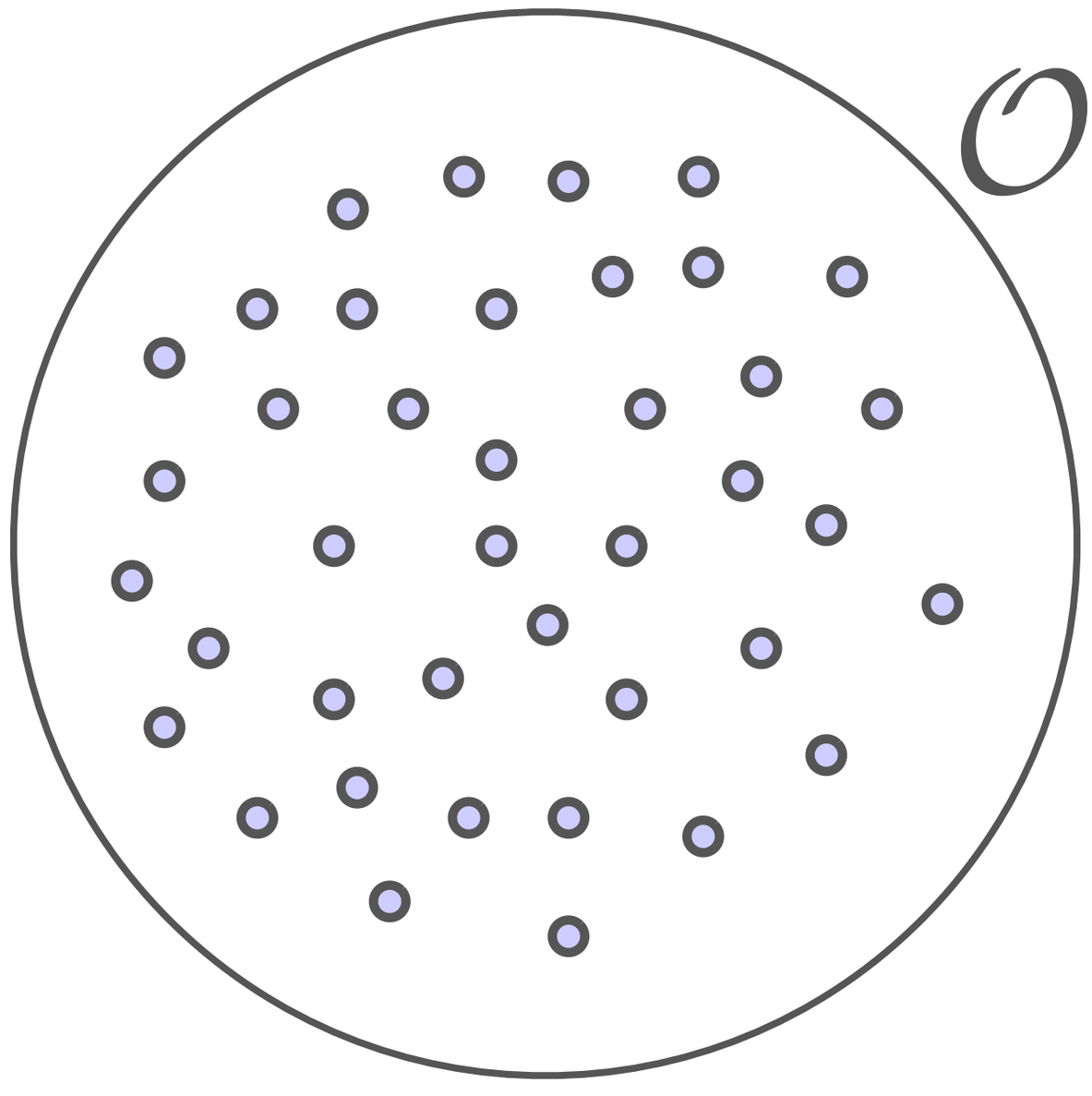
$C \sqsubseteq D$



Black-Box

$C \sqsubseteq D$

$C \sqcap \neg D$

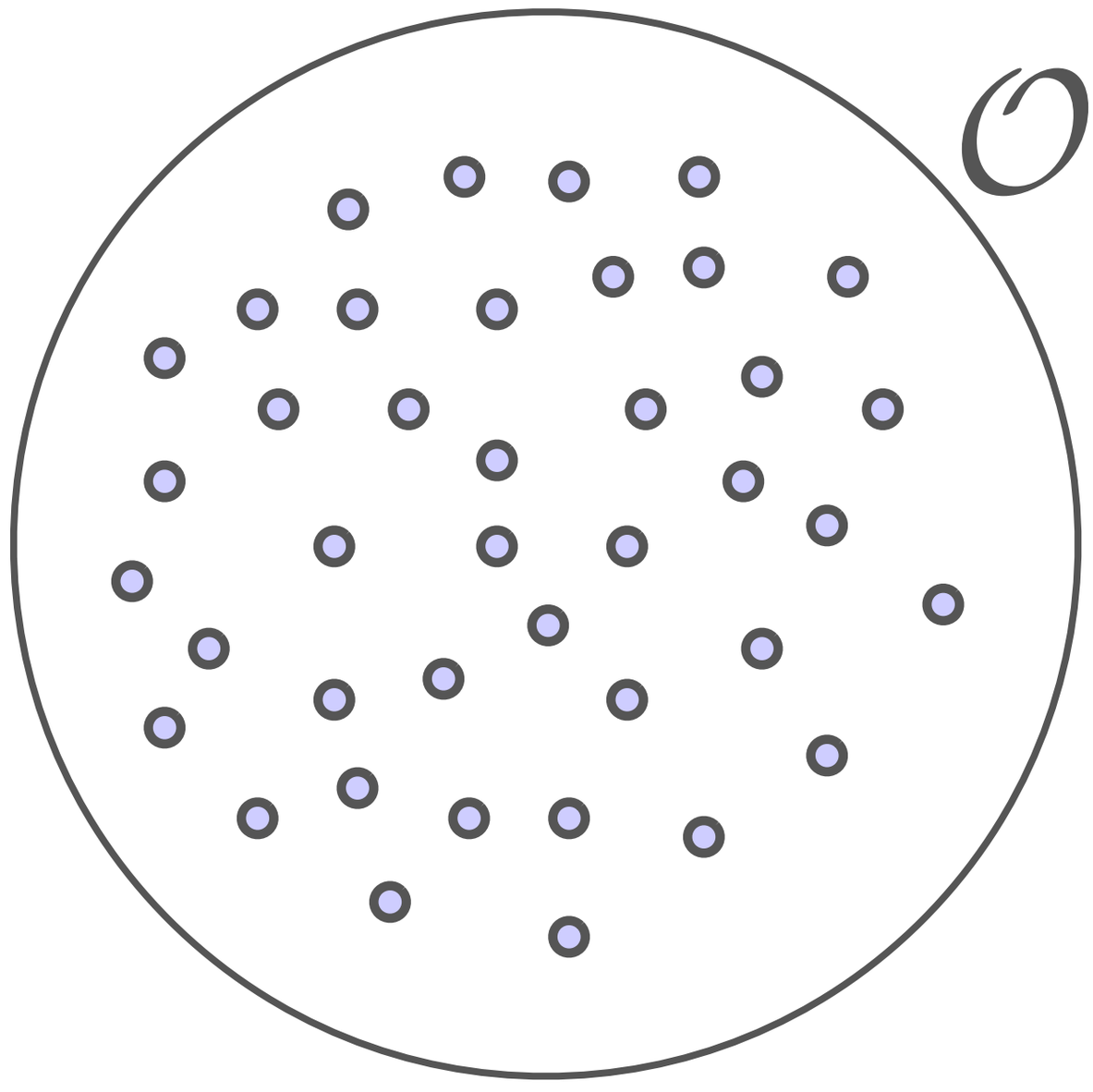


Black-Box

$C \sqsubseteq D$

$C \sqcap \neg D$

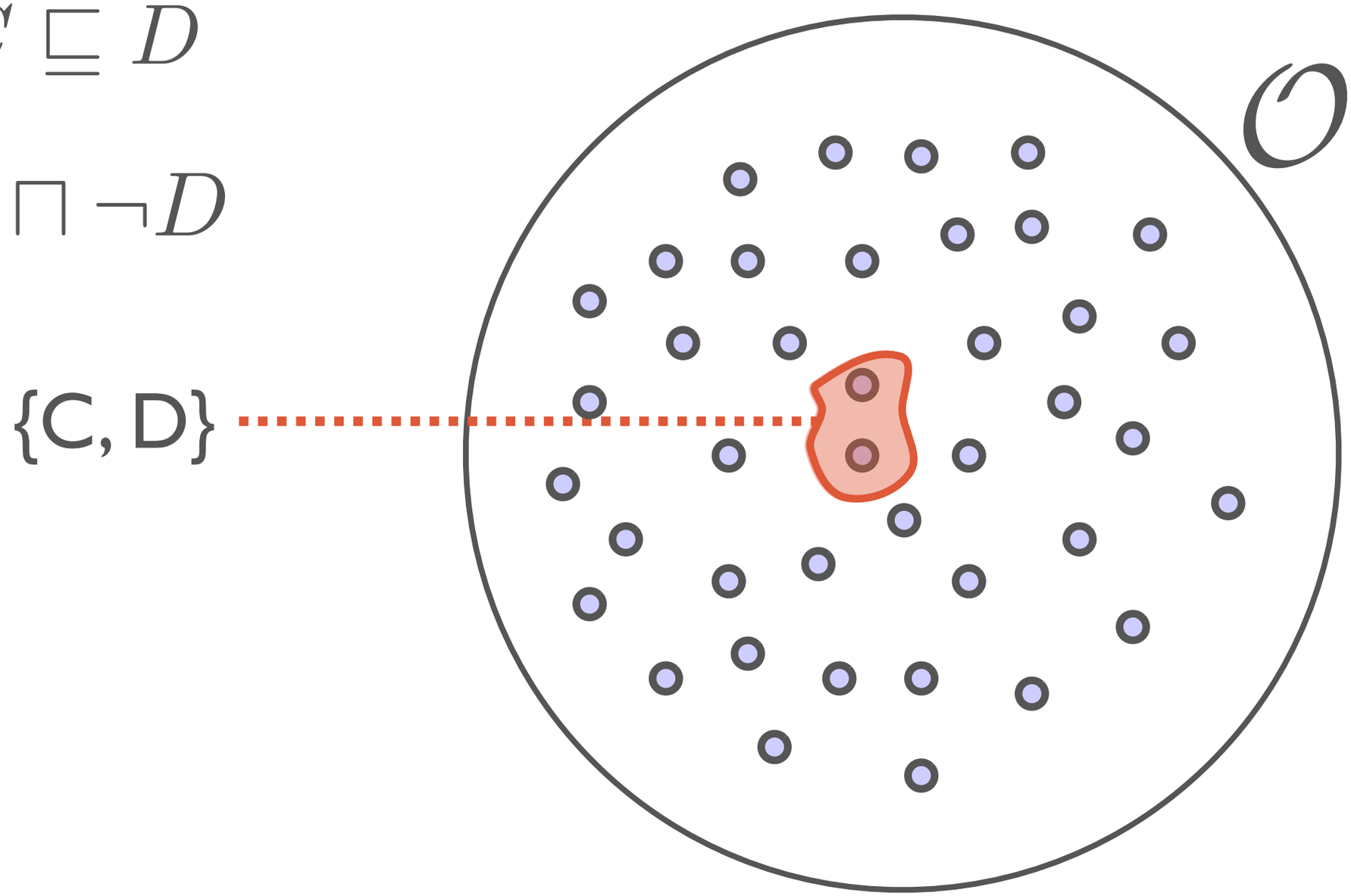
{C, D}



Black-Box

$C \sqsubseteq D$

$C \sqcap \neg D$

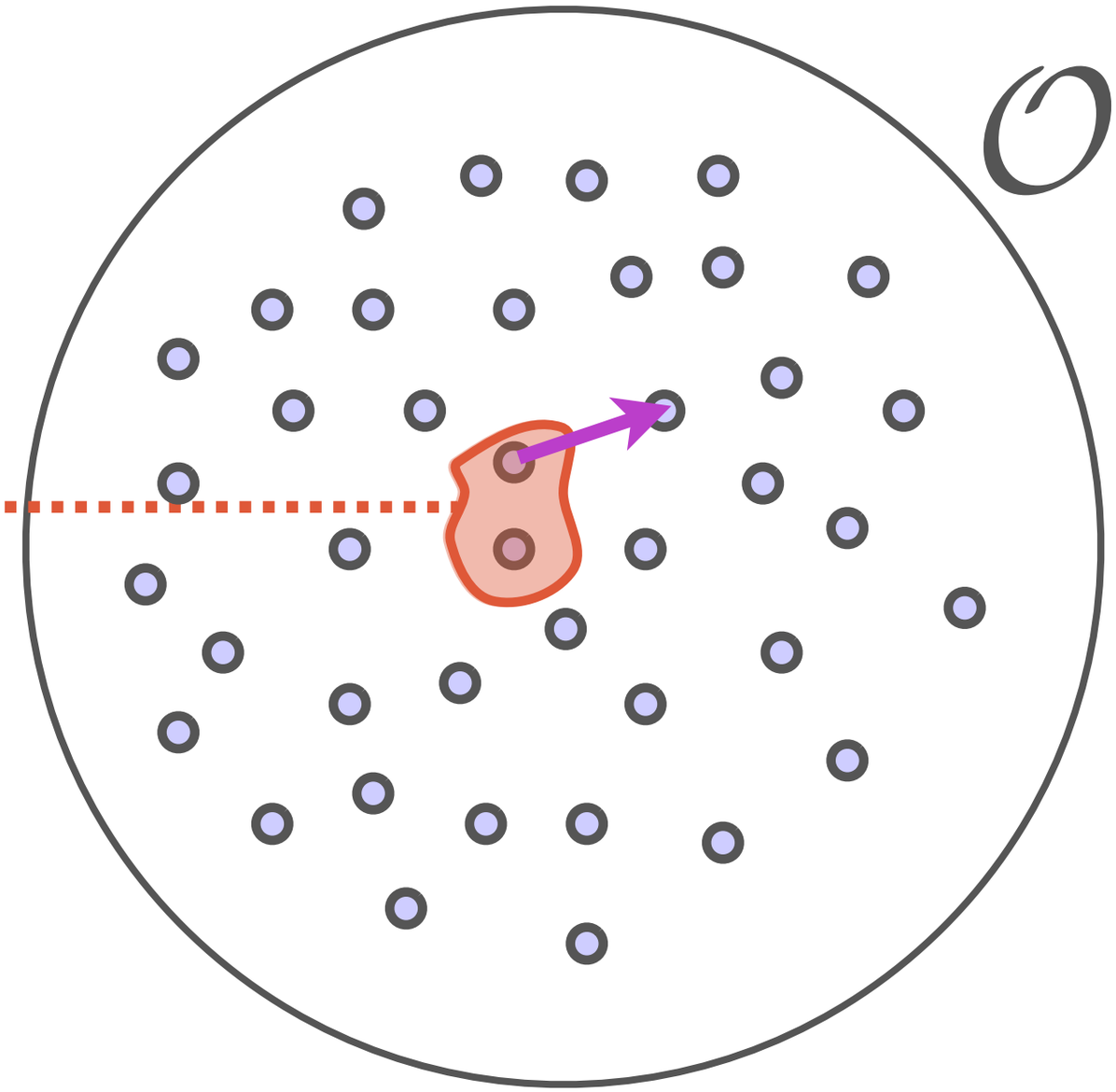


Black-Box

$C \sqsubseteq D$

