TECHNISCHE UNIVERSITÄT DRESDEN

Computational Logic Group

Hannes Strass (based on slides by Bernardo Cuenca Grau, Ian Horrocks, Przemysław Wałęga)

Faculty of Computer Science, Institute of Artificial Intelligence, Computational Logic Group

Horn Logics and Datalog

Lecture 3, 24th Oct 2022 // Foundations of Knowledge Representation, WS 2022/23

Recap

- Looked at using Propositional Logic (PL) for representing knowledge
- effectively implementable (SAT solvers), but lacks expressiveness

Want a KR language that can …

1. Represent sets of objects
2. Represent relationships between objects
3. Write statements that are true for some or all objects satisfying certain conditions
4. Express everything we can express in propositional logic (*and*, *or*, *implies*, *not*, …)

$\rightsquigarrow$ First-order logic (FOL)

However: FOL satisfiability is undecidable

$\rightsquigarrow$ cannot hope for implementations

Propositional Horn Fragment

PL Horn Fragment: only allows the following formulas ($n > 0$):

$$P_1 \wedge \ldots \wedge P_n \rightarrow Q \qquad \textit{rules}$$
$$P \qquad \textit{facts}$$

With P_i, Q being atoms, and where Q can be $\bot$.

Horn Clauses: Clauses with at most one positive literal.

$$\neg P_1 \vee \ldots \vee \neg P_n \vee Q$$

> (fact) entailment. Instance is set $\mathcal{H}$ of Horn formulas and atom P
> Answer is true if every model of $\mathcal{H}$ is also a model of P
> and false otherwise.

PL Horn entailment is solvable in polynomial time.

Lifting PL Horn to FOL Horn

First-Order Horn Clauses: Clauses with at most one positive literal

But now, atoms can contain variables, constants, and function symbols.

Some examples of First-Order Horn clauses:

$$\neg \mathit{JuvArthritis}(x) \lor \mathit{Arthritis}(x)$$
$$\neg \mathit{Arthritis}(x) \lor \neg \mathit{JuvDisease}(x) \lor \mathit{JuvArthritis}(x)$$
$$\neg \mathit{Child}(x) \lor \neg \mathit{Adult}(x)$$
$$\neg \mathit{Affects}(x, y) \lor \mathit{Person}(y)$$
$$\neg \mathit{JuvDisease}(x) \lor \mathit{Affects}(x, f(x))$$
$$\mathit{JuvDisease}(\mathit{JRA})$$

Horn Logics

Horn Formulas: FOL sentences that in CNF yield Horn clauses.

Horn Logics: Syntactic FOL fragments allowing only Horn Formulas.

Some examples of Horn formulas:

$$\forall x.(\mathit{Arthritis}(x) \wedge \mathit{JuvDisease}(x) \rightarrow \mathit{JuvArthritis}(x))$$

$$\forall x.(\mathit{Child}(x) \wedge \mathit{Adult}(x) \rightarrow \bot)$$

$$\forall x.(\forall y.(\mathit{Affects}(x, y) \rightarrow \mathit{Person}(y)))$$

$$\forall x.(\mathit{JuvDisease}(x) \rightarrow \exists y.(\mathit{Affects}(x, y) \wedge \mathit{Child}(y)))$$

$$\forall x.(\forall y.(\forall z.(\mathit{fatherOf}(x, y) \wedge \mathit{brotherOf}(x, z) \rightarrow \mathit{uncleOf}(z, y))))$$

$$\mathit{JuvDisease}(\mathit{JRA})$$

Expressivity

We cannot express "disjunctive formulas":

- Covering statements:

$$\forall x.(\mathit{Person}(x) \to \mathit{Adult}(x) \lor \mathit{Child}(x) \lor \mathit{Teenager}(x))$$

- Negation on the left of implication

$$\forall x.(\mathit{Person}(x) \land \neg \mathit{Woman}(x) \to \mathit{Man}(x))$$

As well as many others ...

