Artificial Intelligence

11. Predicate Logic Reasoning, Part II: Reasoning

And Now: How to Actually *Think* in Terms of Predicates

Jörg Hoffmann, Daniel Fiser, Daniel Höller, Sophia Saller

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Summer Term 2022

Agenda

Let's *Reason* About Blocks, Baby ...

I asked: What do you see here?

You said: "All blocks are red"; "All blocks are on the table"; "A is a block".

I said: From propositional logic "AllBlocksAreRed" and "isBlockA", we can't conclude that A is red, because these are treated like atomic statements, ignoring their inner structure ("all blocks", "is a block").

Predicate Logic: "$\forall x[Block(x) \rightarrow Red(x)]$"; "$Block(A)$".

$\rightarrow$ All fine, but how *do* we conclude in PL1 that A is red?

Reminder: Our Agenda for This Topic

→ Our treatment of the topic “Predicate Logic Reasoning” consists of Chapters 10 and 11.

- **Chapter 10:** Basic definitions and concepts; normal forms.

  → Sets up the framework and basic operations.

- **This Chapter:** Compilation to propositional reasoning; unification; lifted resolution.

  → Algorithmic principles for reasoning about predicate logic.

Our Agenda for This Chapter

- **Reduction to Propositional Reasoning:** Can we reduce PL1 reasoning to propositional reasoning?

  → Yes we can! (But it's tricky, and involves generating huge grounded encodings . . .)

- **Substitutions, and Unification:** What basic operations are required to avoid grounding everything out?

  → Specifies how to instantiate variables with terms.

- **PL1 Resolution:** How do we reason directly at PL1 level?

  → The foundational procedure for doing so.

- **On Criminals and Cats:** And now, in practice?

  → Spells out some examples.

Reasoning About PL1 *Via Propositional Logic?*

What for do we need PL1, then?

- "First-order logic as syntactic sugar for propositional logic."
- Remember all these propositions in the Wumpus world?
- Anyway, it's of course not that easy in general (cf. slide 12).

How?

Reasoning About PL1 Via Propositional Logic

1. Bring into Skolem normal form (SNF).
2. Generate (the finite subsets of) the Herbrand expansion (up next).
3. Use propositional reasoning.

$\rightarrow$ Apply DPLL, clause learning, ...

- Herbrand expansion may be very large (infinite, in general).
- Still, this often works well in practice.

Herbrand Universe

We assume: Skolem normal form. (We don't require φ to be in CNF.)

$$\underbrace{\forall x_1 \forall x_2 \forall x_3 \ldots \forall x_n}_{\text{universal prefix}} \; + \; \underbrace{\varphi}_{\text{(quantifier-free) matrix}}$$

Notation: For any (finite) set θ^* of PL1 formulas, denote by $CF(\theta^*)$ the set of constant symbols, and of function symbols (arity ≥ 1), occuring in θ^*. If no constant symbol occurs in θ^*, we add a new such symbol c into $CF(\theta^*)$.

Definition (Herbrand Universe). *Let θ^* be a set of PL1 formulas in SNF. Then the Herbrand universe $HU(\theta^*)$ over θ^* is the set of all ground terms that can be formed from $CF(\theta^*)$.*

Example: $\theta^* = \{\forall x[\neg Dog(x) \vee Chases(x, f(x))]\}$

$CF(\theta^*) = \{c, f\}$; $HU(\theta^*) = \{c, f(c), f(f(c)), \ldots\}$.

Herbrand Expansion

Definition (Herbrand Expansion). *Let θ^* be a set of PL1 formulas in SNF. The Herbrand expansion $HE(\theta^*)$ is defined as:*

$$HE(\theta^*) = \{\varphi\frac{x_1}{t_1}, \dots, \frac{x_n}{t_n} \mid (\forall x_1 \dots \forall x_n \varphi) \in \theta^*, t_i \in HU(\theta^*)\}$$

$\rightarrow$ Instantiate each matrix φ with all terms from $HU(\theta^*)$. As $HE(\theta^*)$ contains ground atoms only, it can be interpreted as propositional logic.

Example: $\theta^* = \{\forall x[\neg Dog(x) \vee Chases(x, f(x))]\}$
$\rightarrow HE(\theta^*) = \{[\neg Dog(c) \vee Chases(c, f(c))],$
$[\neg Dog(f(c)) \vee Chases(f(c), f(f(c)))], \dots\}$.

Theorem (Herbrand). *Let θ^* be a set of PL1 formulas in SNF. Then θ^* is satisfiable iff $HE(\theta^*)$ is satisfiable.* (Proof omitted.)

$\rightarrow$ **Observe:** Without function symbols, the Herbrand expansion is finite, and PL1 reasoning is equivalent to propositional reasoning.

When Herbrand Reasons About Blocks . . .

Example: KB $= \{\forall x[Block(x) \rightarrow Red(x)], Block(A)\}$

Want: Deduce that A is red, i.e., KB $\models \varphi$ for $\varphi := Red(A)$.

Deduction: $\theta :=$ KB $\cup \{\neg\varphi\}$ is unsatisfiable iff KB $\models \varphi$.