$C \sqcap \neg D$

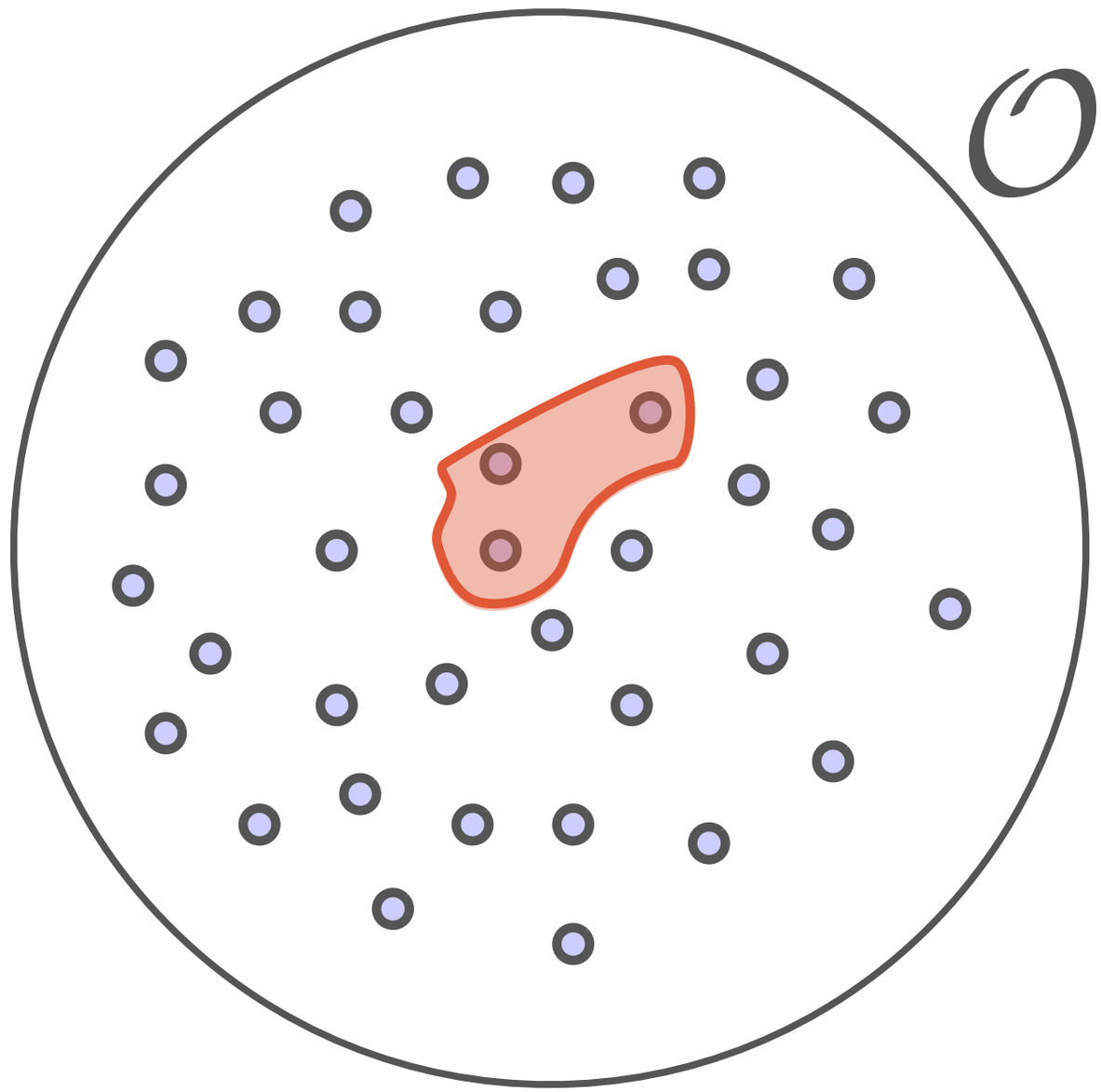
{C, D}



Black-Box

$C \sqsubseteq D$

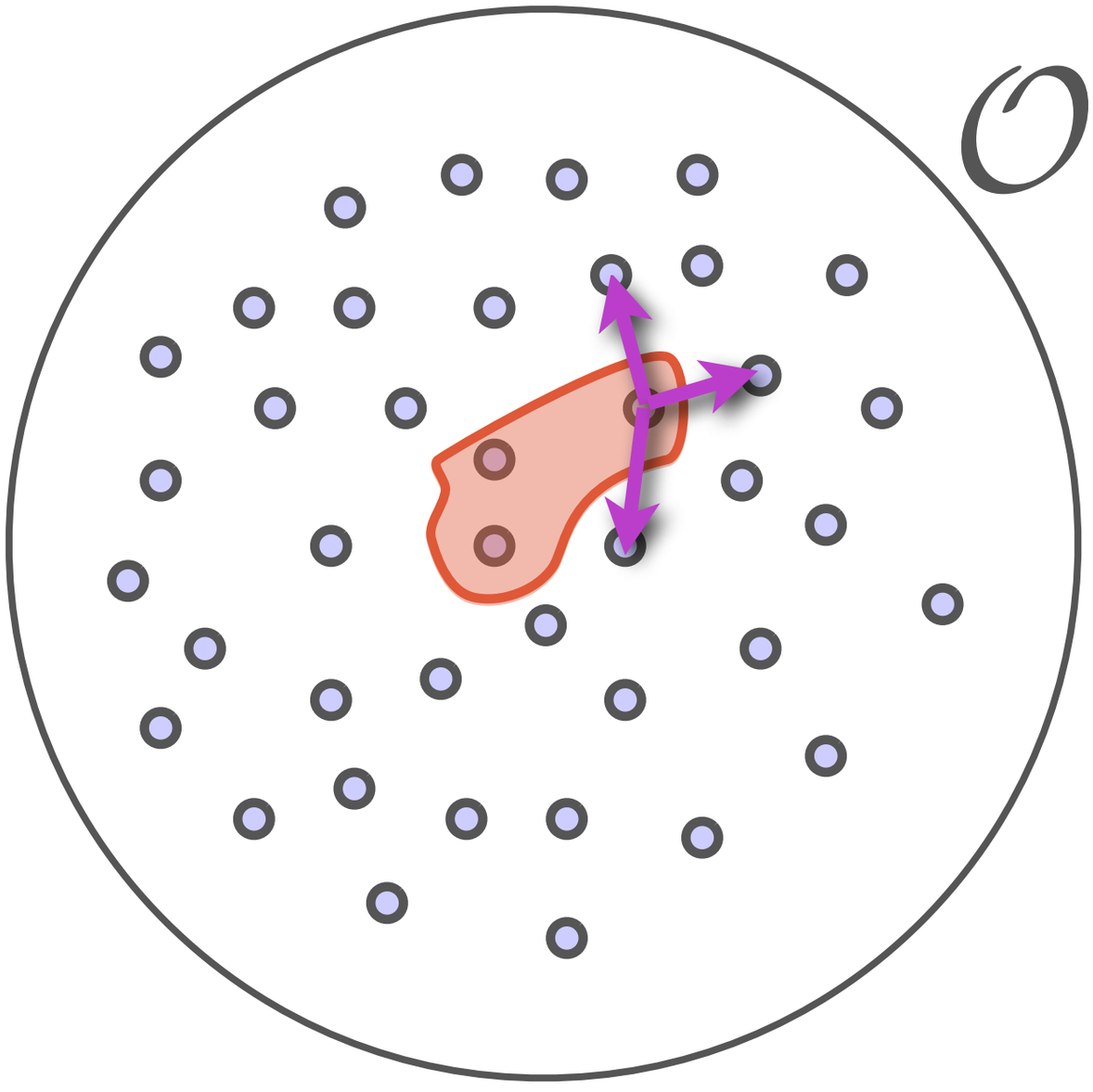
$C \sqcap \neg D$



Black-Box

$C \sqsubseteq D$

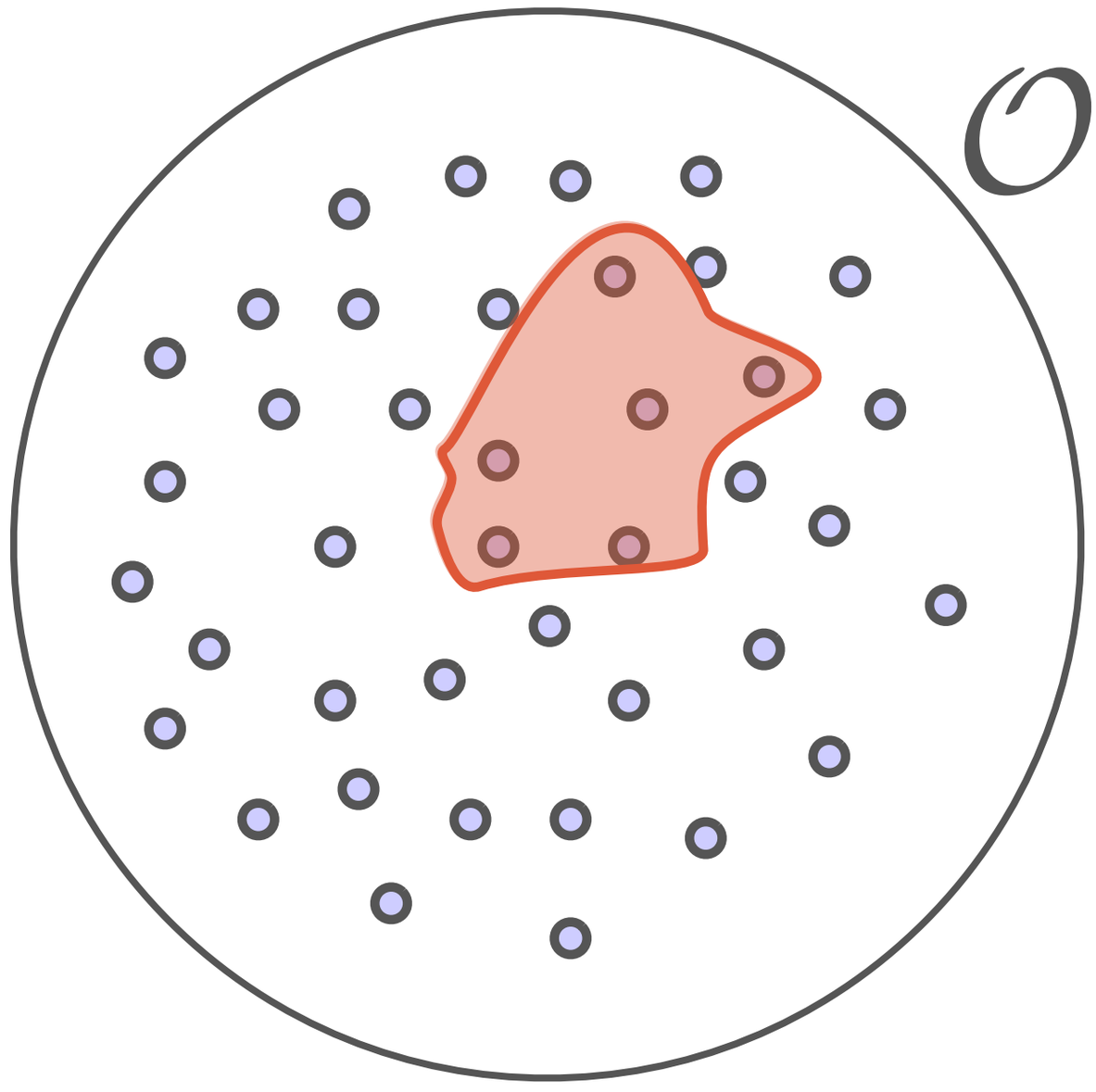
$C \sqcap \neg D$



Black-Box

$C \sqsubseteq D$

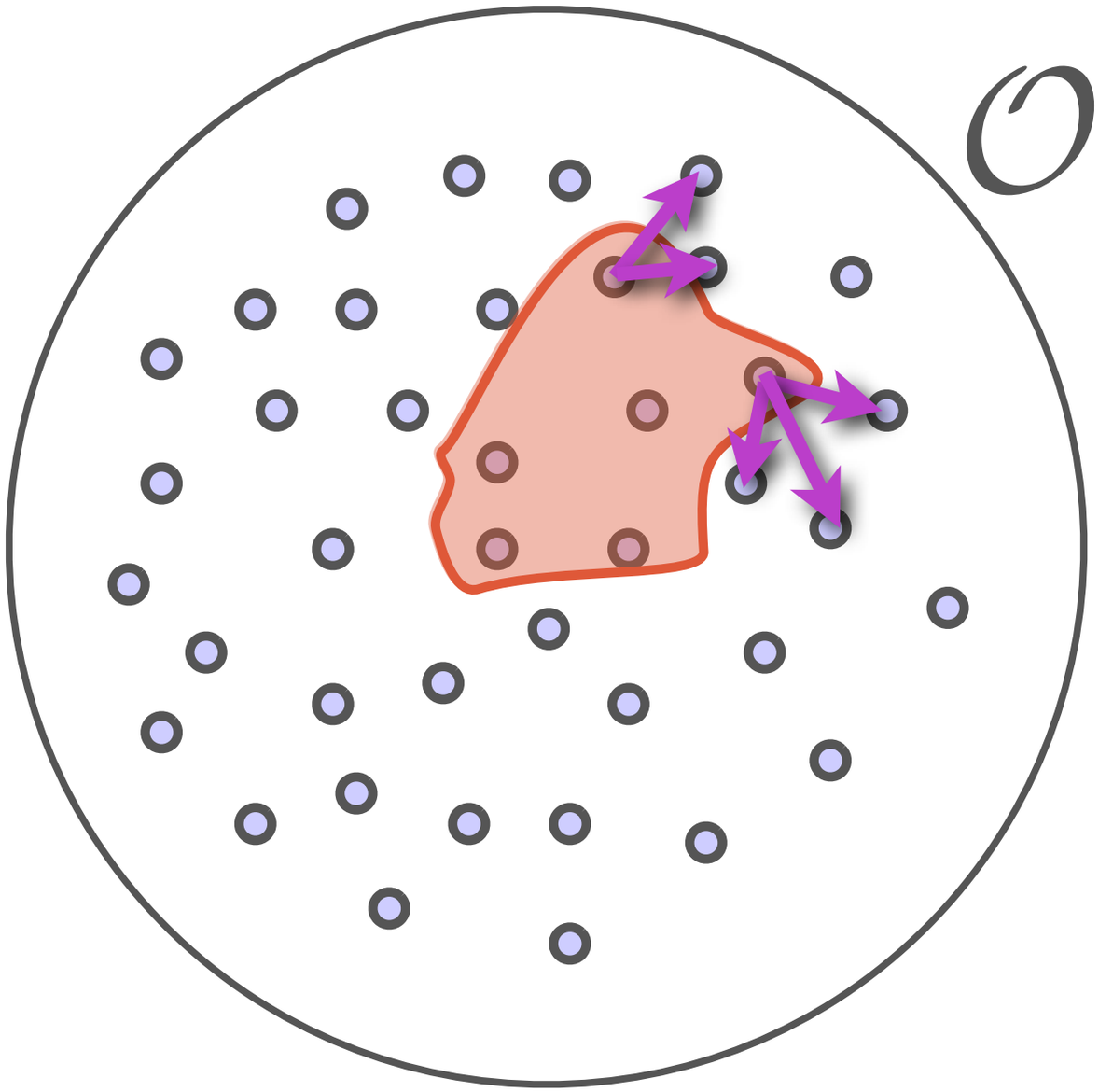
$C \sqcap \neg D$



Black-Box