Note, however, that some formulas apparently "disjunctive" are Horn:

$$\forall x.(\mathit{Adult}(x) \lor \mathit{Child}(x) \lor \mathit{Teenager}(x) \to \mathit{Person}(x))$$

Expressivity

We cannot express "disjunctive formulas":

- Covering statements:

$$\forall x.(Person(x) \rightarrow Adult(x) \vee Child(x) \vee Teenager(x))$$

- Negation on the left of implication

$$\forall x.(Person(x) \wedge \neg Woman(x) \rightarrow Man(x))$$

As well as many others ...

Note, however, that some formulas apparently "disjunctive" are Horn:

$$\forall x.(Adult(x) \vee Child(x) \vee Teenager(x) \rightarrow Person(x))$$

...because they can be rewritten into formulas that are obviously Horn:

$$\forall x.(Adult(x) \rightarrow Person(x))$$
$$\forall x.(Child(x) \rightarrow Person(x))$$
$$\forall x.(Teenager(x) \rightarrow Person(x))$$

Existential Rules

$$\forall\vec{x}.\forall\vec{z}.(\varphi(\vec{x},\vec{z}) \to \exists\vec{y}.\psi(\vec{x},\vec{y})) \qquad \textit{Existential Rule}$$

$$\forall\vec{x}.\forall\vec{z}.(\varphi(\vec{x},\vec{z}) \to \bot) \qquad \bot\text{–}\textit{Rule}$$

$$P(\vec{a}) \qquad \textit{Fact}$$

$\varphi(\vec{x},\vec{z})$: conjunction of function-free atoms with vars $\vec{x} \cup \vec{z}$.
$\psi(\vec{x},\vec{y})$: conjunction of function-free atoms with vars $\vec{x} \cup \vec{y}$.

$$\forall x.(\mathit{Arthritis}(x) \land \mathit{JuvDisease}(x) \to \mathit{JuvArthritis}(x)) \qquad \textit{Rule}$$

$$\forall x.(\mathit{Child}(x) \land \mathit{Adult}(x) \to \bot) \qquad \bot\text{-}\textit{Rule}$$

$$\forall x.(\mathit{JuvDisease}(x) \to \exists y.(\mathit{Affects}(x,y) \land \mathit{Child}(y))) \qquad \textit{Rule}$$

$$\mathit{JuvDisease}(\mathit{JRA}) \qquad \text{Fact}$$

Examples of Horn formulas outside this logic:

$$\forall x.(\mathit{Adult}(x) \lor \mathit{Child}(x) \lor \mathit{Teenager}(x) \to \mathit{Person}(x))$$

Reasoning with Existential Rules

Fact Entailment: An instance is a pair $\langle \mathcal{R}, \mathcal{F} \rangle$ of rules and facts and a fact P.
The answer is true iff $\langle \mathcal{R}, \mathcal{F} \cup \{\neg P\} \rangle$ is unsatisfiable.

Resolution can be optimised for Horn clauses.

General strategy: allow only certain kinds of resolution inferences:

- Need to show completeness
  Unsatisfiability must imply that the empty clause is derivable.
- No need to show soundness
  Still just resolution, which is sound.

Recall FOL Resolution Rule

$$\frac{\alpha \vee \varphi \qquad \neg\beta \vee \psi}{(\varphi \vee \psi)MGU(\alpha, \beta)}$$

α, β are atoms
MGU(α, β) is Most General Unifier of α and β

Examples:

$$\frac{(\neg ArthritisPat(x) \vee Affects(f(x), x)) \qquad ArthritisPat(g(a))}{Affects(f(g(a)), g(a))} \qquad \{x \mapsto g(a)\}$$

$$\frac{Affects(x, John) \qquad \neg Affects(JRA, y)}{\square} \qquad \{x \mapsto JRA, y \mapsto John\}$$

$$\frac{JuvDisease(h(g(f(x), a))) \qquad \neg JuvDisease(h(g(y, y)))}{\textit{Rule not applicable}}$$

Recall FOL Factoring Rule

$$\frac{\gamma \vee \delta \vee \psi}{(\gamma \vee \psi)MGU(\gamma, \delta)} \quad \gamma, \delta \text{ literals, same sign}$$