Skolem normal form θ^*: $\{\forall x[\neg Block(x) \vee Red(x)], Block(A), \neg Red(A)\}$

Herbrand universe: $HU(\theta^*) = \{A\}$

Herbrand expansion: $HE(\theta^*) = \{[\neg Block(A) \vee Red(A)], Block(A), \neg Red(A)\}$

Proof of $Red(A)$: E.g., unit propagation yields the empty clause.

Herbrand: The Infinite Case

→ **Recall:** Without function symbols, the Herbrand expansion is finite, and PL1 reasoning is equivalent to propositional reasoning.

→ But what if there *are* function symbols?

Theorem (Compactness of Propositional Logic). *Any set θ of propositional logic formulas is unsatisfiable if and only if at least one finite subset of θ is unsatisfiable.* (Proof omitted.)

Method: Enumerate all finite subsets θ_1 of the Herbrand expansion $HE(\theta^*)$, and test propositional satisfiability of θ_1. θ is unsatisfiable if and only if one of the θ_1 is. Only ... which θ_1 will do the job?

→ If the Herbrand expansion is *infinite*, to show unsatisfiability ($=$ to prove that some property does indeed follow from the KB), we must somehow choose a "relevant" finite subset thereof.

→ Direct PL1 reasoning ameliorating this caveat: later in this chapter.

Herbrand, Infinite Case: What If θ is Satisfiable?

Theorem (A). *The set of unsatisfiable PL1 formulas is recursively enumerable.*

Proof. Enumerate all PL1 formulas φ. Incrementally for all of these in parallel, enumerate all finite subsets θ_1 of the Herbrand expansion $HE(\varphi^*)$. Test propositional satisfiability of each θ_1. By compactness of propositional logic, if $HE(\varphi^*)$ is unsatisfiable then one of the θ_1 is.

Theorem (B). *It is undecidable whether a PL1 formula is satisfiable.* (Proof omitted.)

Corollary. *The set of satisfiable PL1 formulas is not recursively enumerable.* (Proof: Else, with Theorem (A), PL1 satisfiability would be decidable, in contradiction to Theorem (B).)

$\rightarrow$ If a PL1 formula is unsatisfiable, then we can confirm this. Otherwise, we might end up in an infinite loop.

Questionnaire

Question!

What is the Herbrand universe $HU(\theta^*)$ of
$\theta^* = \{\forall x[Equals(x, succ(f(x)))], \forall x \neg Equals(1, succ(x))\}$**?**

(A): $\{1\}$.

(B): $\{1, f, succ\}$.

(C): $\{1, succ(1), succ(succ(1)), \dots\}$.

(D): $\{1, f(1), succ(1), succ(f(1)), f(succ(1)), \dots\}$.

→ (A): No, we need the entire set of terms.

→ (B): No, we need terms not just function symbols.

→ (C): No, we need *all* possible terms.

→ (D): Yes: Enumerate all ways in which functions can be applied to constant symbols.

Questionnaire, ctd.

Question!

Is the Herbrand expansion of $\theta^* = \{\forall x[Equals(x, succ(f(x)))], \forall x \neg Equals(1, succ(x))\}$ **satisfiable?**

(A): Yes. (B): No.

$\rightarrow$ The correct answer is "No".

The easy way: "Every x is the successor of some other number" (namely of $f(x)$) together with "1 is not the successor of any other number" is not satisfiable. The same is, then, true of the Herbrand expansion simply by Herbrand's theorem (slide 10).

The hard way: *Pretend you're a computer.* Choose a finite unsatisfiable subset of $HE(\theta^*)$. $\rightarrow$ Suggestions for a finite subset of $HU(\theta^*)$?

$\rightarrow$ Turns out we can use $\{1, f(1)\}$. Matrix of the first formula, instantiated with 1, gives $Equals(1, succ(f(1)))$. Matrix of the second formula, instantiated with $f(1)$, gives $\neg Equals(1, succ(f(1)))$. Done with a single resolution step.

Towards PL1 Resolution

Clausal normal form:

universal prefix + disjunction of literals

$$\forall x_1 \forall x_2 \forall x_3 \dots \forall x_n (l_1 \vee \dots \vee l_n)$$

$\rightarrow$ Written $\{l_1, \dots, l_n\}$.

$\rightarrow$ The quantifiers are omitted in the notation!

Example: $\{\{Nat(s(x)), \neg Nat(x)\}, \{Nat(1)\}\}$

We want to somehow apply/adapt the resolution rule:

$$\frac{C_1 \dot\cup \{l\}, C_2 \dot\cup \{\bar{l}\}}{C_1 \cup C_2}$$

Towards PL1 Resolution, ctd.

What about this:

→ $\{\{Nat(s(1)), \neg Nat(1)\}, \{Nat(1)\}\} \models \{Nat(s(1))\}$? Yes.

→ And $\{\{Nat(s(1)), \neg Nat(1)\}, \{Nat(1)\}\} \vdash \{Nat(s(1))\}$? Yes, if we allow to resolve PL1 literals whose atoms are identical.

And what about this?