$C \sqsubseteq D$

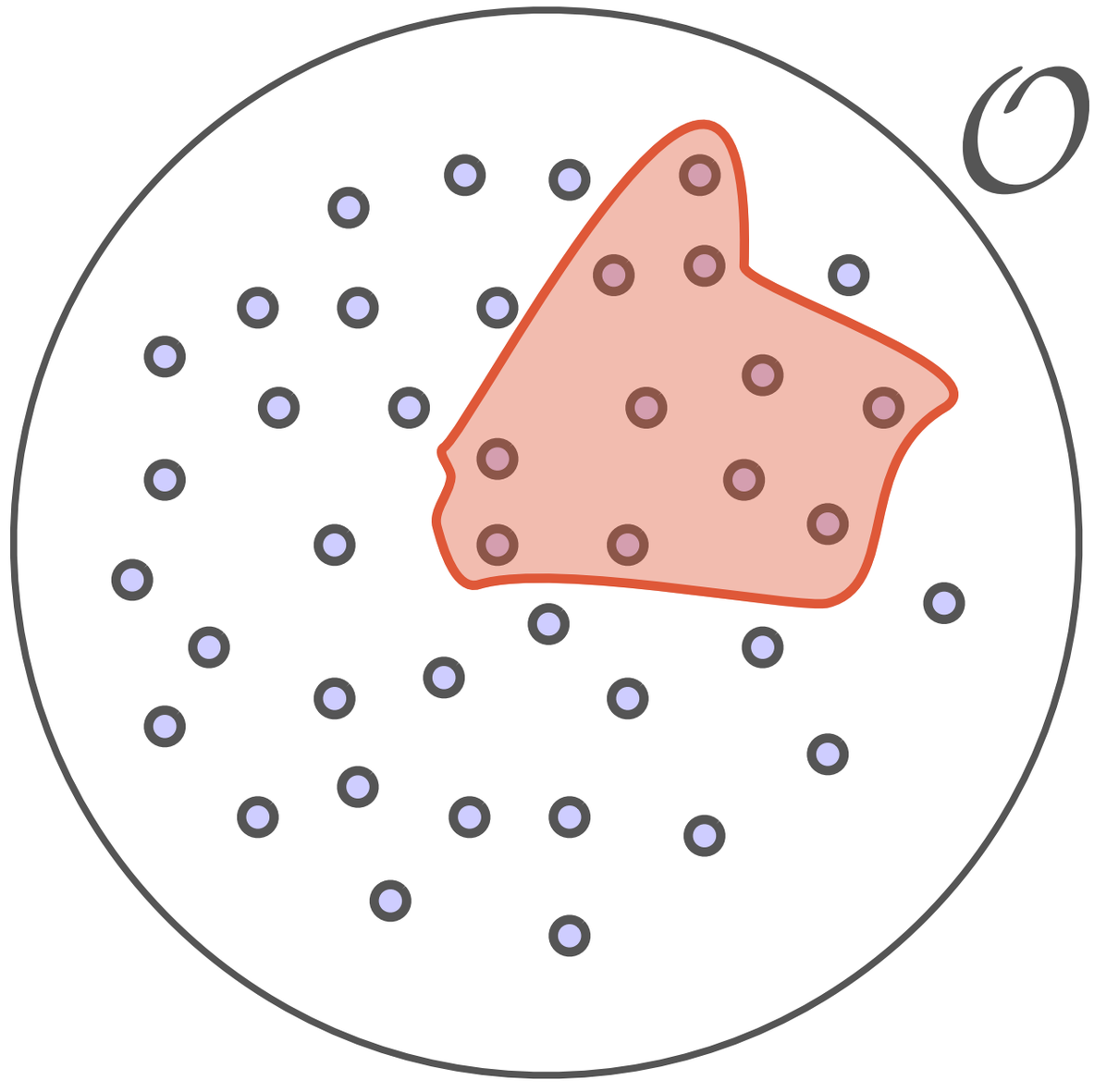
$C \sqcap \neg D$



Black-Box

$C \sqsubseteq D$

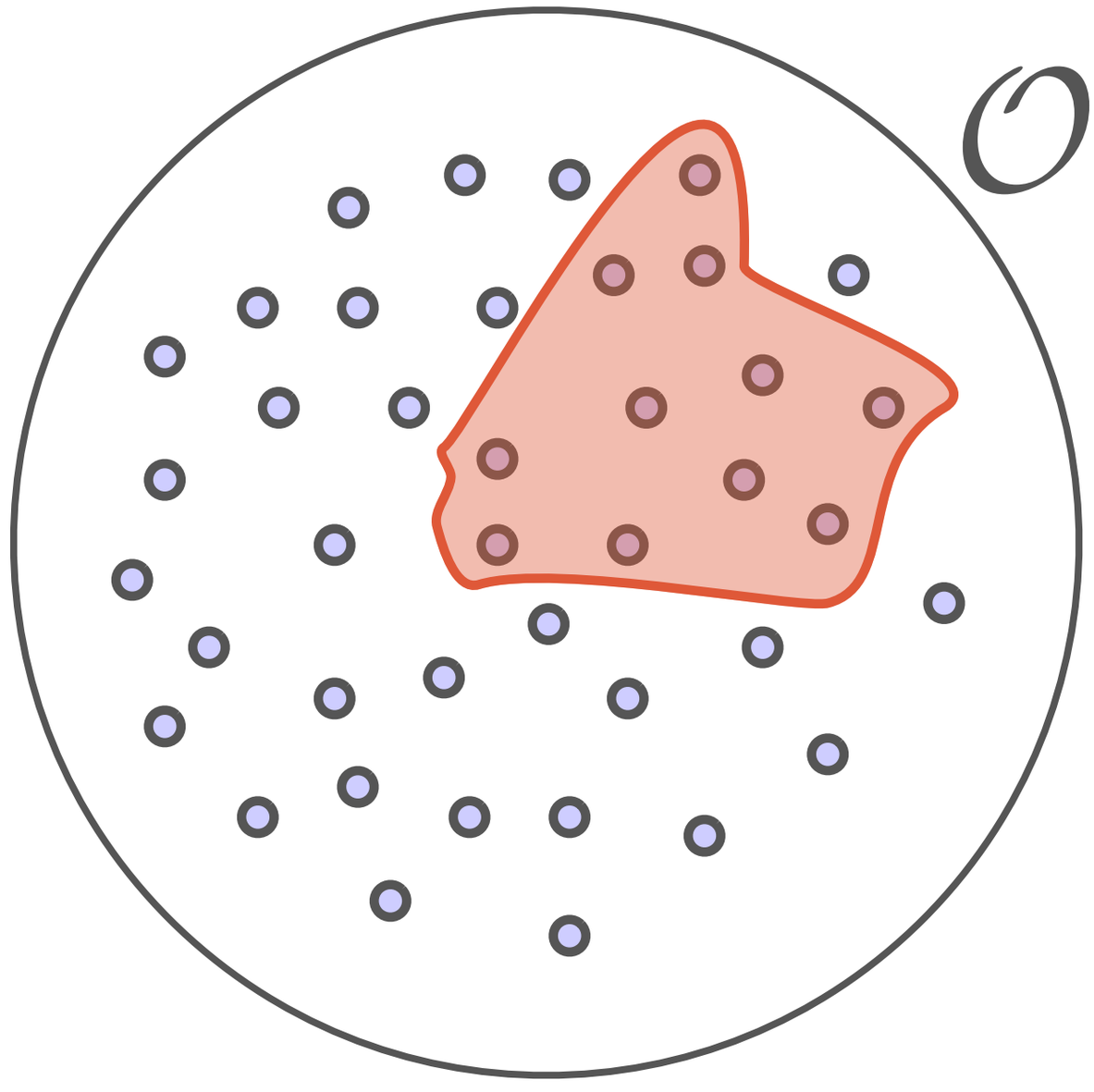
$C \sqcap \neg D$



Black-Box

$$C \sqsubseteq D$$

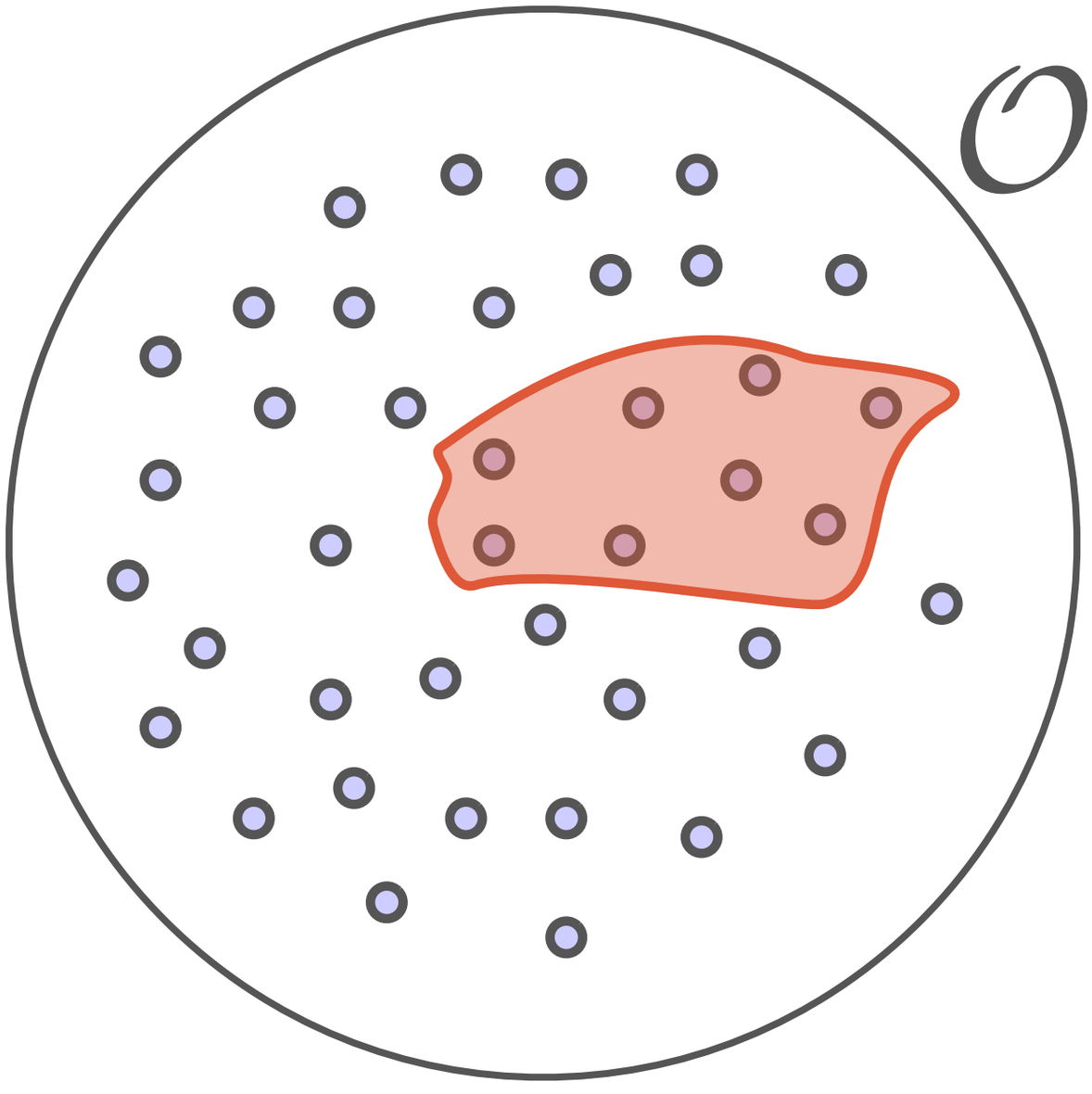
$$C \sqcap \neg D$$



Black-Box

$C \sqsubseteq D$

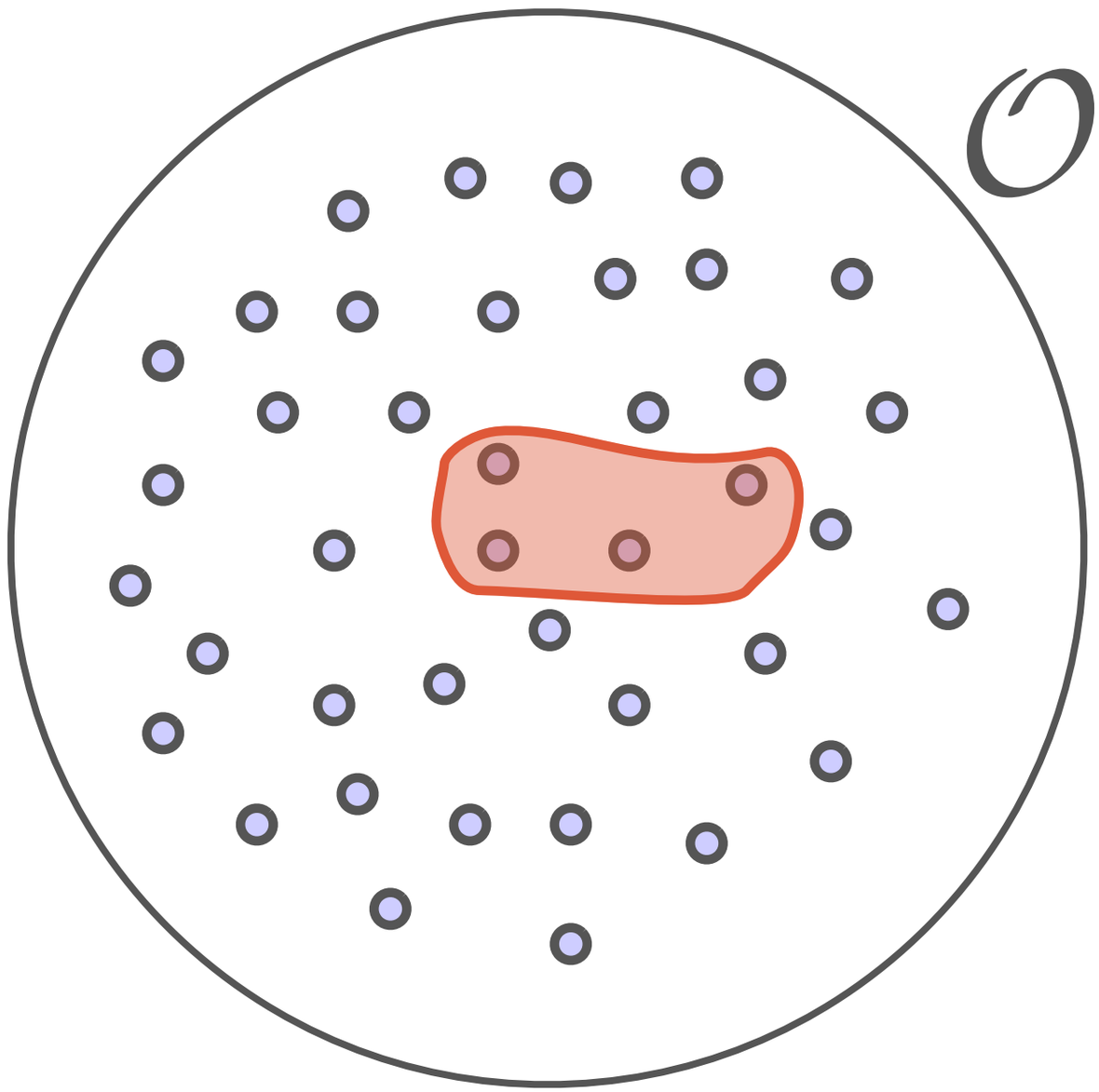
$C \sqcap \neg D$



Black-Box