Examples:

$$\frac{ArthritisPat(x) \vee Affects(f(x), x) \vee ArthritisPat(g(a))}{Affects(f(g(a)), g(a)) \vee ArthritisPat(g(a))} \quad \{x \mapsto g(a)\}$$

$$\frac{Affects(x, John) \vee Affects(JRA, y)}{Affects(JRA, John)} \quad \{x \mapsto JRA, y \mapsto John\}$$

$$\frac{\neg JuvDisease(h(g(f(x), a))) \vee \neg JuvDisease(h(g(y, z)))}{\neg JuvDisease(h(g(f(x), a)))} \quad \{y \mapsto f(x), z \mapsto a\}$$

Recall FOL Resolution Procedure

procedure Sat($\mathcal{S}$)
    repeat
        for all clauses C_1, C_2 in $\mathcal{S}$ **do**
            $\mathcal{S} := \mathcal{S} \cup \text{resolve}(C_1, C_2)$
        end for
    until No new clause can be added to $\mathcal{S}$ or $\Box \in \mathcal{S}$
    If $\Box \in \mathcal{S}$ **return** false
    return true
end procedure

Function resolve(C_1, C_2) applies FO resolution in all possible ways, and then applies factoring in all possible ways.

Recall FOL Resolution Procedure

procedure Sat($\mathcal{S}$)
    repeat
        for all clauses C_1, C_2 in $\mathcal{S}$ **do**
            $\mathcal{S} := \mathcal{S} \cup \text{resolve}(C_1, C_2)$
        end for
    until No new clause can be added to $\mathcal{S}$ or $\square \in \mathcal{S}$
    If $\square \in \mathcal{S}$ **return** false
    return true
end procedure

Function resolve(C_1, C_2) applies FO resolution in all possible ways, and then applies factoring in all possible ways.

Wait ... in all possible ways?

Resolution with Free Selection

Resolution with free selection: a complete strategy

- Calculus parameterised by a Selection Function S
- S assigns to each Horn clause C a non-empty subset of its literals:
  - $S(C)$ contains the single positive literal, OR
  - $S(C)$ contains a subset of negative literals
- Restrict resolution such that we only resolve on selected literals

We are free to design the selection function ourselves:

If we satisfy the basic constraints, completeness is guaranteed.

Resolution with Free Selection

A reasonable selection function:

- Select the set of all negative literals in each clause
- If there is no negative literal, select the (unique) positive literal

As usual, we prepare for resolution (Skolemisation, CNF, clause form)

$$
\begin{aligned}
A(x) \to \exists y.R(x,y) \land B(y) \quad &\leadsto \quad \neg A(x) \lor R(x, f(x)) \\
& \quad \neg A(x) \lor B(f(x)) \\
B(x) \to C(x) \quad &\leadsto \quad \neg B(x) \lor C(x) \\
R(x,y) \land C(y) \to D(y) \quad &\leadsto \quad \neg R(x,y) \lor \neg C(y) \lor D(x) \\
A(a) \quad &\leadsto \quad A(a)
\end{aligned}
$$

We now want to see whether $D(a)$ follows ...

Resolution with Free Selection

$$\neg A(x) \lor R(x, f(x)) \quad (1)$$
$$\neg A(x) \lor B(f(x)) \quad (2)$$
$$\neg B(x) \lor C(x) \quad (3)$$
$$\neg R(x, y) \lor \neg C(y) \lor D(x) \quad (4)$$
$$A(a) \quad (5)$$
$$\neg D(a) \quad (6)$$

With this selection, we don't need to resolve (1) and (4)

Observation: This strategy amounts to Unit Resolution

One of the premises of resolution must be a unit clause!

Resolution with Free Selection

$\neg A(x) \lor R(x, f(x))$		(1)
$\neg A(x) \lor B(f(x))$		(2)
$\neg B(x) \lor C(x)$		(3)
$\neg R(x, y) \lor \neg C(y) \lor D(x)$		(4)
$A(a)$		(5)
$\neg D(a)$		(6)
$R(a, f(a))$	(1) + (5)	(7)
$B(f(a))$	(2) + (5)	(8)
$C(f(a))$	(8) + (3)	(9)
$\neg C(f(a)) \lor D(a)$	(7) + (4)	(10)
$D(a)$	(9) + (10)	(11)
$\square$	(11) + (6)	(12)

Resolution with Free Selection

We still have termination problems ...