→ $\{\{Nat(s(x)), \neg Nat(x)\}, \{Nat(1)\}\} \models \{Nat(s(1))\}$? Yes, due to the universal quantification (clausal normal form, cf. previous slide).

→ But $\{\{Nat(s(x)), \neg Nat(x)\}, \{Nat(1)\}\} \vdash \{Nat(s(1))\}$? No, the atoms aren't identical.

→ We need a way to *make* them identical: unification! Based on the notion of substitution. Here: $\{\frac{x}{1}\}$.

→ Applying a substitution *specializes* the clause, which is valid because the variables are universally quantified.

Substitutions

Definition (Substitution). *A substitution* $s = \{\frac{x_1}{t_1}, \dots, \frac{x_n}{t_n}\}$ *is a function that substitutes variables* x_i *for terms* t_i*, where* $x_i \neq t_i$ *for all* i*. Applying substitution* s *to a formula* φ *yields the expression* φs*, which is* φ *with all occurrences of* x_i *simultaneously replaced by* t_i*.*

Example: For $s = \{\frac{x}{y}, \frac{y}{h(a,b)}\}$, $P(x, y)s = P(y, h(a, b))$.

→ Variable instantiation and renaming, as used in the prenex and Skolem transformations as well as in the Herbrand expansion, are special cases of substitution.

→ Here, we will use substitution on atoms P only. Applying s to a set of atoms means to apply it to each one.

Remember: $x, y, z, v, w, \dots$: variables; $a, b, c, d, e, \dots$: constants.

Substitution Examples

Remember: $x, y, z, v, w, \ldots$: variables; $a, b, c, d, e, \ldots$: constants.

Examples: Can we apply a substitution to $P(x, f(y), b)$ so that it becomes:

1. $P(z, f(w), b)$: Yes: $s = \{\frac{x}{z}, \frac{y}{w}\}$
2. $P(x, f(a), c)$? No; $\frac{b}{c}$ not possible because b is a constant, not a variable.
3. $P(y, f(h(a, b, w)), b)$? Yes: $s = \{\frac{x}{y}, \frac{y}{h(a,b,w)}\}$
4. $Q(x, f(y), b)$? No. The predicate symbols must be the same.
5. $P(x, f(f(y)), b)$? Yes: $s = \{\frac{y}{f(y)}\}$.

Composing Substitutions

Definition (Composition). *Given substitutions s_1 and s_2, by $s_1 s_2$ we denote the composed substitution, a single substitution whose outcome is identical to $s_2 \circ s_1$.*

Example: With $s_1 = \{\frac{z}{g(x,y)}, \frac{v}{w}\}$ and $s_2 = \{\frac{x}{a}, \frac{y}{b}, \frac{w}{v}, \frac{z}{d}\}$, we have $P(x, y, z, v)s_1 s_2 = P(a, b, g(a, b), v)$.

How to obtain $s_1 s_2$ given s_1 and s_2?

(i) Apply s_2 to the replacement terms t_i in s_1.

(ii) For any variable x_i replaced by s_2 but not by s_1, apply the respective variable/term pair $\frac{x_i}{t_i}$ of s_2.

(iii) Remove any pairs of variable x and term t where $x = t$.

Example: $\{\frac{z}{g(x,y)}, \frac{v}{w}\}\{\frac{x}{a}, \frac{y}{b}, \frac{w}{v}, \frac{z}{d}\} = \{\frac{z}{g(a,b)}, \frac{x}{a}, \frac{y}{b}, \frac{w}{v}\}$.

Properties of Substitutions

For any formula φ and substitutions s_1, s_2, s_3:

→ $(\varphi s_1)s_2 = \varphi(s_1 s_2)$ by definition (composing functions).

→ $(s_1 s_2)s_3 = s_1(s_2 s_3)$ by definition (composing functions).

→ $s_1 s_2 = s_2 s_1$? No (not commutative), e.g. $\varphi = Dog(x)$, $s_1 = \{\frac{x}{Lassie}\}$, $s_2 = \{\frac{x}{Garfield}\}$.

(And by the way:)

Proposition. *A substitution $s = \{\frac{x_1}{t_1}, \dots, \frac{x_n}{t_n}\}$ is idempotent, i.e., $\varphi ss = \varphi s$ for all φ, iff t_i does not contain x_j for $1 \leq i, j \leq n$.*

Proof. "$\Leftarrow$": The second application of s does not do anything because all x_i have been removed. "$\Rightarrow$": if t_i contains x_j then the second application of s replaces x_j with $t_j \neq x_j$.

Example: For $s = \{\frac{x}{y}, \frac{y}{h(a,b)}\}$,
$P(x, y)s = P(y, h(a, b)) \neq P(x, y)ss = P(h(a, b), h(a, b))$.

Unification

Definition (Unifier). *We say that a substitution s is a unifier for a set of atoms $\{P_1, \dots, P_k\}$ if $P_i s = P_j s$ for all i, j.*

Notation: We'll usually write $\{P_i\}$ for $\{P_1, \dots, P_k\}$.

Example: $\{P(x, f(y, z), b), P(x, f(b, w), b)\}$

$\rightarrow s = \{\frac{y}{b}, \frac{z}{w}, \frac{x}{h(a,b)}\}$? Yes. But not "the best" one.