$C \sqsubseteq D$

$C \sqcap \neg D$

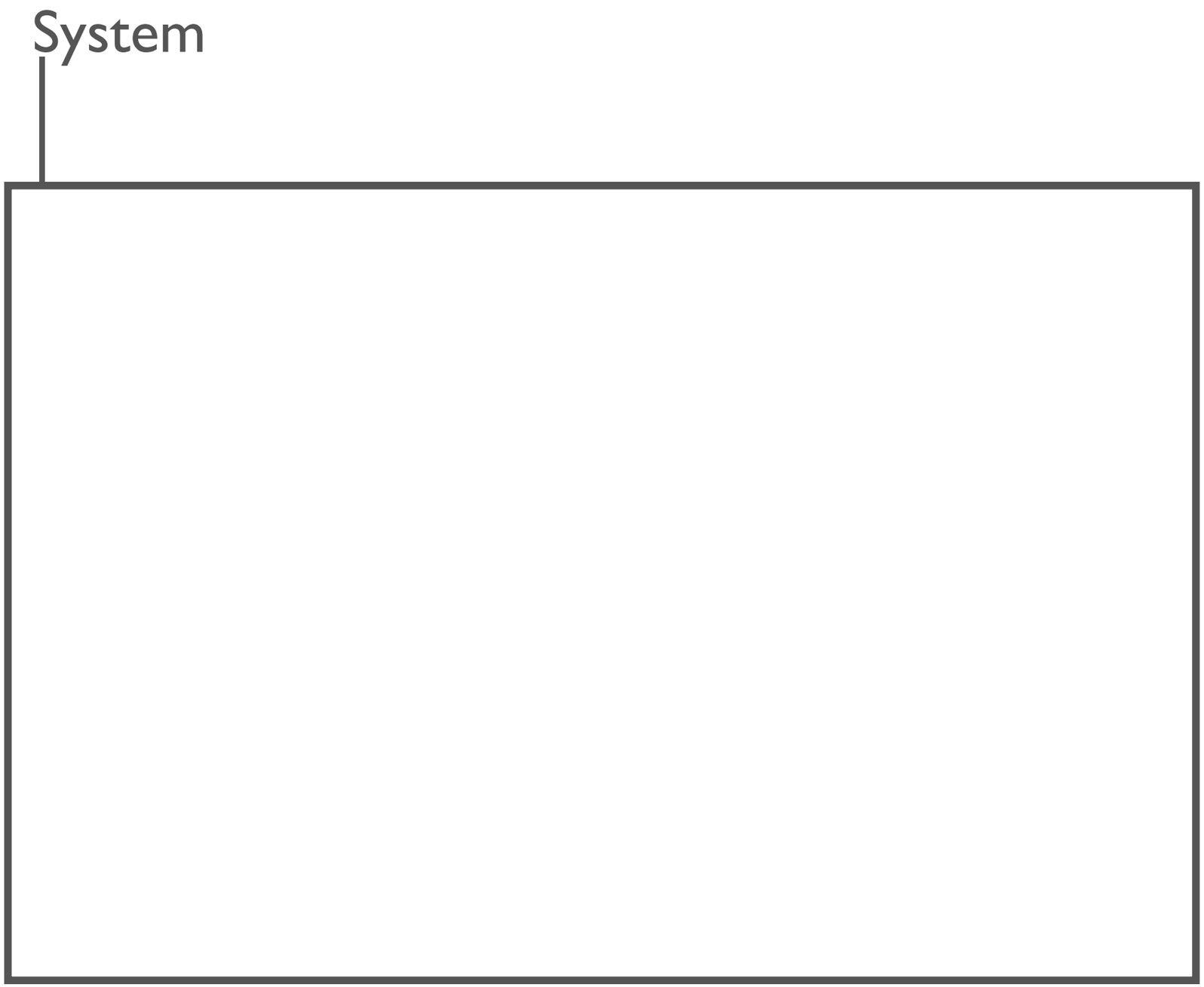


Getting the Rest

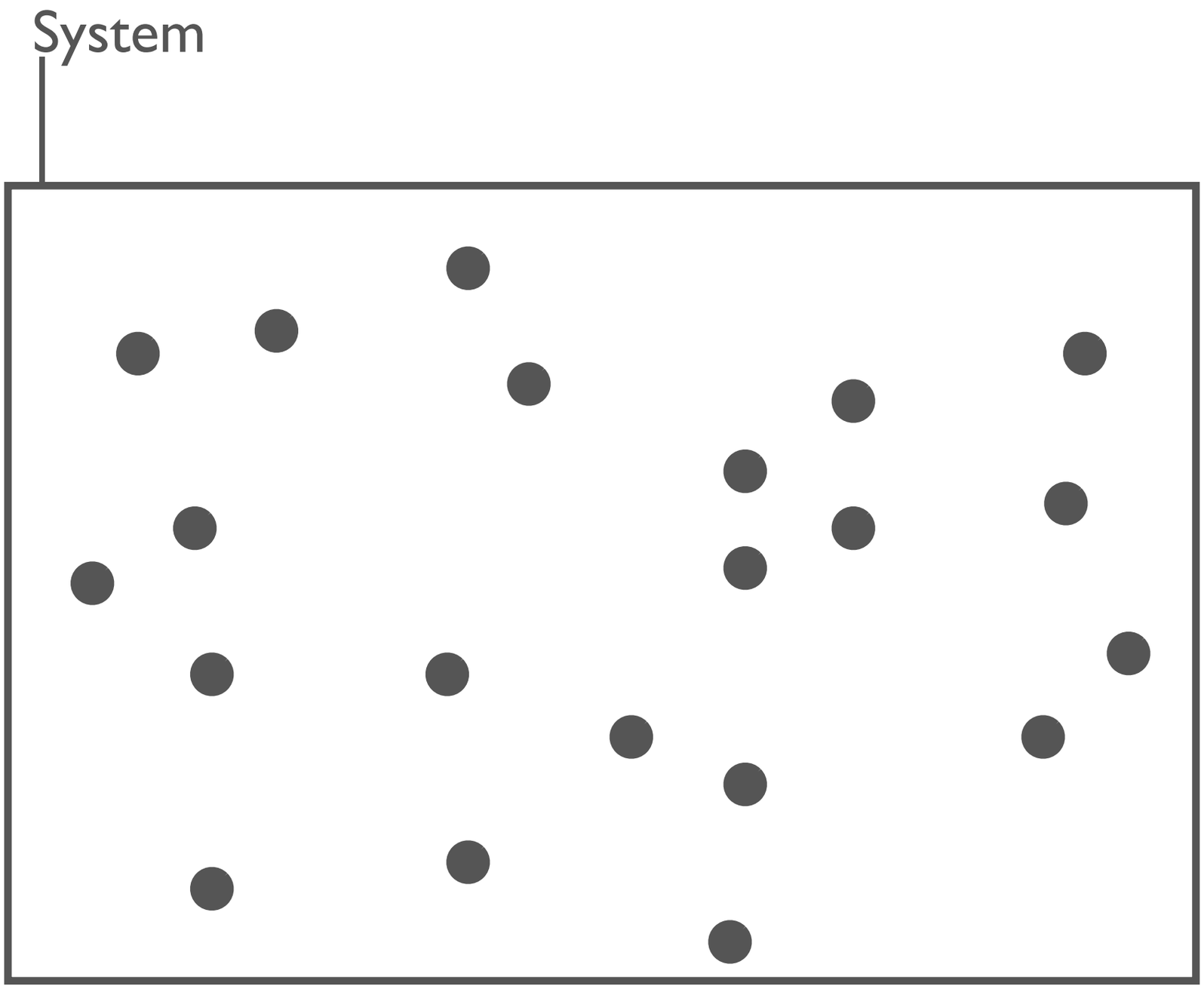
- Fairly straight-forward use of HST (Reiter)
  - Given a set of conflict sets
    - Systematically build minimal hitting sets
- We reverse this
  - Generate the conflict sets dynamically
  - Hitting sets guide our search