$$A(x) \to \exists y.R(x,y) \land A(y) \quad \rightsquigarrow \quad \neg A(x) \lor R(x,f(x))$$
$$\neg A(x) \lor A(f(x))$$
$$A(a) \quad \rightsquigarrow \quad A(a)$$

$$\neg A(x) \lor R(x,f(x))$$
$$\neg A(x) \lor A(f(x))$$
$$A(a)$$
$$R(a,f(a))$$
$$A(f(a))$$
$$R(a,f(f(a)))$$
$$A(f(f(a)))$$
$$\dots$$

Resolution with Free Selection

We still have termination problems . . .

$$A(x) \rightarrow \exists y.R(x,y) \wedge A(y) \quad \rightsquigarrow \quad \neg A(x) \vee R(x,f(x))$$
$$\neg A(x) \vee A(f(x))$$
$$A(a) \quad \rightsquigarrow \quad A(a)$$

$\neg A(x) \vee R(x,f(x))$

$\neg A(x) \vee A(f(x))$

$A(a)$

$R(a,f(a))$

$A(f(a))$

$R(a,f(f(a)))$

$A(f(f(a)))$

$\ldots$

Theorem

Unsatisfiability and fact entailment over existential rules are undecidable (semi-decidable).

That is, as difficult as checking unsatisfiability in FOL.

Datalog

To achieve decidability we need to sacrifice expressivity.

Datalog: The quintessential rule-based KR language

$$\forall\vec{x}.\forall\vec{z}.(\varphi(\vec{x},\vec{z}) \to \psi(\vec{x})) \qquad \textit{Rule}$$

$$\forall\vec{x}.\forall\vec{z}.(\varphi(\vec{x},\vec{z}) \to \bot) \qquad \bot\textit{–Rule}$$

$$P(\vec{a}) \qquad \textit{Fact}$$

$\varphi(\vec{x},\vec{z})$ and $\psi(\vec{x})$: conjunctions of function-free atoms

We can still express

$$\forall x.(\forall y.(\forall z.(\mathit{fatherOf}(x,y) \land \mathit{brotherOf}(x,z) \to \mathit{uncleOf}(z,y))))$$

$$\forall x.(\forall y.(\mathit{Affects}(x,y) \to \mathit{Person}(y)))$$

But, we can no longer express

$$\forall x.(\mathit{JuvDisease}(x) \to \exists y.(\mathit{Affects}(x,y) \land \mathit{Child}(y)))$$

Decidability of Entailment

Theorem

Fact entailment in Datalog is decidable.

Decidability follows directly from Herbrand's theorem

- Our problem reduces to unsatisfiability of $\mathcal{S} = \mathcal{R} \cup \mathcal{F} \cup \{\neg P\}$
- $\mathcal{R} \cup \mathcal{F} \cup \{\neg P\}$ is a set of clauses without function symbols so Herbrand universe finite
- Gilmore's FOL unsatisfiability algorithm terminates.

Decidability of Entailment

Our algorithm is an adaptation of Gilmore's when Herbrand universe is finite

1: **procedure** Datalog-Gil($\langle \mathcal{R}, \mathcal{F} \rangle, P$)
2: Compute Herbrand Universe U
3: $\mathcal{R}' :=$ ground($\mathcal{R}, U$)
4: **return** Horn-Prop($\langle \mathcal{R}', \mathcal{F} \rangle, P$)
5: **end procedure**

Subroutine Horn-Prop solves entailment problem for Horn PL

Complexity Considerations

$$\forall x.(\forall y.(\forall z.(\mathit{fatherOf}(x,y) \land \mathit{brotherOf}(x,z) \to \mathit{uncleOf}(z,y))))$$

$$\mathit{fatherOf}(\mathit{John},\mathit{Mary})$$

$$\mathit{brotherOf}(\mathit{John},\mathit{Peter})$$

Herbrand universe: constants in $\langle \mathcal{R}, \mathcal{F} \rangle$

$$U = \{\mathit{John},\mathit{Mary},\mathit{Peter}\}$$

Grounding leads to exponential size set of propositional clauses

$$\mathit{fatherOf}(\mathit{John},\mathit{John}) \land \mathit{brotherOf}(\mathit{John},\mathit{John}) \to \mathit{uncleOf}(\mathit{John},\mathit{John})$$