$\rightarrow s = \{\frac{y}{b}, \frac{z}{w}\}$? Yes. This is a most general unifier (MGU):

Definition (MGU). *We say that a unifier g of $\{P_i\}$ is an MGU if, for any unifier s of $\{P_i\}$, there exists a substitution s' s.t. $\{P_i\}s = \{P_i\}gs'$.*

$\rightarrow$ If any unifier exists, then an idempotent MGU exists.

$\rightarrow$ We'll next introduce an algorithm that finds it.

Disagreement Set

Definition (Disagreement Set). *The disagreement set $D(\{t_i\})$ of a set of terms $\{t_i\}$ is the leftmost and outermost set of sub-terms where some of the t_i disagree.*[1]
The disagreement set $D(\{P_i\})$ of a set of atoms $\{P_i\}$ is the disagreement set $D(\{t_i\})$ where $\{t_i\}$ is the term set at the leftmost argument for which some of the P_i disagree.

Examples:

$\{P(x, c, f(y)), P(x, z, z)\}$: $\{c, z\}$

$\{P(x, a, f(y)), P(y, a, f(y))\}$: $\{x, y\}$

$\{P(v, f(z), g(w)), P(v, f(z), g(f(z)))\}$: $\{w, f(z)\}$

$\{P(v, f(z), g(w)), P(v, f(z), g(f(z))), P(v, f(z), f(x))\}$: $\{g(w), g(f(z)), f(x)\}$

[1] Formally for a set of two terms $\{t, t'\}$:

$$D(\{t,t'\}) := \begin{cases} \{t,t'\} & \text{at least one of } t \text{ and } t' \text{ is a variable or constant} \\ \{t,t'\} & t = f(t_1,\dots,t_n), t' = g(t'_1,\dots,t'_m), f \neq g \\ D(\{t_i,t'_i\}) & \text{otherwise, where } i \text{ is minimal with } t_i \neq t'_i \end{cases}$$

Unification Algorithm: What We Can *Not* Do

Example: Can we unify $\{P(x, y, b), P(x, f(y), b)\}$?

No. Whichever way we replace y on the left-hand side, the same change will appear within "$f(y)$" on the right-hand side. So the two will be different again. E.g., consider $s = \{\frac{y}{f(y)}\}$: $P(x, y, b)s = P(x, f(y), b)$ $\neq P(x, f(f(y)), b) = P(x, f(y), b)s$.

$\rightarrow$ If the only way to unify $\{P_i\}$ is to unify a variable x with a term t that contains x, then $\{P_i\}$ cannot be unified.

Unification Algorithm

Theorem. *The following algorithm succeeds if and only if there exists a unifier for* $\{P_i\}$*. In the positive case, the algorithm returns an idempotent MGU of* $\{P_i\}$*.* (Proof omitted.)

$k \leftarrow 0$, $T_k = \{P_i\}$, $s_k = \{\}$;
while T_k is not a singleton **do**
  Let D_k be the disagreement set of T_k;

/* if t_k contains x_k then unification is impossible, cf. slide 25 */
  Let $x_k, t_k \in D_k$ be a variable and term s.t. t_k does not contain x_k;
  if such x_k, t_k do not exist **then exit** with output "failure";

  $s_{k+1} \leftarrow s_k\{\frac{x_k}{t_k}\}$; /* t_k does not contain any of $x_1, \dots, x_k$ */

  $T_{k+1} \leftarrow T_k\{\frac{x_k}{t_k}\}$; /* x_k does not occur in T_{k+1} */

  $k \leftarrow k + 1$;
endwhile
exit with output s_k;

Unification Algorithm: An Example

$$\{P(x, f(y), y), P(z, f(b), b)\}$$

$D_0 = \{x, z\}$
$s_1 := \{\frac{x}{z}\}$
$T_1 = \{P(z, f(y), y), P(z, f(b), b)\}$

$D_1 = \{y, b\}$
$s_2 := s_1\{\frac{y}{b}\} = \{\frac{x}{z}, \frac{y}{b}\}$
$T_2 = \{P(z, f(b), b), P(z, f(b), b)\} = \{P(z, f(b), b)\}$

$\rightarrow T_2$ is a singleton. Return s_2.

Questionnaire

Question!

Can $\{Knows(John, x), Knows(x, Elizabeth)\}$ **be unified?**

(A): Yes (B): No

→ No. We would have to substitute two different constants for x. Algorithm trace: $D_0 = \{John, x\}$ $s_1 := \{\frac{x}{John}\}$ $T_1 = \{Knows(John, John), Knows(John, Elizabeth)\}$; $D_1 = \{John, Elizabeth\}$ which does not contain a variable, stop with "failure".

Question!

What about $\{Knows(John, x), Knows(y, Elizabeth)\}$**?**

(A): Yes (B): No

→ Yes. Algorithm trace: $D_0 = \{John, y\}$ $s_1 := \{\frac{y}{John}\}$ $T_1 = \{Knows(John, x), Knows(John, Elizabeth)\}$; $D_1 = \{x, Elizabeth\}$ $s_2 := s_1\{\frac{x}{Elizabeth}\}$ $T_2 = \{Knows(John, Elizabeth)\}$. T_2 is a singleton. Return s_2.