Reiter's Algorithm

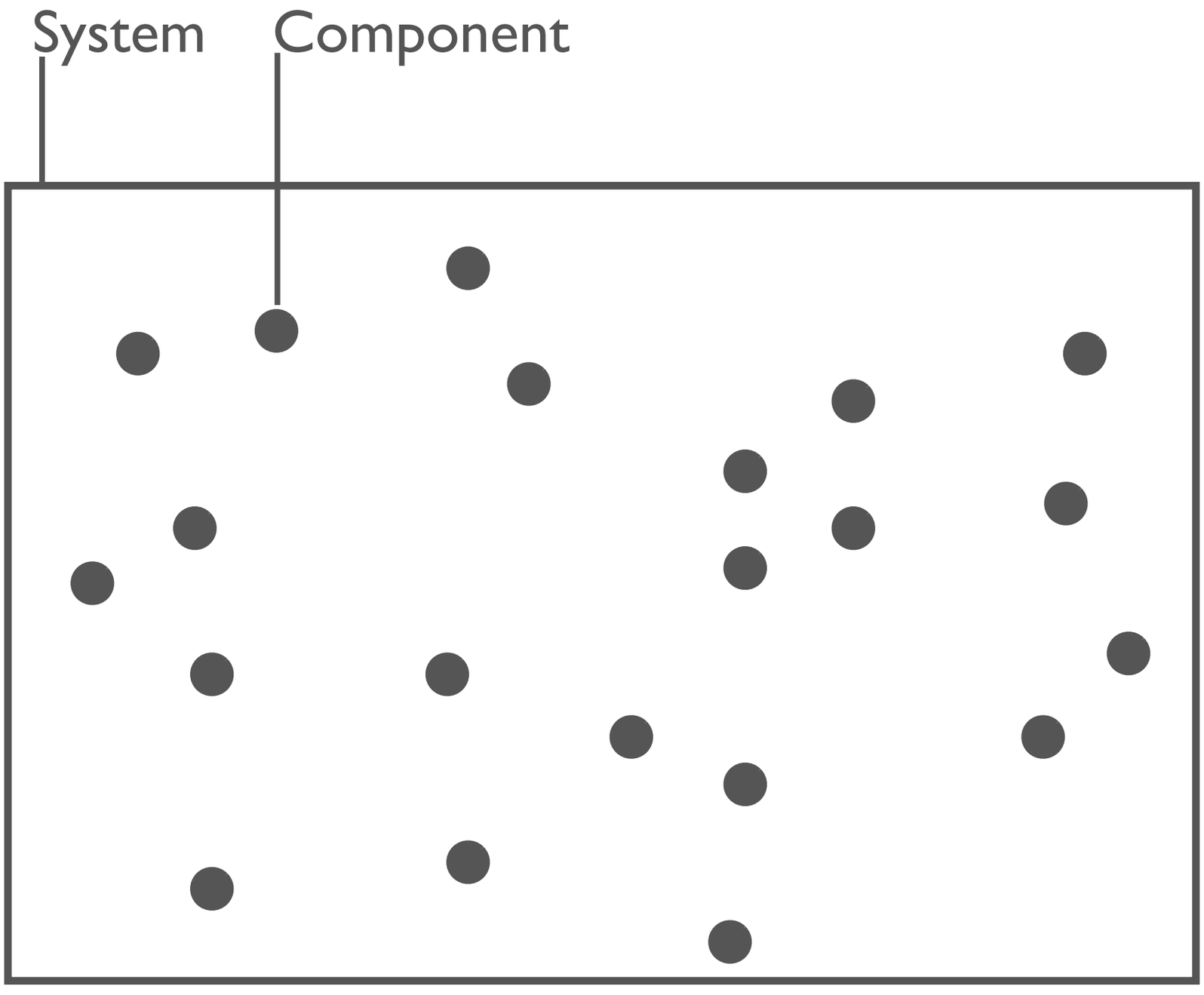
Reiter's Algorithm



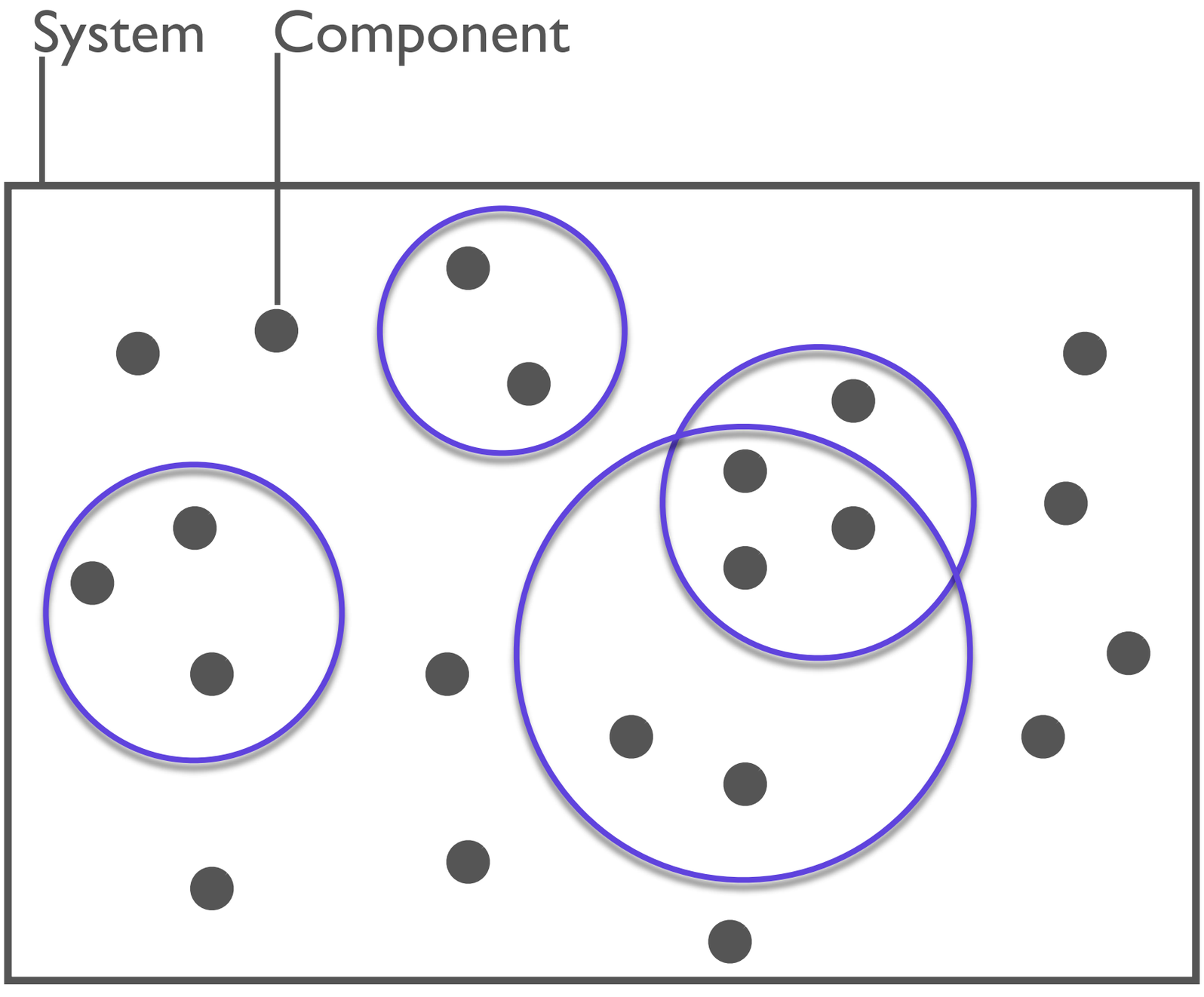
Reiter's Algorithm



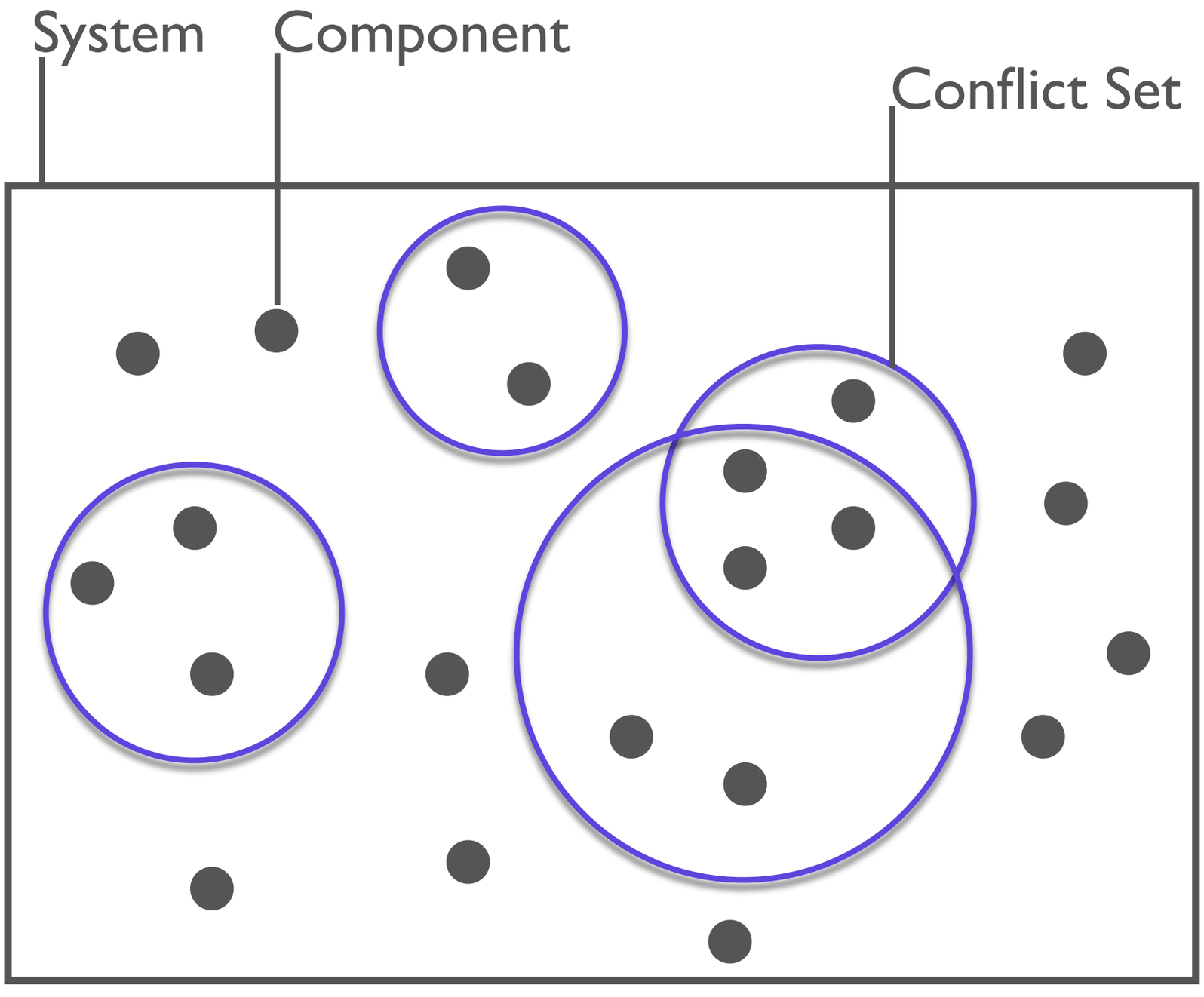
Reiter's Algorithm



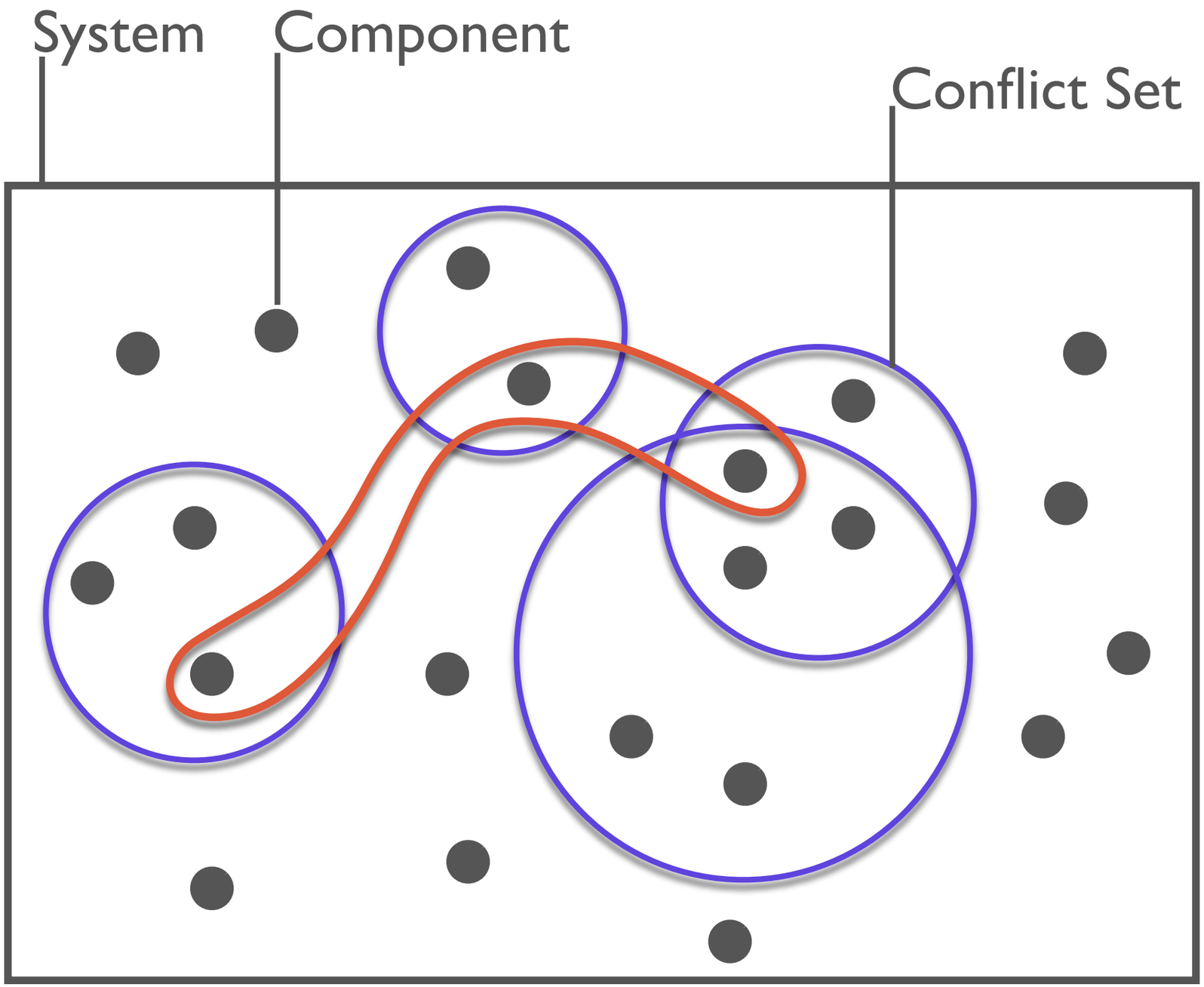
Reiter's Algorithm



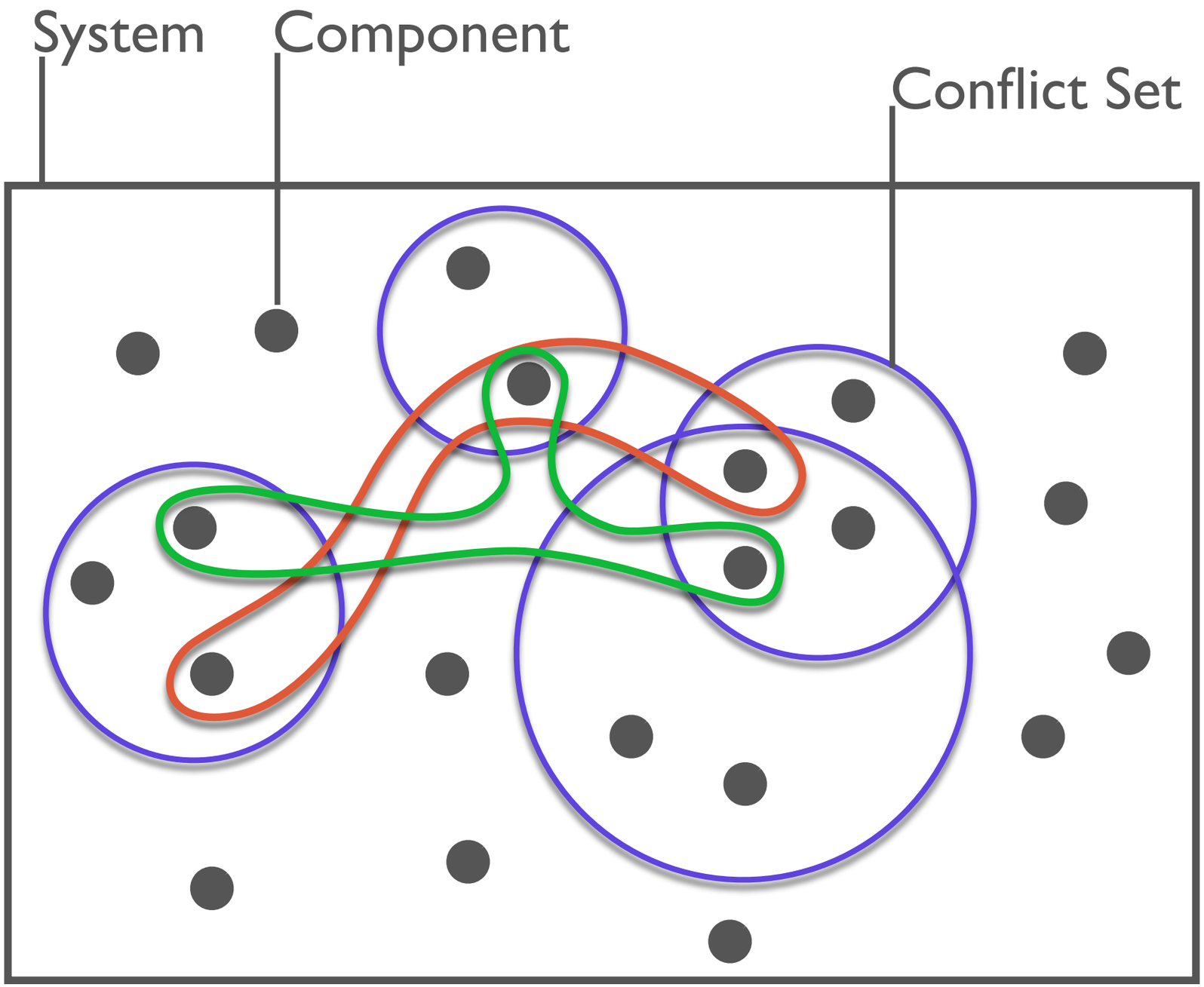
