THE UNIVERSITY OF IOWA

Axiom Pinpointing

Justifying consequences in Automated Reasoning

M. Fareed Arif

April 27 2020

Agenda of the Talk

01 Motivation

02 Problem Formulation

03 Proposed Solutions

04 Ongoing Work

Motivation

Intelligent applications need to represent and handle knowledge effectively

Motivation

Intelligent applications need to represent and handle knowledge effectively

A Knowledge Representation Language, say $\mathcal{K}$, provides formal semantics and reasoning methods for driving an implicit consequence, called axiom α from explicitly represented elements.

$$\boxed{\mathcal{K} \models \alpha}$$

Motivation

A Knowledge Representation Language, say $\mathcal{K}$, provides formal semantics and reasoning methods for driving an implicit consequence, called axiom α from explicitly represented elements.

$$\mathcal{K} \models \alpha$$

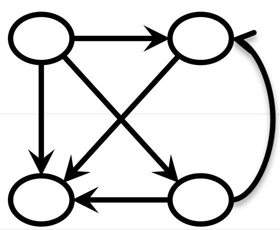
Understand the error and repair to rectify the error

Debugging Ontologies

SNOMED-CT is a medical ontology used by U.S Federal Government systems
For the electronic exchange of clinical health information

$\text{AmpOfFinger} \sqsubseteq \text{AmpOfHand}$

Amputation of finger is an Amputation of a Hand [BP08]



Ontology $\mathcal{O}$

Unexpected Behavior $\mathcal{B}$

Does $\mathcal{O}$ entails $\mathcal{B}$?

Bug Explanation and Repair

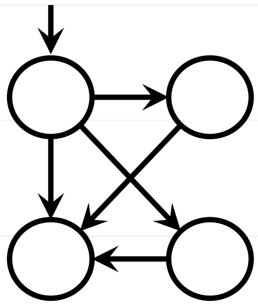
Motivation

A Knowledge Representation Language, say $\mathcal{K}$, provides formal semantics and reasoning methods for driving an implicit consequence, called axiom α from explicitly represented elements.

$$\mathcal{K} \models \alpha$$

If **Consequence** is a property of an interest then **explaining** it enhances both the correctness and acceptance of an intelligent computational system

Symbolic Model Checking



Abstract Model $\mathcal{M}$

Requirements $\mathcal{R}$

Does $\mathcal{M}$ meets $\mathcal{R}$?

Proof Explanation

Proof explanation in Symbolic Model Checking [GWG17]

Pinpointing lines in the source code responsible for an error [JM11]

Axiom Pinpointing:

Identifying the axioms in a $\mathcal{K}$ that are responsible for a given consequence α

Identified axioms are called justification

A single consequence can have multiple justifications

Finding not only one but all justification-based explanations

Problem Relevance

Axiom pinpointing is a well-studied problem across several domains

Different research communities know **justification** by different names

[RR87]

CSP

MESCs

[MM14]

DL

MUPS

MinAs

[SC03, BP07]

SAT

MUSes

[BL03,LS08]