$$\mathit{fatherOf}(\mathit{John},\mathit{Mary}) \land \mathit{brotherOf}(\mathit{John},\mathit{Mary}) \to \mathit{uncleOf}(\mathit{Mary},\mathit{Mary})$$

$$\mathit{fatherOf}(\mathit{John},\mathit{Peter}) \land \mathit{brotherOf}(\mathit{John},\mathit{Peter}) \to \mathit{uncleOf}(\mathit{Peter},\mathit{Peter})$$

and so on

Size of the grounding grows as $\mathcal{O}(c^v)$, where

- c is the max. number of constants in facts.
- v is the max. number of variables in rules.

Complexity Considerations

Propositional entailment in Horn PL can be decided in polynomial time.

Overall process takes exponential time (because of grounding).

Theorem

Fact entailment in Datalog is decidable in ExpTime.

In fact, the problem is also ExpTime-hard (beyond this course).

$\rightsquigarrow$ Naive grounding algorithm is worst-case optimal.

Practical Considerations

From a practical point of view, we can do much better:

- Avoid computing the grounding upfront
- Instantiate variables to constants "on the fly"

We develop two resolution-based strategies:

1. Forward chaining:

   Start from facts and instantiate rules to derive new facts whenever possible until goal is derived

2. Backward chaining:

   Start from goal and proceed "backwards" to derive the empty clause

Both strategies can be seen as Resolution with Free Selection.

Forward Chaining (Example)

Start from facts and instantiate rules to derive new facts whenever possible until goal (or $\Box$) is derived

$$\forall x.(\mathit{JuvArthritis}(x) \to \mathit{JuvDisease}(x)) \tag{13}$$

$$\forall x.(\forall y.(\mathit{JuvDisease}(x) \land \mathit{Affects}(x, y) \to \mathit{Child}(y))) \tag{14}$$

$$\mathit{JuvArthritis}(\mathit{JRA}) \tag{15}$$

$$\mathit{Affects}(\mathit{JRA}, \mathit{John}) \tag{16}$$

Match existing facts to rule bodies to derive new facts.

From Fact (15) and Rule (13) we obtain the following by unit resolution

$$\mathit{JuvDisease}(\mathit{JRA}) \tag{17}$$

From Facts (17) and (16) and Rule (14), derive goal and stop.

$$\mathit{Child}(\mathit{John})$$

Forward Chaining and Resolution

$\mathcal{S}_{fw}$: select all negative literals in clauses, and the (unique) positive literal if the clause does not have negative literals.

$$\neg JuvArthritis(x) \lor JuvDisease(x)$$
$$JuvArthritis(JRA)$$

We obtain the following by resolution:

$$JuvDisease(JRA)$$

Forward Chaining and Resolution

$\mathcal{S}_{fw}$: select all negative literals in clauses, and the (unique) positive literal if the clause does not have negative literals.

Deriving a new fact by matching other facts to a rule may require several resolution steps (Hyperresolution).

$$\neg JuvDisease(x) \lor \neg Affects(x, y) \lor Child(y)$$
$$Affects(JRA, John)$$
$$JuvDisease(JRA)$$

We obtain the following by resolution:

$$\neg JuvDisease(JRA) \lor Child(John)$$
$$Child(John)$$

In forward chaining, we do both steps in one.

Forward Chaining

procedure Forward($\langle \mathcal{R}, \mathcal{F} \rangle$, P)

  $\mathcal{F}' := \mathcal{F}$

  repeat

    for each rule $R = \neg B_1 \vee \neg B_2 \vee \ldots, \vee \neg B_n \vee H \in \mathcal{R}$ **do**

      if $\{D_1, \ldots, D_n\} \subseteq \mathcal{F}'$ such that B_i unifies with D_i **then**

        $\theta :=$ Unify($\{B_1 \doteq D_1, \ldots, B_n \doteq D_n\}$)

        $\mathcal{F}' := \mathcal{F}' \cup \{H\theta\}$

      end if

    end for

  until No new atom can be added to $\mathcal{F}'$ or $P \in \mathcal{F}'$ or $\Box \in \mathcal{F}'$

  if $P \in \mathcal{F}'$ or $\Box \in \mathcal{F}'$ **then**

    return true

  else

    return false

  end if

end procedure

Backward Chaining (Example)