Reiter's Algorithm



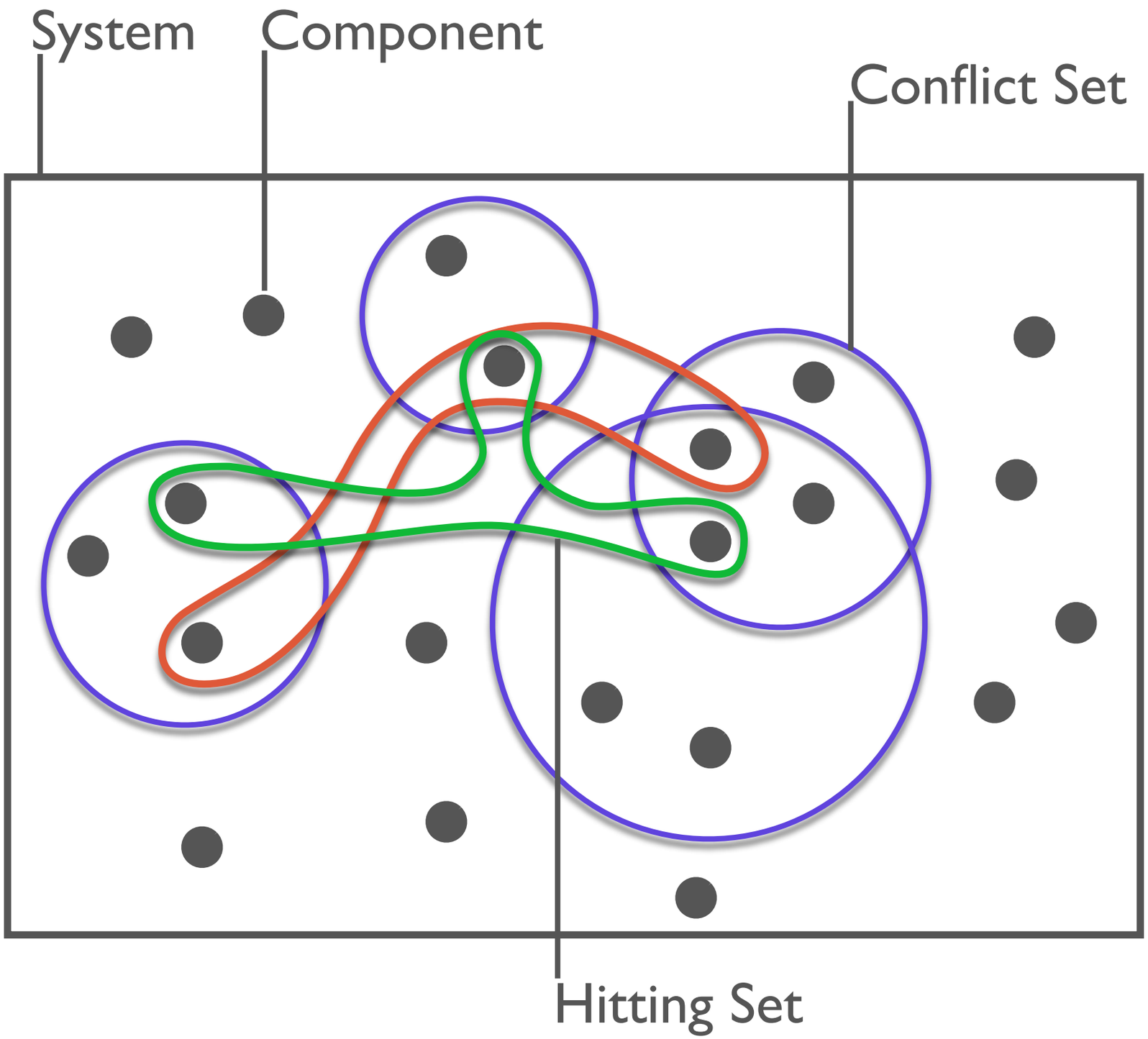
Reiter's Algorithm



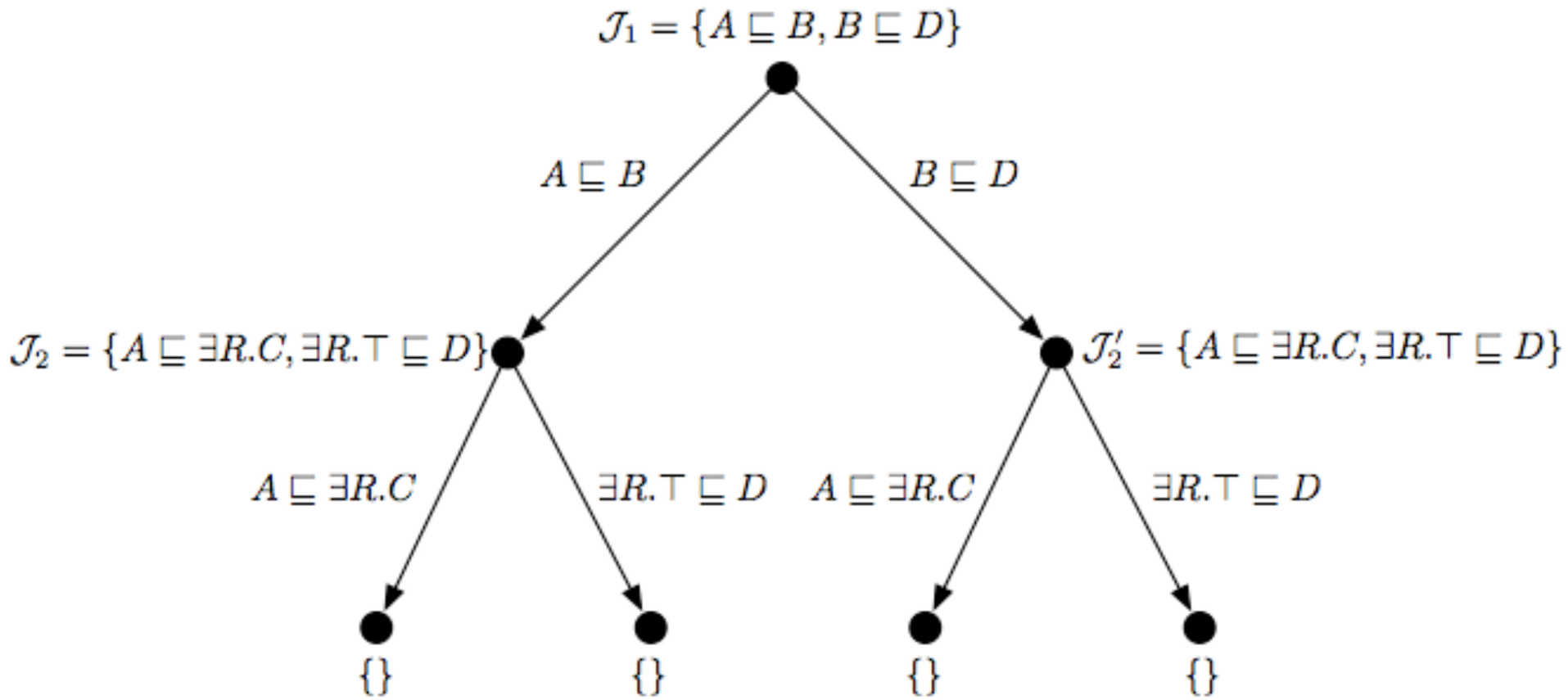
Reiter's Algorithm



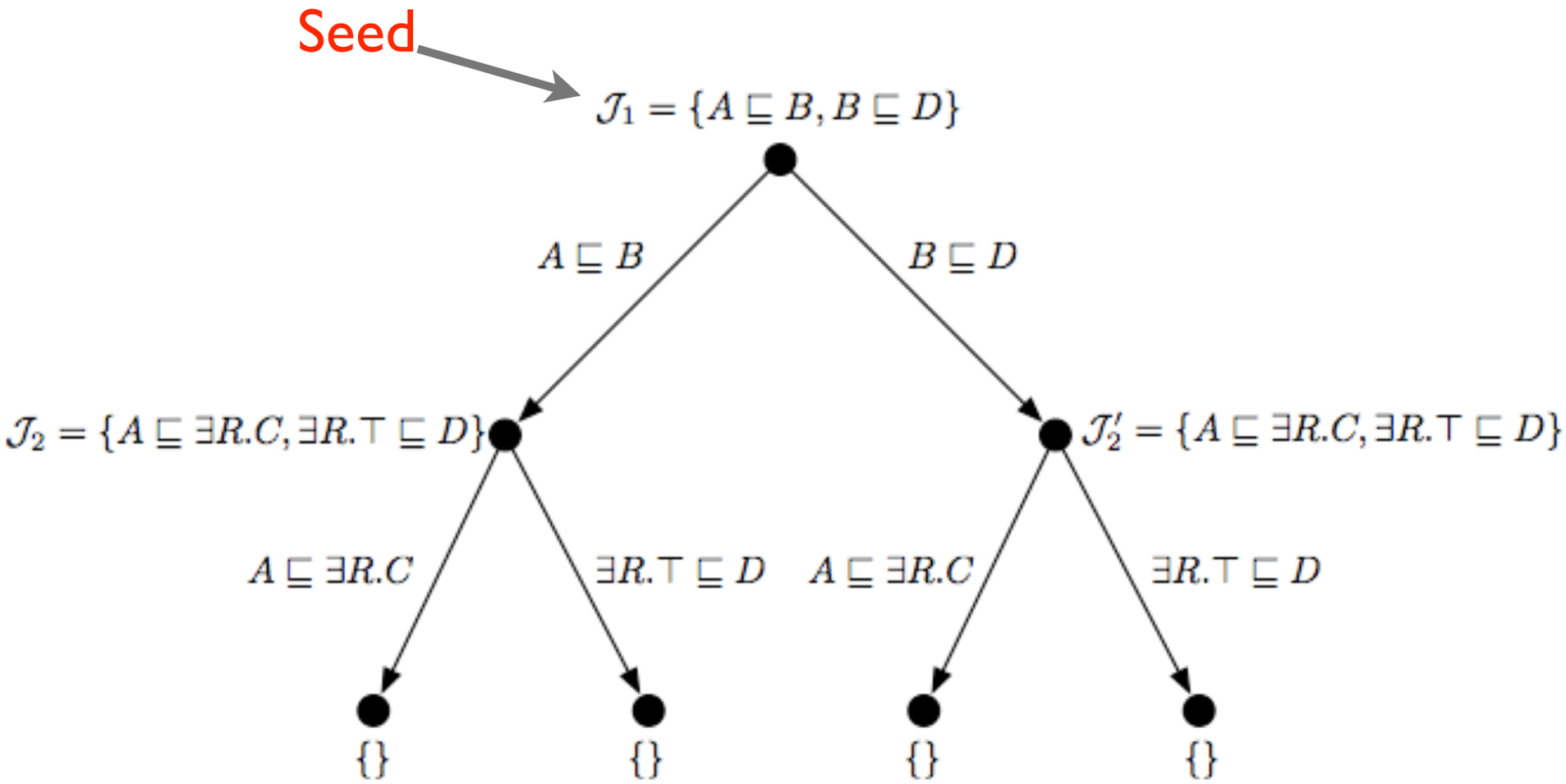
Reiter's Algorithm



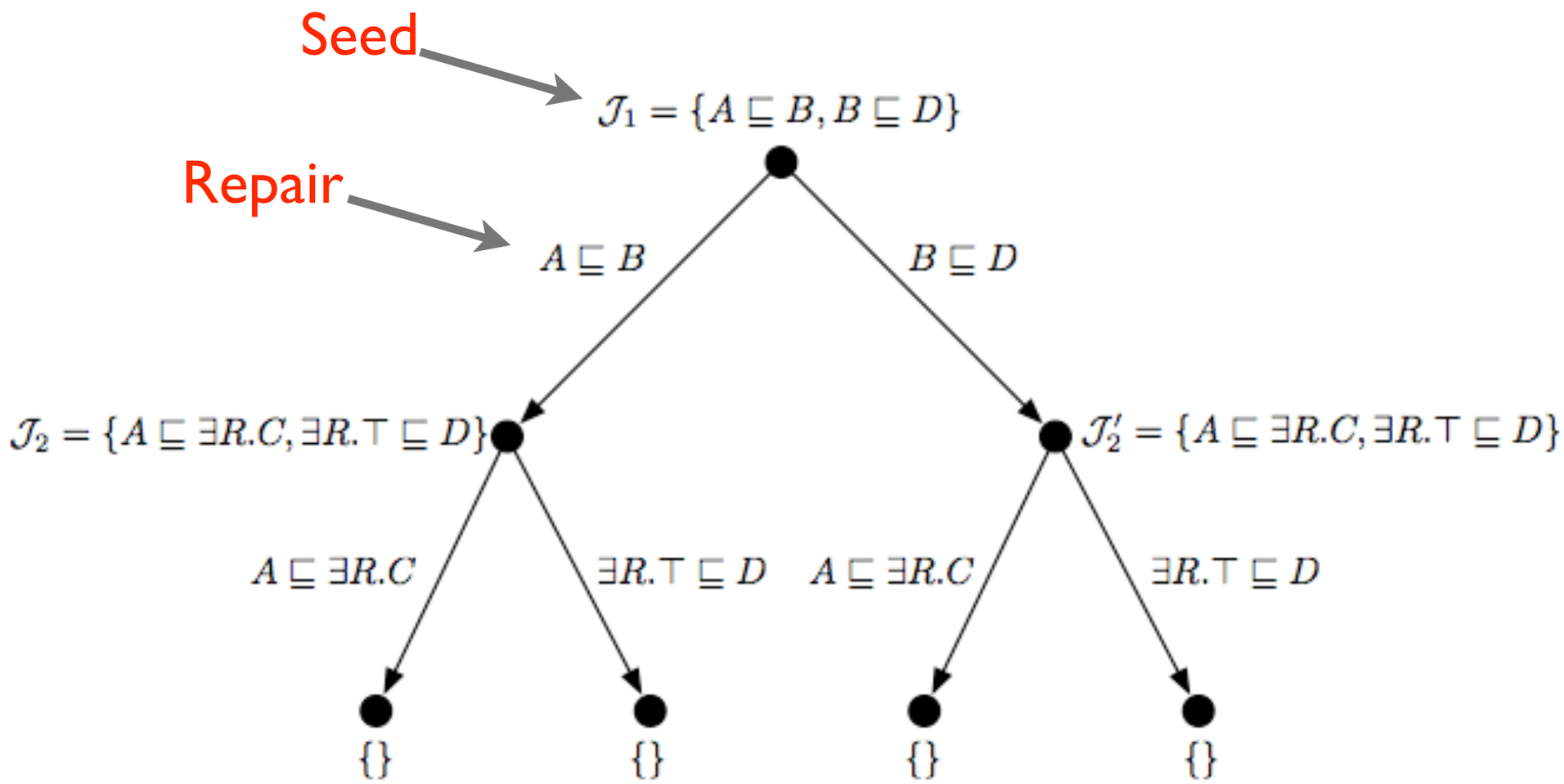
Dynamically find CSs