DBs

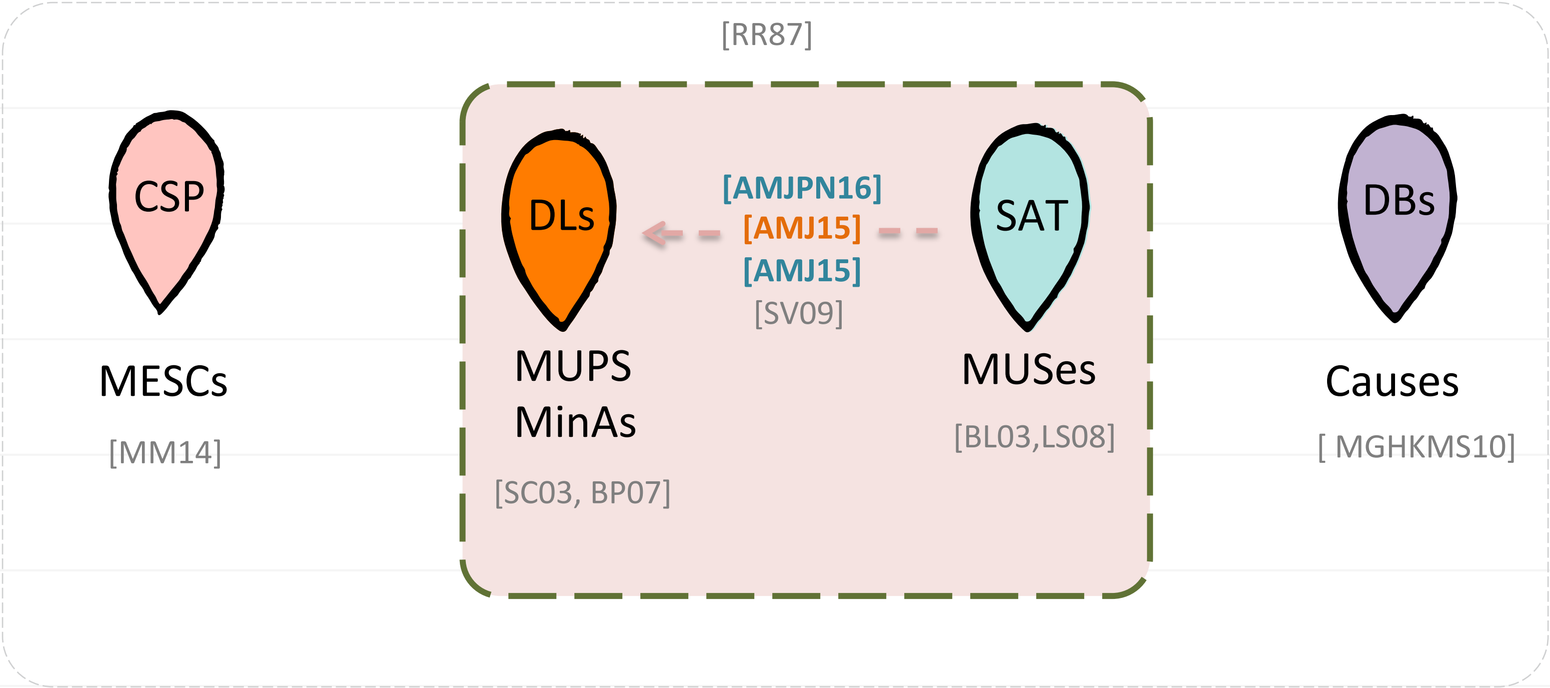
Causes

[MGHKMS10]

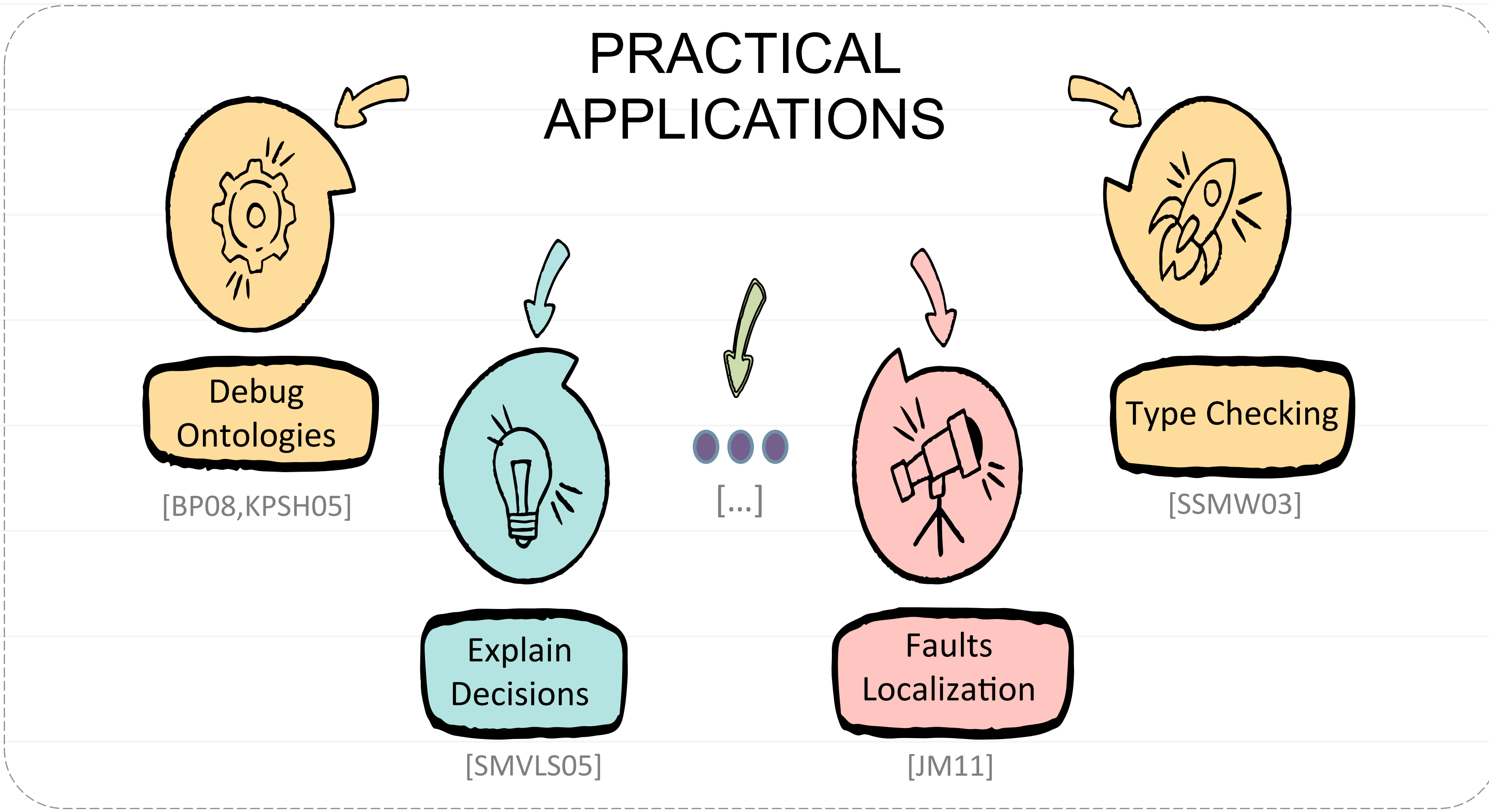
Problem Relevance

Axiom pinpointing is a well-studied problem across several domains

Different research communities know **justification** by different names



Axiom pinpointing is well-studied problem across several domains and has many interesting applications