→ Note: Here we have standardized the variables apart. (Remember: Last step of transformation to clausal normal form, Chapter 10.)

PL1 Resolution: Setup

We assume: Clausal normal form, variables standardized apart.

universal prefix + disjunction of literals

$$\forall x_1 \forall x_2 \forall x_3 \dots \forall x_n (l_1 \vee \dots \vee l_n)$$

$\rightarrow$ Written $\{l_1, \dots, l_n\}$.

Example: $\{\{Nat(s(x)), \neg Nat(x)\}, \{Nat(1)\}\}$

Terminology and Notation

- A literal l is an atom or the negation thereof; the negation of a literal is denoted $\overline{l}$ (e.g., $\overline{\neg Q} = Q$).
- A clause C is a set (=disjunction) of literals.
- Our input is a set Δ of clauses.
- The empty clause is denoted $\Box$.
- A calculus is a set of inference rules.

PL1 Resolution: Setup, ctd.

Derivations: *We say that a clause C can be derived from Δ using calculus $\mathcal{R}$, written $\Delta \vdash_{\mathcal{R}} C$, if (starting from Δ) there is a sequence of applications of rules from $\mathcal{R}$, ending in C.*

$\rightarrow$ In contrast to propositional resolution, we will consider here three different resolution calculi $\mathcal{R}$.

Soundness: *A calculus $\mathcal{R}$ is sound if $\Delta \vdash_{\mathcal{R}} C$ implies $\Delta \models C$.*

Completeness: *A calculus $\mathcal{R}$ is refutation-complete if $\Delta \models \bot$ implies $\Delta \vdash_{\mathcal{R}} \Box$, i.e., if Δ is unsatisfiable then we can derive the empty clause.*

- Together: Δ is unsatisfiable iff we can derive the empty clause.
- Propositional resolution is sound & refutation-complete for propositional Δ.

Deduction as Proof by Contradction

To decide whether KB $\models \varphi$, decide satisfiability of $\psi :=$ KB $\cup \{\neg\varphi\}$. ψ is unsatisfiable iff KB $\models \varphi$.

Reminder: Propositional Resolution

Definition (Propositional Resolution). *Resolution uses the following inference rule (with exclusive union $\dot\cup$ meaning that the two sets are disjoint):*

$$\frac{C_1 \dot\cup \{l\}, C_2 \dot\cup \{\bar{l}\}}{C_1 \cup C_2}$$

If Δ contains parent clauses of the form $C_1 \dot\cup \{l\}$ and $C_2 \dot\cup \{\bar{l}\}$, the rule allows to add the resolvent clause $C_1 \cup C_2$. l and $\bar{l}$ are called the resolution literals.

Example: $\{P, \neg R\}$ resolves with $\{R, Q\}$ to $\{P, Q\}$.

Lemma. *The resolvent follows from the parent clauses.*

Proof. If $I \models C_1 \dot\cup \{l\}$ and $I \models C_2 \dot\cup \{\bar{l}\}$, then I must make at least one literal in $C_1 \cup C_2$ true.

Binary PL1 Resolution

Definition (Binary PL1 Resolution). *Binary PL1 resolution is the following inference rule:*

$$\frac{C_1\dot{\cup}\{P_1\}, C_2\dot{\cup}\{\neg P_2\}}{[C_1 \cup C_2]g}$$

If Δ contains parent clauses of the form $C_1\dot{\cup}\{P_1\}$ and $C_2\dot{\cup}\{\neg P_2\}$, where $\{P_1, P_2\}$ can be unified and g is an MGU thereof, the rule allows to add the resolvent clause $[C_1 \cup C_2]g$. P_1 and $\neg P_2$ are called the resolution literals.

Example: From $\{Nat(s(x)), \neg Nat(x)\}$ and $\{Nat(1)\}$ we can derive $Nat(s(1))$ using the MGU $g = \{\frac{x}{1}\}$.

Lemma (Soundness). *The resolvent follows from the parent clauses.*

Proof. [1. Substitution instantiates a universal clause to a special case.] If I satisfies the parent clauses, then due to the universal quantification it must satisfy the substituted parent clauses; these take the form $C_1\dot{\cup}\{l\}$ and $C_2\dot{\cup}\{\bar{l}\}$.

[2. Same argument as in propositional case.] But then (similar to propositional case), for every assigment to the remaining (universally quantified) variables, I must make at least one literal in $C_1 \cup C_2$ true.

Why Do We Need To Standardize Variables Apart?

Example: $\Delta = \{\{Knows(John, x)\}, \{\neg Knows(x, Elizabeth), King(x)\}\}$

$\rightarrow$ We should be able to conclude that? John is a king.