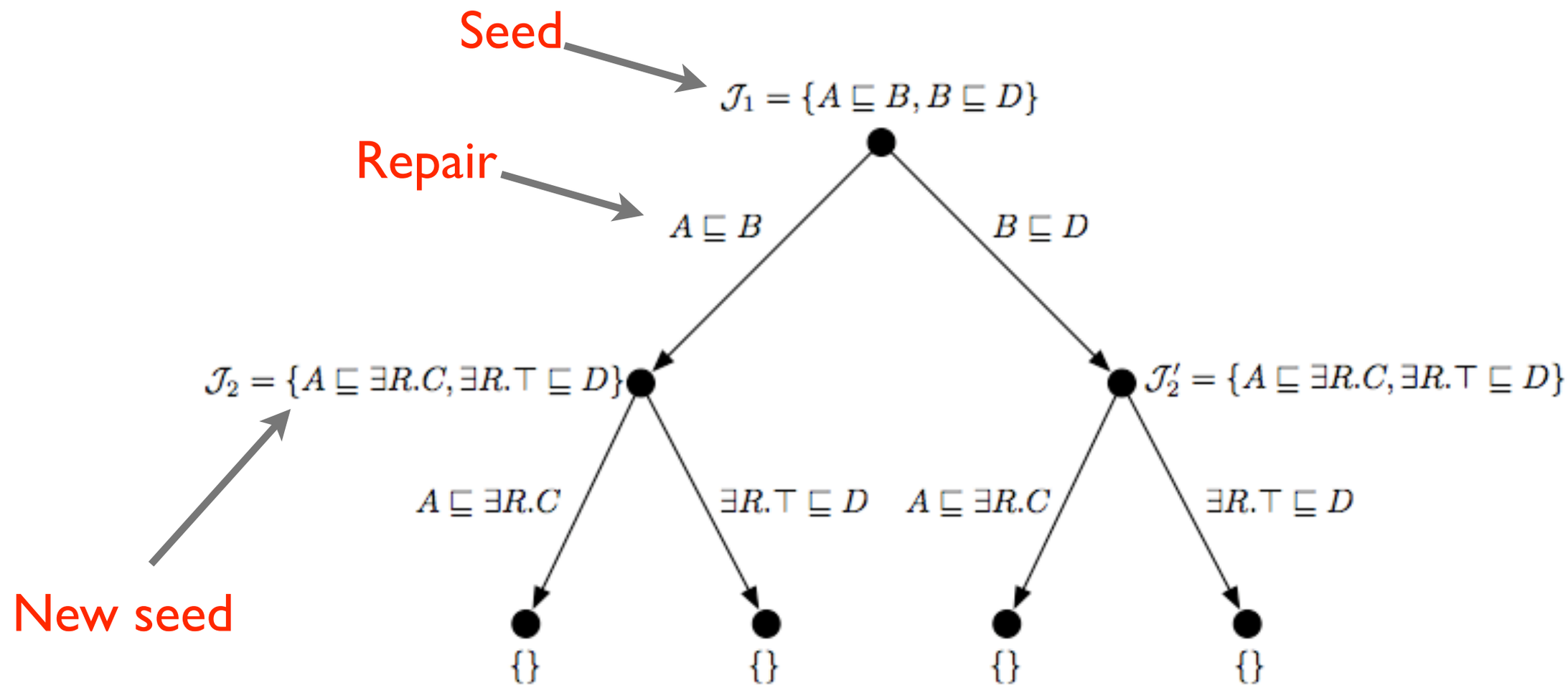
Dynamically find CSs



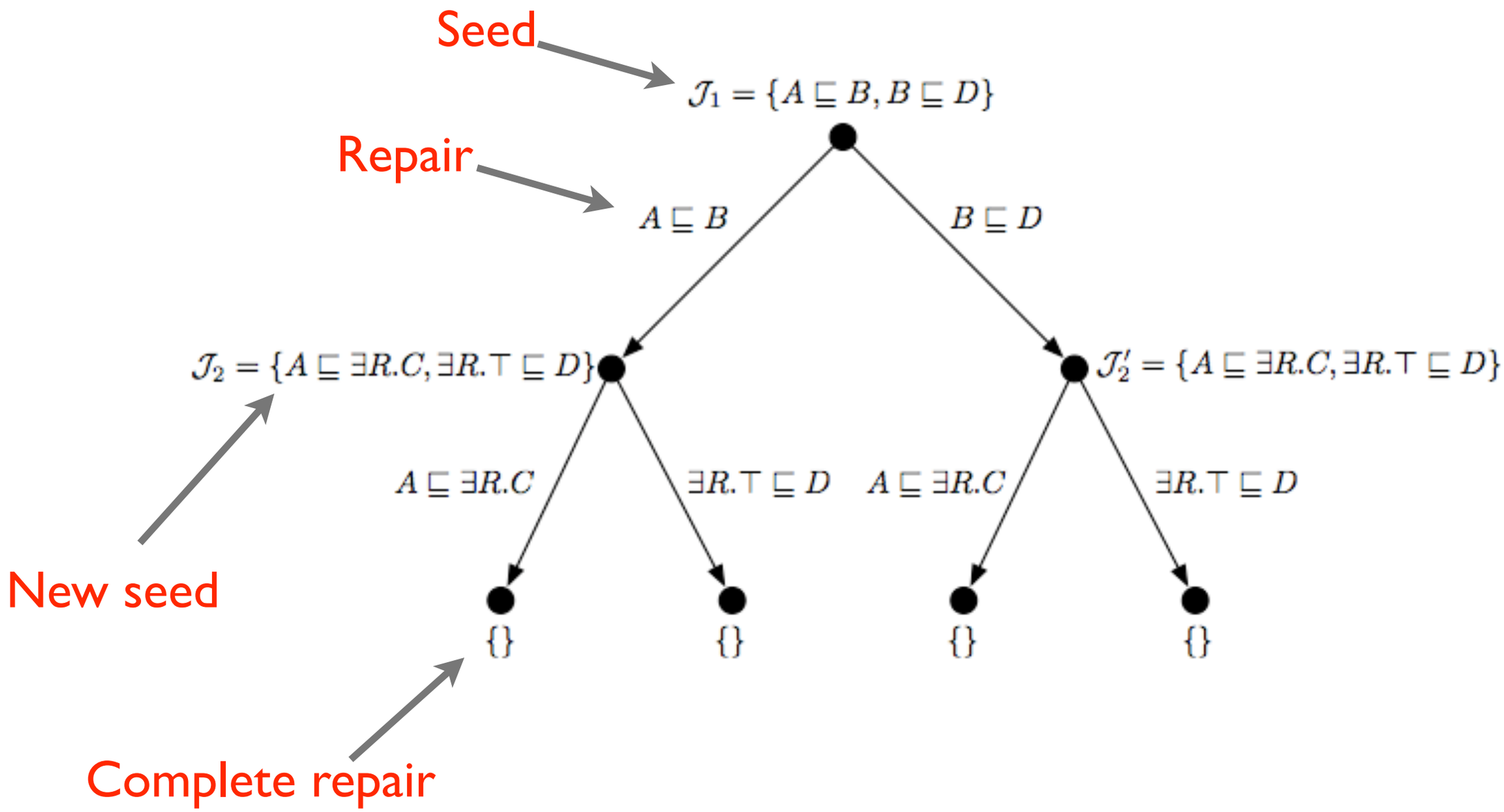
Dynamically find CSs



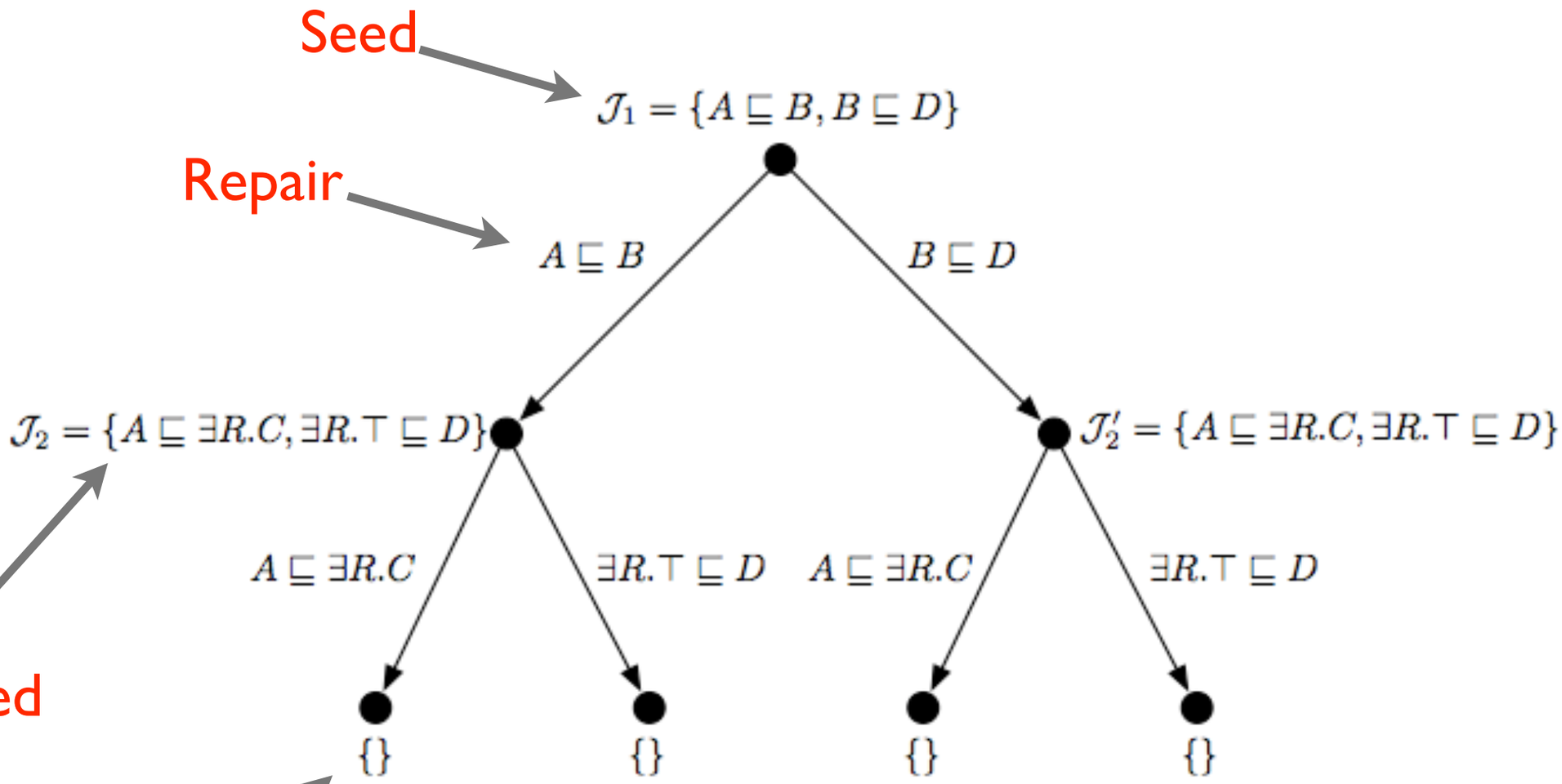
Dynamically find CSs



Dynamically find CSs



Dynamically find CSs



All repairs gives us all problems
Goal directed!