02 Problem Formulation

A Simple Ontology Language

A simple Ontology Language $\mathcal{L}(\mathcal{O})$ that contains four components

$$\mathcal{L}(\mathcal{O}) \begin{cases} \mathcal{A} \longrightarrow \text{a class of well-formed axioms} \\ \mathcal{C} \longrightarrow \text{a class of consequences} \\ \mathscr{O} \subseteq \mathscr{P}(\mathcal{A}) \longrightarrow \text{a valid-set of ontologies} \\ \models \subseteq \mathscr{O} \times \mathcal{C} \longrightarrow \text{entailment relation} \end{cases}$$

We only consider a monotonic relation:

$$\mathcal{O}, \mathcal{O}' \in \mathscr{O} \text{ and } c \in \mathcal{C} \text{ if } \mathcal{O} \models c \text{ and } \mathcal{O} \subseteq \mathcal{O}', \text{then } \mathcal{O}' \models c$$

A Simple Ontology Language:

Given an ontology $\mathcal{O} \in \mathscr{O}$ and a consequence $c \in \mathcal{O}$

Entailment:

Decide $\mathcal{O} \models c$ holds?

Justification: $\mathcal{O} \models c$

A minimal subset Ontology $\mathcal{M} \subseteq \mathcal{O}$ that meets two conditions:

1. $\mathcal{M} \models c$
2. For every $\mathcal{M}' \subsetneq \mathcal{M}, \mathcal{M} \not\models c$

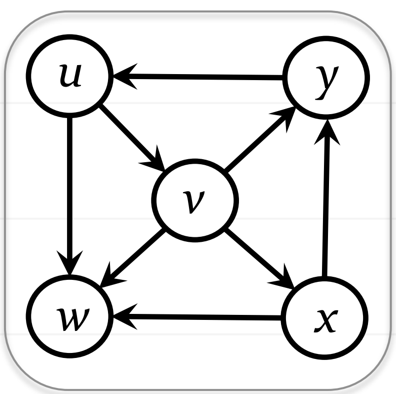
Graph Ontology Example:

$\mathcal{V}$ $\longrightarrow$ a countable set of vertices

$\mathcal{A} = \mathcal{C} = \{(v, w) | v, w \in \mathcal{V}\}$ $\longrightarrow$ edges

$\mathscr{O} \subseteq \mathscr{P}(\mathcal{A})$ $\longrightarrow$ a finite set of graphs

$\models$ $\longrightarrow$ any two reachable vertices in a graph



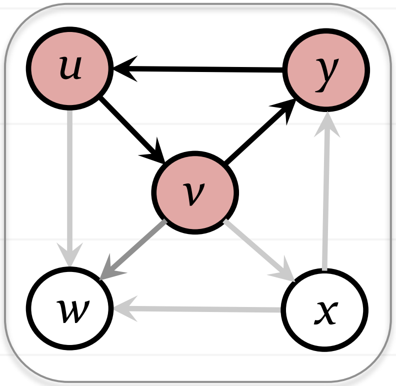
$\mathcal{G} \in \mathscr{O}$ is a finite graph, such that

$\mathcal{G} \models (u, x)$ but $\mathcal{G} \not\models (w, u)$

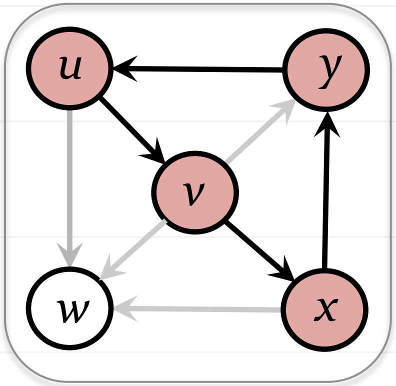
Axiom Pinpointing:

A justification is a minimal sub-ontology that still entails the consequence

Find one or more justification in ontology $\mathcal{G}$ for a consequence: u is reachable from u.



$\mathcal{G}1 \models (\mathrm{u,u})$



$\mathcal{G}2 \models (\mathrm{u,u})$

Over-constrained Formulas in SAT:

Minimal Unsatisfiable Subset [MUS]:

$\mathcal{M} \subseteq \mathcal{F}$ is minimally unsatisfiable subformula of $\mathcal{F}$ if and only if

$\mathcal{M}$ is unsatisfiable and $\forall_{\mathcal{M}' \subsetneq \mathcal{M}} \mathcal{M}'$ is satisfiable

Minimal Correction Subset [MCS]:

$\mathcal{C} \subseteq \mathcal{F}$ is minimally correction subformula of $\mathcal{F}$ if and only if

$\mathcal{F} \setminus \mathcal{C}$ is satisfiable and $\forall_{\mathcal{C}' \subsetneq \mathcal{C}} \mathcal{F} \setminus \mathcal{C}'$ is unsatisfiable

03 Proposed Solutions

Black-box Method:

Compute one justification by removing an axiom at a time

Data: Ontology $\mathcal{O}$, consequence c

Result: A justification $\mathcal{M}$ for c w.r.t. $\mathcal{O}$

$\mathcal{M} \leftarrow \mathcal{O}$

for $\alpha \in \mathcal{O}$ **do**

  if $\mathcal{M} \setminus \{\alpha\} \models c$ **then**

    $\mathcal{M} \leftarrow \mathcal{M} \setminus \{\alpha\}$

Return $\mathcal{M}$

There is a relation between the order of axioms selected for removal and the computed justification

Glass-box Method:

Modified Tableau-based reasoning method for standard reasoning can track the relevant traces but are inefficient

Tableau-based pinpointing approaches does not behave well in practice

Glass-box Method:

Modified Tableau-based reasoning method for standard reasoning can track the relevant traces but are inefficient

Tableau-based pinpointing approaches does not behave well in practice

A propositional formula encodes the execution of a consequence-based algorithm

Enumeration over Justifications

Increase our understanding about the derivation by finding more than one or all justifications

To find all Justifications, one can simply enumerate all sub-ontologies

Problem: Exponential many entailment checks, one for each subset of $\mathcal{O}$

The question is: can we do better?

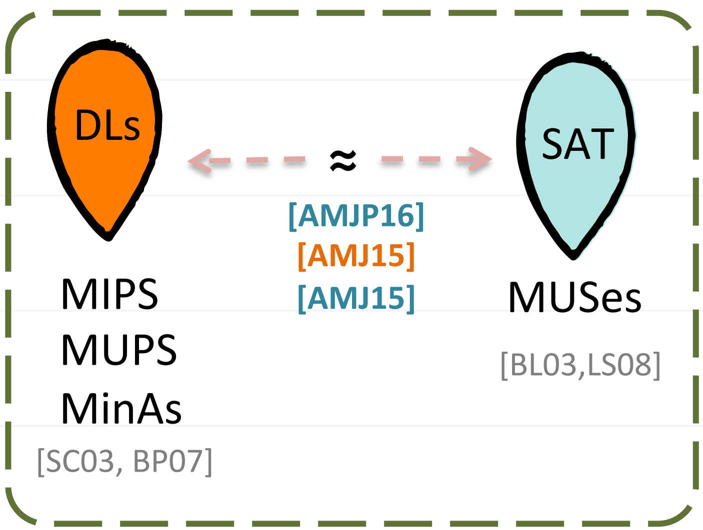
Contribution

[RR87]

Minimal Conflict Set (Justification) ⟷ Hitting set Duality ⟷ Diagnosis (Repair)

[AMJ15]

MUSes are justifications and MCSes are repairs



Optimizations:

[AMJ15]

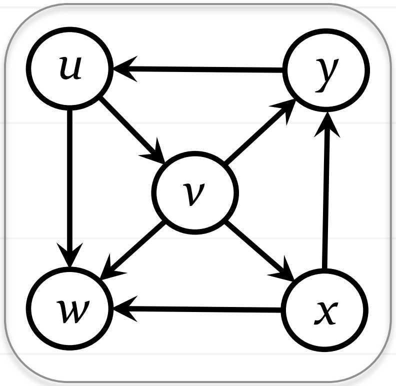
If a set $\mathcal{O}' \subseteq \mathcal{O}$ is such that $\mathcal{O}' \models c$, then ignore all strict superset of $\mathcal{O}'$ because minimality condition fails for superset

If $\mathcal{O}' \not\models c$ then ignore subset of $\mathcal{O}'$ because subsets fail to entail c

Consequence-based Axiom Pinpointing:

[SV09]

A Horn clause simulates each rule possible application and a propositional variable represents a conclusion

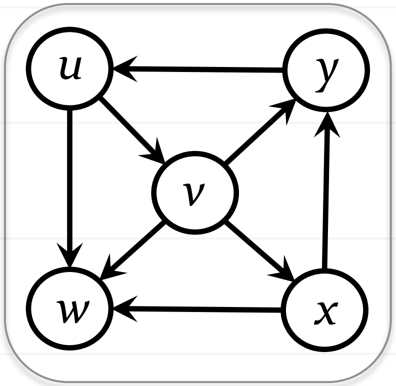


$$\mathcal{F}_{\mathcal{G}} = x_{(u,v)} \wedge x_{(v,y)} \wedge x_{(x,w)} \wedge x_{(y,u)}$$
$$x_{(u,w)} \wedge x_{(v,w)} \wedge x_{(x,y)} \wedge x_{(v,x)}$$

Consequence-based Axiom Pinpointing:

[SV09]

A Horn clause simulates each rule possible application and a propositional variable represents a conclusion



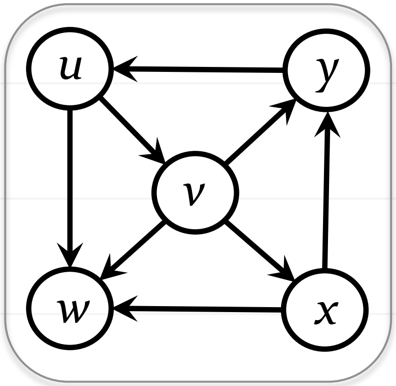
Constraints encode the edges of the example graph

Graph Ontology Example:

A consequence-based algorithm contains only one rule:

$$\{(X,Y),(Y,Z)\} \Rightarrow \{(X,Z)\}$$

$$\mathcal{F}_{\mathcal{H}} = \begin{array}{l} x_{(u,v)} \wedge x_{(v,y)} \rightarrow x_{(u,y)} \\ x_{(u,v)} \wedge x_{(v,w)} \rightarrow x_{(u,w)} \\ x_{(u,v)} \wedge x_{(v,x)} \rightarrow x_{(u,x)} \\ x_{(u,v)} \wedge x_{(v,y)} \rightarrow x_{(u,y)} \\ x_{(u,x)} \wedge x_{(x,w)} \rightarrow x_{(u,w)} \\ x_{(u,x)} \wedge x_{(x,y)} \rightarrow x_{(u,y)} \\ x_{(u,y)} \wedge x_{(y,u)} \rightarrow x_{(u,u)} \end{array}$$

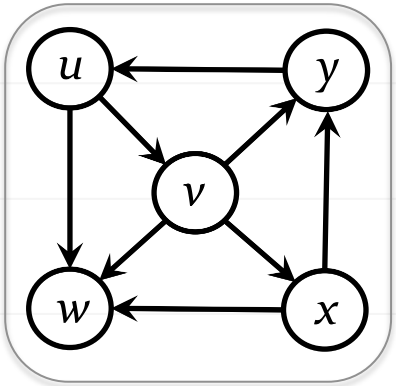


Graph Ontology Example:

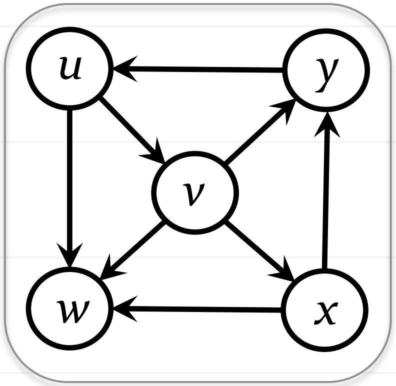
A consequence-based algorithm contains only one rule:

$$\{(X, Y), (Y, Z)\} \Rightarrow \{(X, Z)\}$$

Constraints encode all the reachable paths in the example graph



Graph Ontology Example:

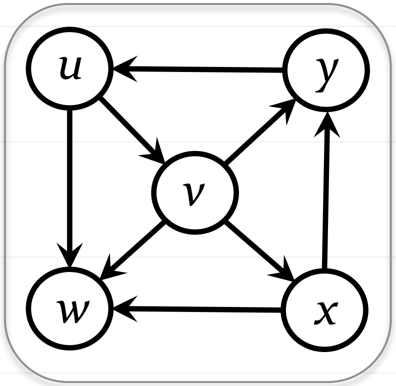


$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$

$$\mathcal{F}_{\mathcal{H}} = x_{(u,v)} \wedge x_{(v,y)} \rightarrow x_{(u,y)} \wedge$$
$$x_{(u,v)} \wedge x_{(v,w)} \rightarrow x_{(u,w)} \wedge x_{(u,x)} \wedge x_{(x,w)} \rightarrow x_{(u,w)}$$
$$x_{(u,v)} \wedge x_{(v,x)} \rightarrow x_{(u,x)} \wedge x_{(u,x)} \wedge x_{(x,y)} \rightarrow x_{(u,y)}$$
$$x_{(u,v)} \wedge x_{(v,y)} \rightarrow x_{(u,y)} \wedge x_{(u,y)} \wedge x_{(y,u)} \rightarrow x_{(u,u)}$$

$$\mathcal{F}_{\mathcal{G}} = \begin{array}{l} x_{(u,v)} \wedge x_{(v,y)} \wedge x_{(x,w)} \wedge x_{(y,u)} \\ x_{(u,w)} \wedge x_{(v,w)} \wedge x_{(x,y)} \wedge x_{(v,x)} \end{array}$$

Graph Ontology Example:



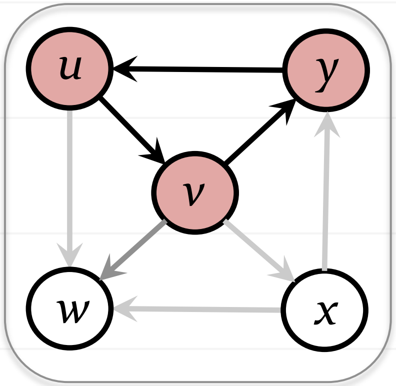
$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$

Constraints encode all the reachable paths in the example graph

Constraints encode the edges of the example graph

Graph Ontology Example:

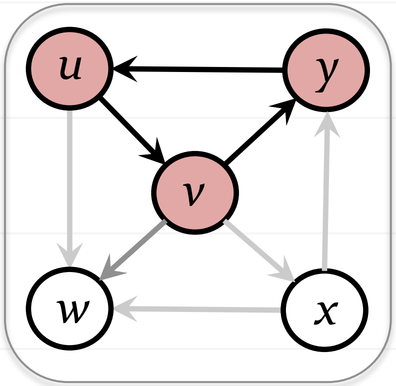
$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$



$$\mathcal{G} \models (u, u) \quad \approx \quad \mathcal{F} \models x_{(u,u)}$$

Graph Ontology Example:

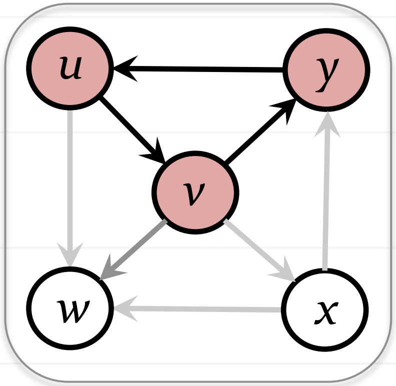
$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$



Entailment Check

Graph Ontology Example:

$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$



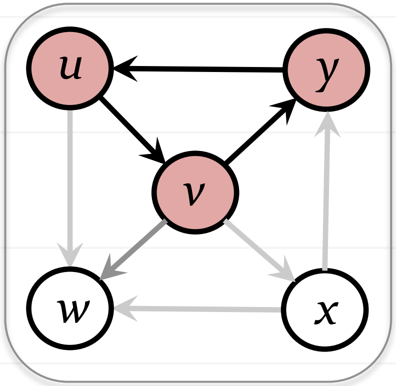
$$\mathcal{F}_{\mathcal{H}} \quad \begin{array}{l} x_{(u,v)} \wedge x_{(v,y)} \to x_{(u,y)} \\ x_{(u,y)} \wedge x_{(y,u)} \to x_{(u,u)} \end{array}$$

$$\mathcal{F}_{\mathcal{G}} \quad x_{(u,v)} \quad x_{(v,y)} \quad x_{(y,u)}$$

$$\mathcal{G} \models (u,u) \quad \approx \quad \mathcal{F} \models x_{(u,u)}$$

Graph Ontology Example:

$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$



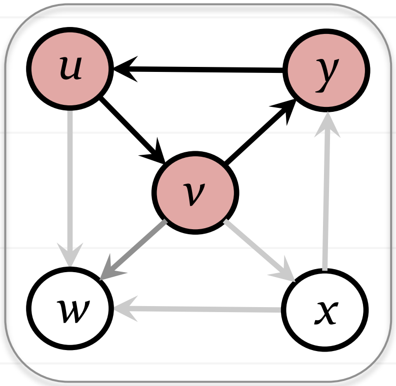
$$\mathcal{F}_{\mathcal{H}} \quad \begin{array}{l} x_{(u,v)} \wedge x_{(v,y)} \to x_{(u,y)} \\ x_{(u,y)} \wedge x_{(y,u)} \to x_{(u,u)} \end{array}$$

$$\mathcal{F}_{\mathcal{G}} \quad x_{(u,v)} \quad x_{(v,y)} \quad x_{(y,u)}$$

$$\mathcal{G} \models (u, u) \quad \approx \quad \mathcal{F} \cup \{\neg x_{(u,u)}\}$$

Graph Ontology Example:

$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$



$\mathcal{F}_{\mathcal{H}}$

$$x_{(u,v)} \wedge x_{(v,y)} \rightarrow x_{(u,y)}$$

$$x_{(u,y)} \wedge x_{(y,u)} \rightarrow x_{(u,u)}$$

$\mathcal{F}_{\mathcal{G}}$

$$x_{(u,v)} \quad x_{(v,y)} \quad x_{(y,u)}$$

$$\mathcal{G} \models (u, u) \quad \approx \quad \mathcal{F} \cup \{\neg x_{(u,u)}\}$$

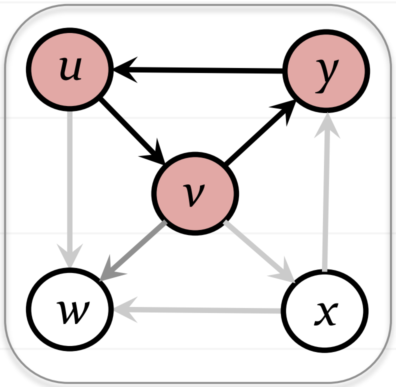
$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}} \wedge \neg x_{(u,u)}$$

Graph Ontology Example:

[AMJ15]

$$\mathcal{F} = \mathcal{F}_{\mathcal{H}} \wedge \mathcal{F}_{\mathcal{G}}$$

$$\mathcal{G} \models (u, u)$$



$$\mathcal{F}_{\mathcal{H}} \quad \begin{array}{l} x_{(u,v)} \wedge x_{(v,y)} \to x_{(u,y)} \\ x_{(u,y)} \wedge x_{(y,u)} \to x_{(u,u)} \end{array}$$

$$\mathcal{F}_{\mathcal{G}} \quad x_{(u,v)} \quad x_{(v,y)} \quad x_{(y,u)}$$

$$\mathcal{F} = \overbrace{\mathcal{F}_{\mathcal{H}}}^{\mathcal{G}_0} \wedge \underbrace{\mathcal{F}_{\mathcal{G}}}_{\mathcal{G}_1, \ldots, \mathcal{G}_8} \wedge \neg x_{(u,u)}$$

Graph Ontology Example:

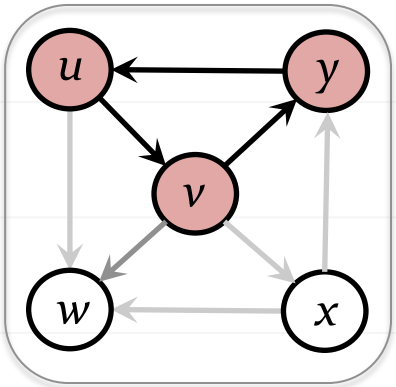
[AMJ15]