Unification 1: $\{P_1, P_2\} = \{Knows(John, x), Knows(x, Elizabeth)\}$

$\rightarrow$ Is there a unifier for $\{P_i\}$? No. We would have to substitute two different constants for x. (Cf. slide 28)

Unification 2: $\{P_1, P_2\} = \{Knows(John, x), Knows(y, Elizabeth)\}$

$\rightarrow$ Is there a unifier for $\{P_i\}$? Yes: $\{\frac{x}{Elizabeth}, \frac{y}{John}\}$. (Cf. slide 28)

$\rightarrow$ Standardizing the variables in clauses apart is sometimes necessary to allow unification.

($\rightarrow$ An alternative would be to not use unification, and instead substitute atoms separately to the same outcome; we don't consider this here.)

Questionnaire

Question!

Which are PL1 resolvents of
$\{\neg Chases(x, Garfield), Chases(Lassie, x)\}$ **and** $\{Chases(Bello, y)\}$**?**

(A): $\{Chases(Lassie, Bello)\}$ (B): $\{Chases(Garfield, Bello)\}$

(C): $\square$ (D): $\{Chases(Bello, Garfield)\}$

→ (A): Yes, we can obtain this resolvent with $g = \{\frac{x}{Bello}, \frac{y}{Garfield}\}$.

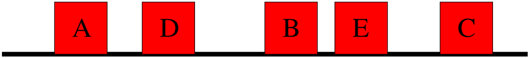
→ (B): No. The only potential resolution literal in the first clause is $\neg Chases(x, Garfield)$; the remaining literal $Chases(Lassie, x)$ can't be instantiated to $Chases(Garfield, Bello)$.

→ (C): No.

→ (D): No, same as (B).

When PL1 Resolution Reasons About Blocks . . .

Example: KB $= \{\forall x[\mathit{Block}(x) \rightarrow \mathit{Red}(x)], \mathit{Block}(A)\}$



Want: Deduce that A is red, i.e., KB $\models \varphi$ for $\varphi := \mathit{Red}(A)$.

Deduction: $\theta :=$ KB $\cup \{\neg\varphi\}$ is unsatisfiable iff KB $\models \varphi$.

Skolem normal form θ^*: $\{\forall x[\neg\mathit{Block}(x) \vee \mathit{Red}(x)], \mathit{Block}(A), \neg\mathit{Red}(A)\}$

Clausal normal form Δ: $\{\{\neg\mathit{Block}(x), \mathit{Red}(x)\}, \{\mathit{Block}(A)\}, \{\neg\mathit{Red}(A)\}\}$

PL1 resolution proof:

$\rightarrow$ Resolve 1st with 2nd clause using $g = \{\frac{x}{A}\}$, yielding $\{\mathit{Red}(A)\}$.

$\rightarrow$ Resolve that clause with 3rd clause using $g = \{\}$, yielding $\Box$.

Where Binary PL1 Resolution Fails

Example: $\Delta = \{\{P(x_1, x_2), P(x_2, x_1)\}, \{\neg P(y_1, y_2), \neg P(y_2, y_1)\}\}$

$\rightarrow$ Is Δ satisfiable? No. Remember that the variables in PL1 clauses are universally quantified. To satisfy Δ, we would (in particular) have to have $P(o, o)$ and $\neg P(o, o)$ for every object o in the universe.

$\rightarrow$ Can we derive $\Box$ with binary PL1 resolution? No. Every derivable clause has the form $\{l(V_1, V_2), l(V_2, V_1)\}$ where $l \in \{P, \neg P\}$, $V_1 \in \{x_1, y_1\}$, and $V_2 \in \{x_2, y_2\}$. E.g., with $\{\frac{y_1}{x_1}, \frac{y_2}{x_2}\}$, we can derive $\{P(x_2, x_1), \neg P(x_2, x_1)\}$. The empty clause is not derivable.

Notation: Define $\mathcal{R}_{Binary} :=$ {binary PL1 resolution}.

Theorem. *The calculus $\mathcal{R}_{Binary}$ is not refutation-complete.*

Proof. See example above.

However, $\mathcal{R}_{Binary}$ is sound:

Theorem. *The calculus $\mathcal{R}_{Binary}$ is sound.* (Proof: Lemma slide 33)

Solution 1: Full PL1 Resolution

→ Allow to unify *several* resolution literals:

Definition (Full PL1 Resolution). *Full PL1 resolution is the following inference rule:*

$$\frac{C_1\dot{\cup}\{P_1^1,\dots,P_1^n\},C_2\dot{\cup}\{\neg P_2^1,\dots,\neg P_2^m\}}{[C_1\cup C_2]g}$$

where $\{P_1^1,\dots,P_1^n,P_2^1,\dots,P_2^m\}$ *can be unified and* g *is an MGU.*

Example: $\Delta=\{\{P(x_1,x_2),P(x_2,x_1)\},\{\neg P(y_1,y_2),\neg P(y_2,y_1)\}\}$

→ Can we derive $\Box$ with full PL1 resolution? Yes, using for example the unifier $g=\{\frac{x_2}{x_1},\frac{y_1}{x_1},\frac{y_2}{x_1}\}$.

Notation: Define $\mathcal{R}_{Full}:=\{$full PL1 resolution$\}$.