Check whether following rules and facts imply $\mathit{Child}(\mathit{John})$:

$$\forall x.(\mathit{JuvArthritis}(x) \to \mathit{JuvDisease}(x)) \tag{18}$$

$$\forall x.(\forall y.(\mathit{JuvDisease}(x) \land \mathit{Affects}(x, y) \to \mathit{Child}(y))) \tag{19}$$

$$\mathit{JuvArthritis}(\mathit{JRA}) \tag{20}$$

$$\mathit{Affects}(\mathit{JRA}, \mathit{John}) \tag{21}$$

Match "goal" $\mathit{Child}(\mathit{John})$ to rule heads and facts to derive new goals.

To prove $\mathit{Child}(\mathit{John})$, by Rule (19) it is sufficient to show

$$\mathit{JuvDisease}(x) \quad \text{and} \quad \mathit{Affects}(x, \mathit{John})$$

Then, by Fact (21), it would be sufficient to show $\mathit{JuvDisease}(\mathit{JRA})$.
Another possibility is to use Rule (18) and get the following sub-goals

$$\mathit{JuvArthritis}(x) \quad \text{and} \quad \mathit{Affects}(x, \mathit{John})$$

And so on ...

Backward Chaining (Example)

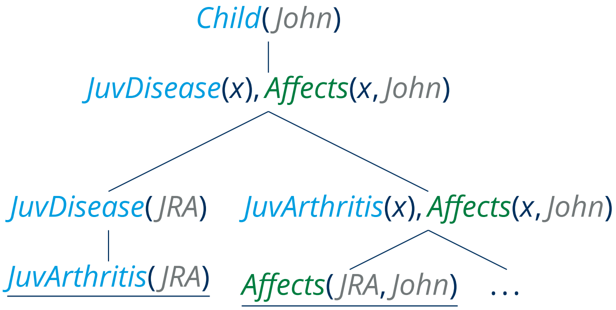
We can represent this kind of backwards reasoning in an AND-OR tree:

$\forall x.(JuvArthritis(x) \to JuvDisease(x))$

$\forall x.(\forall y.(JuvDisease(x) \land Affects(x, y) \to Child(y)))$

$JuvArthritis(JRA)$

$Affects(JRA, John)$



Backward Chaining and Resolution

$\mathcal{S}_{bw}$: select the unique positive literal in clauses, and
all negative literals if the clause does not have positive literals.

Matching the goal to a rule head or a fact corresponds to one resolution step.

$$\frac{\neg JuvDisease(x) \lor \neg Affects(x,y) \lor Child(y) \qquad \neg Child(John)}{\neg JuvDisease(x) \lor \neg Affects(x,John)}$$

Termination Issues

Resolution with free selection may not terminate with $\mathcal{S}_{bw}$.

Example: Show that John is a Scientist.

$$\neg worksWith(x, y) \lor \neg Scientist(y) \lor Scientist(x) \quad (22)$$

$$worksWith(John, Mary) \quad (23)$$

$$\neg Scientist(John) \quad (24)$$

We start resolving on selected atoms:

$$\neg worksWith(John, y) \lor \neg Scientist(y) \qquad (22) + (24) \quad (25)$$

$$\neg worksWith(John, y_1) \lor \neg worksWith(y_1, y_2) \lor \neg Scientist(y_2) \qquad (22) + (25) \quad (26)$$

$$\ldots$$

Keep on generating clauses with chains of *worksWith* atoms of increasing length (variable proliferation).

Thus, the backward chaining tree can have infinite branches.

Other Considerations

Implementing Forward and Backward chaining efficiently is non-trivial:

- Forward chaining: set of deduced facts might get huge
- Backward chaining: recursion may be too deep or search tree too wide.

There are many ways to optimise these algorithms

Semi-naive evaluation, Magic sets, . . .

But, this is beyond the scope of this course.

There are many optimised systems that implement forward/backward chaining.

The KR languages we have described are related to:

- Databases: Datalog query language, and deductive databases
- Logic programming: Prolog