$\mathcal{F} = \mathcal{G}_0 \cup \cdots \cup \mathcal{G}_k$, a group-MUS of $\mathcal{F}$ is a set of groups

$\mathcal{G} \subseteq \{\mathcal{G}_1 \cup \cdots \cup \mathcal{G}_k\}$, such that

$\mathcal{G}_0 \cup \mathcal{G}$ is unsatisfiable and $\mathcal{G}_i \in \mathcal{G}, \mathcal{G}_0 \cup (\mathcal{G} \setminus \mathcal{G}_i)$ is satisfiable

$$\mathcal{F} = \overbrace{\mathcal{F}_{\mathcal{H}}}^{\mathcal{G}_0} \wedge \underbrace{\mathcal{F}_{\mathcal{G}}}_{\mathcal{G}_1, \ldots, \mathcal{G}_8} \wedge \neg x_{(u,u)}$$



$$\mathcal{F} \models x_{(u,u)}$$

Graph Ontology Example:

[AMJ15]

$$\mathcal{F} = \overbrace{\mathcal{F}_{\mathcal{H}}}^{\mathcal{G}_0} \wedge \underbrace{\mathcal{F}_{\mathcal{G}}}_{\mathcal{G}_1, \ldots, \mathcal{G}_8} \wedge \neg x_{(u,u)}$$

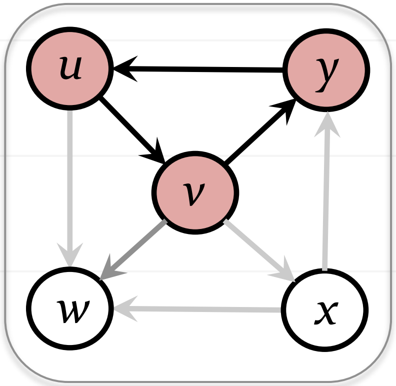
$$\boxed{\mathcal{G}_0 = \mathcal{F}_{\mathcal{H}}}$$

$$\mathcal{G}_1 = \{x_{(u,v)}\} \quad \mathcal{G}_3 = \{x(x,w)\} \quad \mathcal{G}_5 = \{x(u,w)\}$$

$$\mathcal{G}_2 = \{x(v,y)\} \quad \mathcal{G}_4 = \{x(y,u)\} \quad \mathcal{G}_6 = \{x(v,w)\}$$

$$\mathcal{G}_7 = \{x(x,y)\} \quad \mathcal{G}_8 = \{x(v,x)\}$$

Graph Ontology Example:



$$\mathcal{G}_0 = \mathcal{F}_{\mathcal{H}}$$

$\mathcal{G}_1 = \{x_{(u,v)}\}$ $\mathcal{G}_2 \models \{x_{(x,w)}\}$ $\mathcal{G}_5 \models \{x_{(u,w)}\}$

$\mathcal{G}_2 \models \{x_{(v,y)}\}$ $\mathcal{G}_4 \models \{x_{(y,u)}\}$ $\mathcal{G}_6 \models \{x_{(v,w)}\}$

$\mathcal{G}_7 \models \{x_{(x,y)}\}$ $\mathcal{G}_8 \models \{x_{(v,x)}\}$

$$\mathcal{F} = \mathcal{G}_0 \cup \mathcal{G}_1 \cup \cdots \cup \mathcal{G}_8 \{\neg x_{(u,u)}\}$$