Theorem. *The calculus* $\mathcal{R}_{Full}$ *is sound.*

Proof. It suffices to show that, for each application of the rule, the resolvent follows from the parents. That can be shown with the same argument as for binary PL1 resolution (Lemma slide 33).

Solution 2: Binary PL1 Resolution + Factoring

→ Allow to unify literals *within* a clause:

Definition (Factoring). *Factoring is the following inference rule:*

$$\frac{C_1 \dot\cup \{l_1\} \dot\cup \{l_2\}}{[C_1 \cup \{l_1\}]g}$$

where $\{l_1, l_2\}$ can be unified and g is an MGU thereof. $[C_1 \cup \{l_1\}]g$ is called a factor of the parent clause $C_1 \dot\cup \{l_1\} \dot\cup \{l_2\}$.

Example: $\Delta = \{\{P(x_1, x_2), P(x_2, x_1)\}, \{\neg P(y_1, y_2), \neg P(y_2, y_1)\}\}$

→ How can we apply factoring? $\{\frac{x_2}{x_1}\}$ on $\{P(x_1, x_2), P(x_2, x_1)\}$ gives $\{P(x_1, x_1)\}$, $\{\frac{y_2}{y_1}\}$ on $\{\neg P(y_1, y_2), \neg P(y_2, y_1)\}$ gives $\{\neg P(y_1, y_1)\}$. Then we can derive $\Box$ with binary PL1 resolution, using $g = \{\frac{y_1}{x_1}\}$.

Notation: Define $\mathcal{R}_{FactBin} :=$ {binary PL1 resolution,factoring}.

Theorem. *The calculus $\mathcal{R}_{FactBin}$ is sound.*

Proof. Due to the universal quantification, the factor follows from its parent. Done with Lemma slide 33.

What About Completeness? The Lifting Lemma

Lemma (Lifting Lemma). *Let C_1 and C_2 be two clauses with no shared variables, and let C_1^g and C_2^g be ground instances of C_1 and C_2. Say that C^g is a resolvent of C_1^g and C_2^g. Then there exists a clause C such that C^g is a ground instance of C, and:*

(i) *C can be derived from C_1 and C_2 using $\mathcal{R}_{Full}$.*

(ii) *C can be derived from C_1 and C_2 using $\mathcal{R}_{FactBin}$.*

Proof Sketch. The resolution literals (WLOG) $P \in C_1^g$ and $\neg P \in C_2^g$ must be obtainable by grounding $\{P_1^1, \dots, P_1^n\} \subseteq C_1$ with s_1 and $\{\neg P_2^1, \dots, \neg P_2^m\} \subseteq C_2$ with s_2. As C_1 and C_2 share no variables, $s_1 s_2$ is a unifier for $\{P_1^1, \dots, P_1^n, P_2^1, \dots, P_2^m\}$. So an MGU exists and we can apply full PL1 resolution, showing (i). From this, (ii) follows because an application of full PL1 resolution can be simulated using several applications of factoring followed by an application of binary PL1 resolution.

Example: $C_1 = \{P(x_1), P(x_2), R(z)\}$, $C_2 = \{\neg P(y_1), \neg P(y_2), R(z)\}$

E.g., $C_1^g = \{P(o), R(o)\}$ and $C_2^g = \{\neg P(o), R(o)\}$. Then $C^g = \{R(o)\}$; $\{P_1^1, \dots, P_1^n\} = \{P(x_1), P(x_2)\}$, $\{\neg P_2^1, \dots, \neg P_2^m\} = \{\neg P(y_1), \neg P(y_2)\}$; $C = \{R(z)\}$ results from full PL1 resolution with $g = \{\frac{x_2}{x_1}, \frac{y_1}{x_1}, \frac{y_2}{x_1}\}$.

What About Completeness? Proof

Theorem. *The calculi $\mathcal{R}_{Full}$ and $\mathcal{R}_{FactBin}$ are refutation-complete.*

Proof:

$$
\begin{array}{c}
\text{Any set } \theta \text{ of PL1 formulas is representable in clausal form } \Delta. \\
\downarrow \\
\text{Assume } \Delta \text{ is unsatisfiable.} \\
\downarrow \quad \longleftarrow \text{ Herbrand, prop. compactness} \\
\text{Some finite set } \Delta' \text{ of ground instances is unsatisfiable.} \\
\downarrow \quad \longleftarrow \text{ Prop. resolution completeness} \\
\text{Propositional resolution can derive } \Box \text{ from } \Delta'. \\
\downarrow \quad \longleftarrow \text{ Lifting Lemma} \\
\text{Each of } \mathcal{R}_{Full} \text{ and } \mathcal{R}_{FactBin} \text{ can derive } \Box \text{ from } \Delta.
\end{array}
$$

Questionnaire

Question!

Is full PL1 resolution guaranteed to terminate after a finite number of rule applications?

(A): Yes. (B): No.

$\rightarrow$ No. If PL1 resolution were guaranteed to terminate, then it would be a decision procedure for unsatisfiability of PL1 formulas. That problem, however, is only semi-decidable, cf. slide 13.