MANCHESTER 1824

The University of Manchester

Evaluation

Evaluation

Load and Prepare Ontology

↓

Classify Ontology

↓

Extract Entailments

↓

Compute Justifications

MANCHESTER 1824

The University of Manchester

The Data

- Arbitrary subsumptions
  - Consistent ontologies
    - Some UCs

Ontology	Expressivity	Axioms	C/P/I	Entailed	Domain
1. Generations	$\mathcal{ALCIF}$	335	22/4/0	24	Family tree
2. DOLCE-Lite	$\mathcal{SHOIN}(\mathcal{D})$	1417	200/299/39	3	Foundational
3. Economy	$\mathcal{ALH}(\mathcal{D})$	1704	338/53/481	51	Mid-level
4. MadCow	$\mathcal{ALCHOIN}(\mathcal{D})$	105	54/17/13	32	Tutorial
5. Tambis	$\mathcal{SHIN}$	800	395/ 100/ 0	65	Biological science
6. Sweet-JPL	$\mathcal{ALCHO}(\mathcal{D})$	3833	1537/ 121/ 150	183	Earthscience
7. Chemical	$\mathcal{ALCH}(\mathcal{D})$	254	48/20/0	43	Chemical elements
8. Transport	$\mathcal{ALH}(\mathcal{D})$	2051	444/93/183	52	Mid-level
9. MyGrid	$\mathcal{SHOIN}$	8179	550/69/13	297	Bioinformatics services
10. University	$\mathcal{SIOF}(\mathcal{D})$	169	30/12/4	23	Training
11. AminoAcids	$\mathcal{ALCF}$	2077	47/5/3	64	Classifies proteins
12. Sequence Ontology	$\mathcal{ALEHI}+$	1754	1248/17/9	179	The OBO (xp) sequence
13. Gene Ontology	$\mathcal{ALEHI}+$	1759	759/16/0	100	The OBO (xp) gene
14. MGED Ontology	$\mathcal{ALEF}(\mathcal{D})$	236	236/88/0	100	Microarray experiment

MANCHESTER 1824

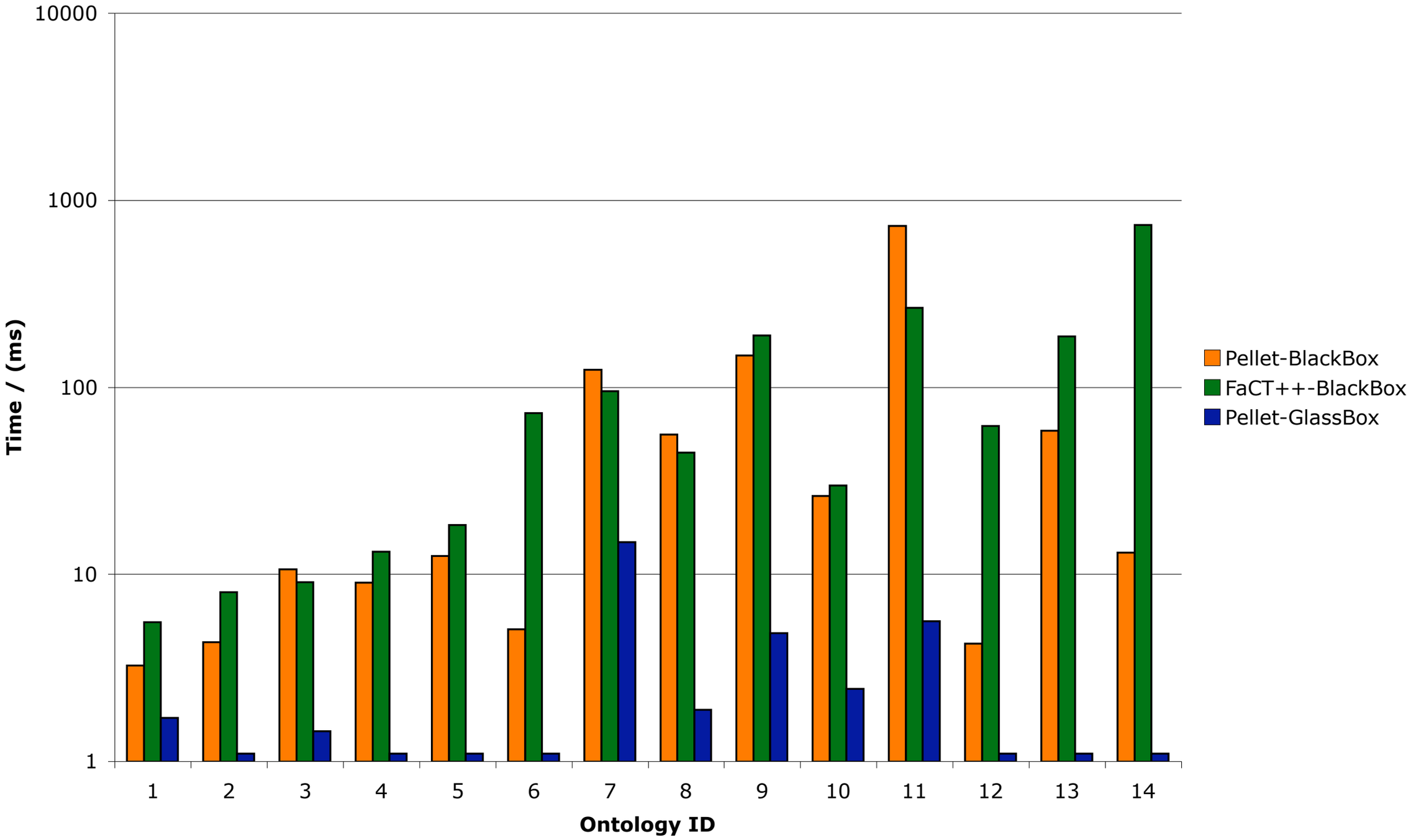
The University of Manchester

Single Justifications

MANCHESTER 1824

The University of Manchester

Single Justifications



MANCHESTER 1824

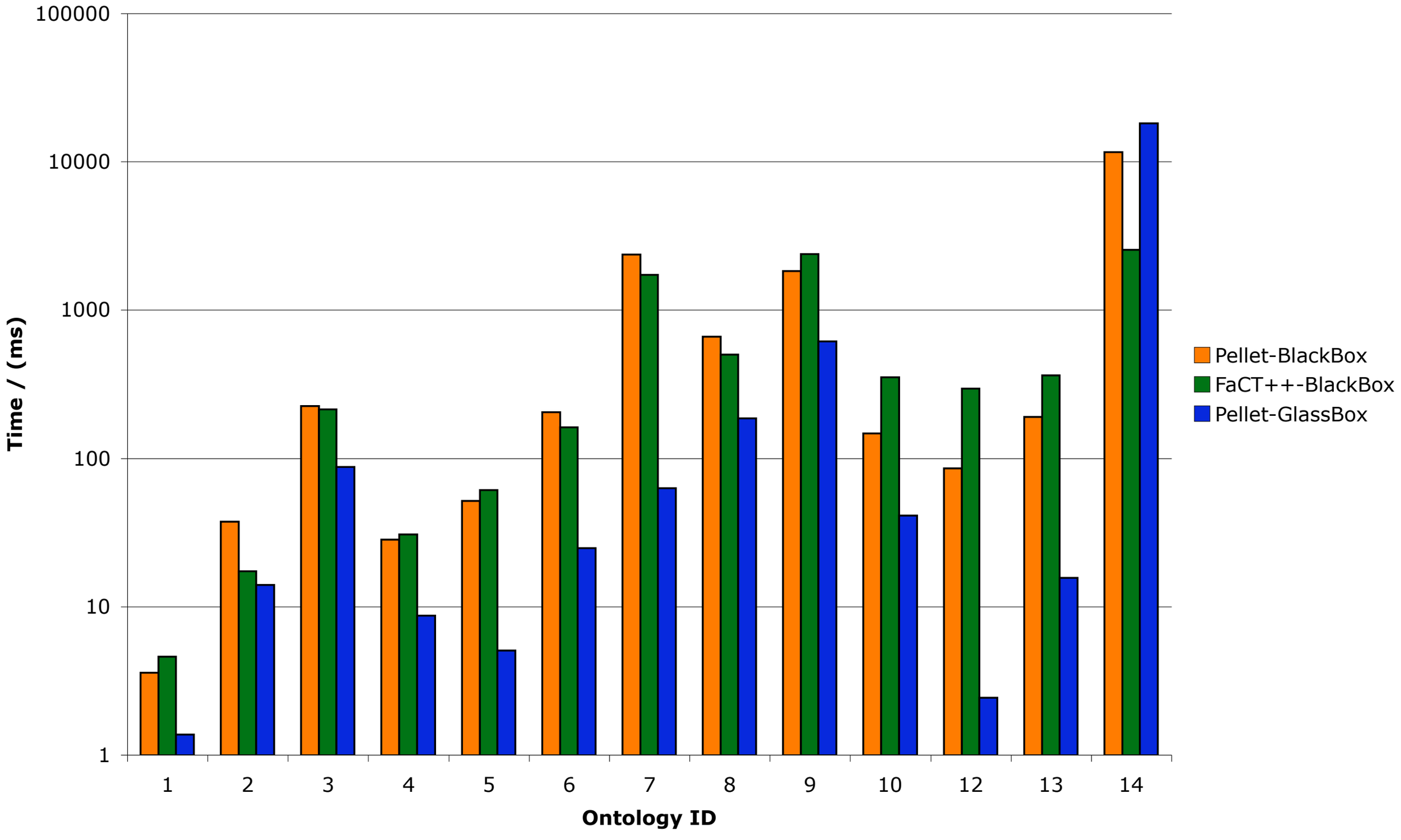
The University of Manchester

All Justifications

MANCHESTER 1824

The University of Manchester

All Justifications



MANCHESTER 1824

The University of Manchester

What about inconsistency?

- Inconsistencies have issues
  - No seed signature to guide one-just
    - Justifications accumulate
      - Current HST implementations crap out at around 80-120 justs
      - Inconsistencies easily have 100s
- Big surprise to us!

Ontology	Reasoner	Time for One (ms)	Number found in 30 mins.	Time for all (ms)
Assignment4	HermiT	-	-	⋆
	Pellet	-	-	⋆
	FaCT++	5711	1	⋆
Boat	HermiT	129	1	193
	Pellet	73	1	93
	FaCT++	11	1	41
ComparaGrid	HermiT	2252	9	15901
	Pellet	1515	9	13164
	FaCT++	-	-	⋆
Country	HermiT	4177	4	383963
	Pellet	4564	4	137348
	FaCT++	1726	4	78547
Fish	HermiT	134	162	⋆
	Pellet	-	-	⋆
	FaCT++	115	162	⋆
IedbExport	HermiT	855	2	2415
	Pellet	1257	2	3860
	FaCT++	765	2	2255
Micro	HermiT	4538	1	⋆
	Pellet	2326	1	59090
	FaCT++	809	1	1574
Spectro	HermiT	30930	76	⋆
	Pellet	10768	76	⋆
	FaCT++	-	-	⋆
Pizza	HermiT	114	3	592
	Pellet	7491	3	⋆
	FaCT++	37	3	329
ClassesAsValues	HermiT	21	1	23
	Pellet	4	2	15
	FaCT++	5	2	13
Travel	HermiT	7873	7	⋆
	Pellet	884	492	⋆
	FaCT++	521	163	⋆
Units	HermiT	3023	54	⋆
	Pellet	473	287	⋆
	FaCT++	285	287	⋆
Travel-semi-rep (semi repaired Travel ont.)	HermiT	8309	6	⋆
	Pellet	3975	6	25722
	FaCT++	1331	6	19280

Lots of side issues

- Reasoners themselves
- Interface to reasoners
  - Can dominate!
    - (RPC over HTTP hopeless)
  - Reasoners very batch oriented
- Strange behavior with subsets
  - Can be easier or (much much) harder
  - (Argues for some glass-box, sometimes)

Use of justifications for other inference services

MANCHESTER 1824

The University of Manchester

As auxiliary service

- For incremental reasoning
  - Currently overwhelmed by modularity techniques
- For default reasoning
  - “Embedding Defaults into Terminological Knowledge Representation Formalisms”
- Optimizing query answering
  - “Scalable Grounded Conjunctive Query Evaluation over Large and Expressive Knowledge Bases