$$\mathcal{M} = \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_4\}$$

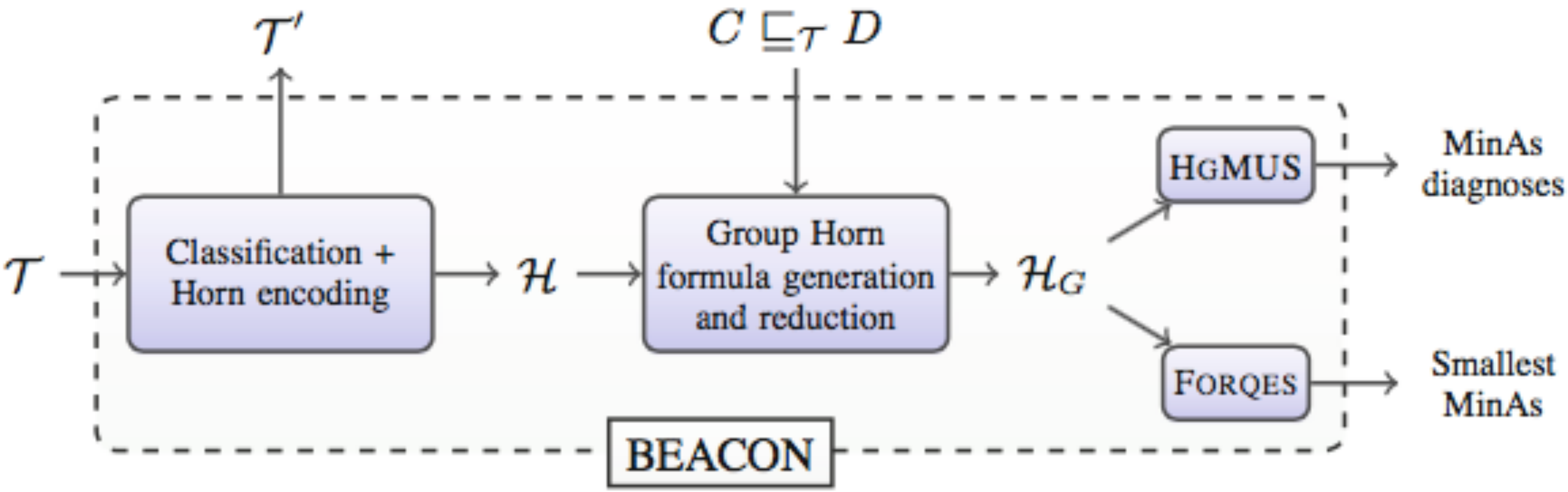
Hitting set Duality

$$\mathcal{S}_1 = \{\mathcal{G}_1\}$$
$$\mathcal{S}_2 = \{\mathcal{G}_2\}$$
$$\mathcal{S}_3 = \{\mathcal{G}_4\}$$

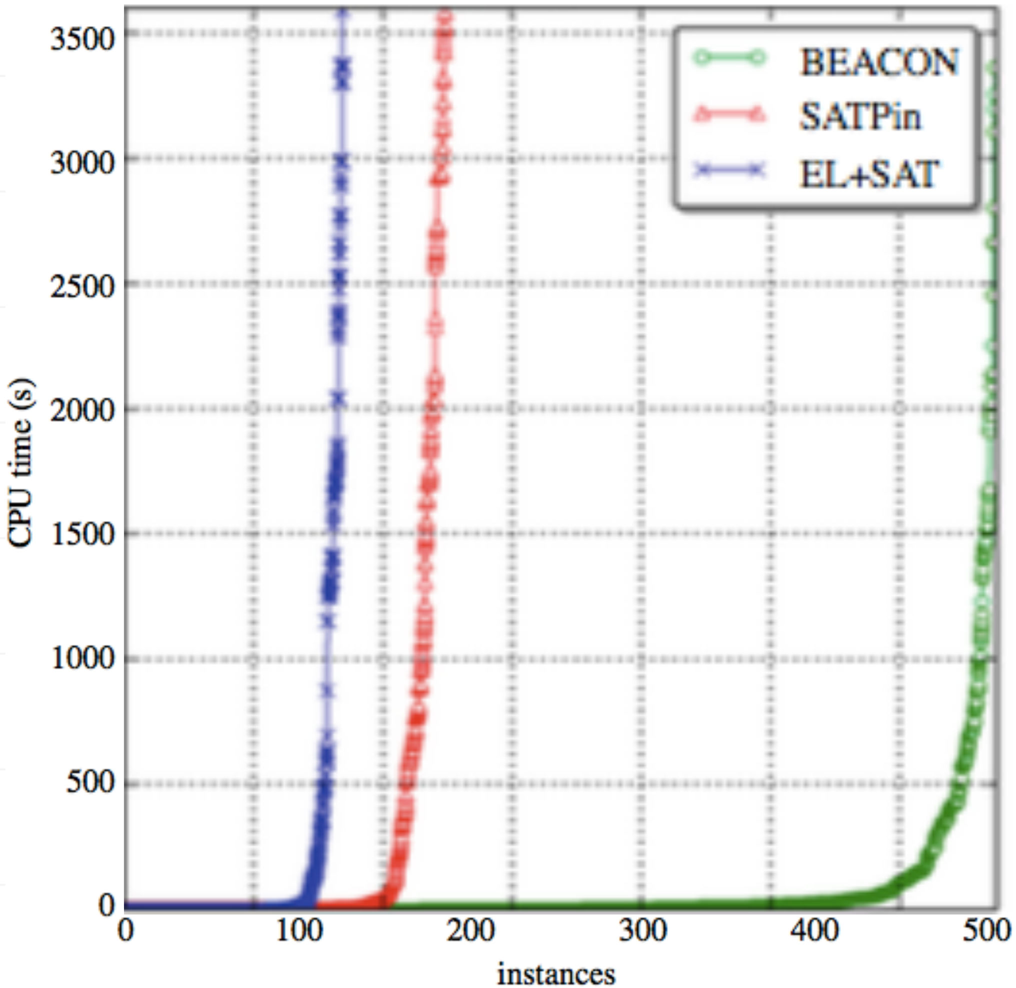
[AMJ15]

BEACON Tool:

[AMJP16]



Performance:



Medical Ontologies

- ❑ SNOMED-CT
- ❑ GENE
- ❑ GALEN
- ❑ NCI

04 Ongoing Work

List of applications

1. Debugging Medical Ontologies [FS08]
2. Requirement analysis [JB17]
3. Equivalence checking [OGNLV10]
4. Counter-example Guided Abstraction Refinement[CGJLV00]
5. Proof-based abstraction refinement [MA03]
6. Boolean function bi-decomposition [CM11]
7. Circuit error Diagnosis [HL99]
8. Type Debugging in Haskell [SSW03]
9. Proof explanation in Symbolic model checking[GWG17]

Domain of Constraints:

1. Axioms of an Ontology
2. Boolean Formulas
3. Temporal logic formula
4. Transition state Predicates

Domain agnostic MUS enumeration Tool

University of Iowa

Computational Logic Centre

M. Fareed Arif

April 27 2020