$\rightarrow$ For illustration, consider these three clauses: (1) $\{\neg P(x), Q(f(x))\}$, (2) $\{\neg Q(y), R(f(y))\}$, (3) $\{\neg R(z), Q(f(z))\}$. There is an infinite sequence of applications of PL1 resolution: (1) and (2) give (4) $\{\neg P(x), R(ff(x))\}$; (3) and (4) give $\{\neg P(x), Q(fff(x))\}$; (2) and (5) give $\{\neg P(x), R(ffff(x))\}$ ...

Binary PL1 Resolution: Examples

Clauses: $\{P(x), Q(f(x))\}$, $\{R(g(x)), \neg Q(f(a))\}$

$\rightarrow$ Standardizing variables apart: $\{P(x), Q(f(x))\}$, $\{R(g(y)), \neg Q(f(a))\}$

$\rightarrow$ MGU: $s = \{\frac{x}{a}\}$

$\rightarrow$ Resolvent: $\{P(a), R(g(y))\}$.

Clauses: $\{P(x, g(c)), Q(x, a)\}$, $\{\neg P(y, g(c)), \neg R(b, z)\}$

$\rightarrow$ Standardizing variables apart: (Nothing to do.)

$\rightarrow$ MGU: $s = \{\frac{x}{y}\}$

$\rightarrow$ Resolvent: $\{Q(y, a), \neg R(b, z)\}$.

Example "Integers"

Formula: $\forall x \exists y (Equals(x, succ(y)))$ ("For every integer x, there is y so that $x = y + 1$")

$\rightarrow$ Is this satisfiable? Yes: E.g., by setting "$Equals$" to be all pairs of objects.

Partial axiomatization of $succ, Equals$:

"1 is not the successor of anybody:" $\forall x \neg Equals(1, succ(x))$

Resolution refutation:

$\forall x \exists y (Equals(x, succ(y))) \mapsto \{Equals(x, succ(f(x)))\}$

$\forall x \neg Equals(1, succ(x)) \mapsto \{\neg Equals(1, succ(x))\} \mapsto \{\neg Equals(1, succ(y))\}$

MGU: D_0: $\{x, 1\}$; s_1: $\{\frac{x}{1}\}$; T_1: $\{Equals(1, succ(f(1))), Equals(1, succ(y))\}$

D_1: $\{f(1), y\}$; s_2: $\{\frac{y}{f(1)}\}$; T_2: $\{Equals(1, succ(f(1)))\}$

$\rightarrow g = \{\frac{x}{1}, \frac{y}{f(1)}\}$. (Note: We needed to standardize variables apart.)

$\rightarrow$ Note the difference to slide 15: Here, no guessing of "the right subset of Herbrand ground terms" was needed.

Col. West, *a Criminal?*

From [Russell and Norvig (2010)]:

> *The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.*

→ Prove that Col. West is a criminal.

Convention: In what follows, for better readability we will sometimes write implications $P \wedge Q \wedge R \rightarrow S$ instead of clauses $\neg P \vee \neg Q \vee \neg R \vee S$.

Col. West, *a Criminal?* Clauses

It is a crime for an American to sell weapons to hostile nations:

Clause:
$American(x_1) \wedge Weapon(y_1) \wedge Sells(x_1, y_1, z_1) \wedge Hostile(z_1) \rightarrow Criminal(x_1)$

Nono has some missiles:
$\exists x[Owns(Nono, x) \wedge Missile(x)]$
SNF & Clauses: $Owns(Nono, M)$; $Missile(M)$

All of Nono's missiles were sold to it by Colonel West.
Clause: $Missiles(x_2) \wedge Owns(Nono, x_2) \rightarrow Sells(West, x_2, Nono)$

Missiles are weapons:
Clause: $Missile(x_3) \rightarrow Weapon(x_3)$

An enemy of America counts as "hostile":
Clause: $Enemy(x_4, America) \rightarrow Hostile(x_4)$

West is an American:
$American(West)$

The country Nono is an enemy of America:
$Enemy(Nono, America)$

Col. West, *a Criminal!* PL1 Resolution Proof

$\{\neg American(x_1), \neg Weapon(y_1), \neg Sells(x_1, y_1, z_1), \neg Hostile(z_1), Criminal(x_1)\}$	$\{\neg Criminal(West)\}$
	$\{\frac{x_1}{West}\}$
$\{American(West)\}$	$\{\neg American(West), \neg Weapon(y_1), \neg Sells(West, y_1, z_1), \neg Hostile(z_1)\}$
$\{\neg Missile(x_3), Weapon(x_3)\}$	$\{\neg Weapon(y_1), \neg Sells(West, y_1, z_1), \neg Hostile(z_1)\}$
	$\{\frac{x_3}{y_1}\}$
$\{Missile(M)\}$	$\{\neg Missile(y_1), \neg Sells(West, y_1, z_1), \neg Hostile(z_1)\}$
	$\{\frac{y_1}{M}\}$
$\{\neg Missiles(x_2), \neg Owns(Nono, x_2), Sells(West, x_2, Nono)\}$	$\{\neg Sells(West, M, z_1), \neg Hostile(z_1)\}$
	$\{\frac{x_2}{M}, \frac{z_1}{Nono}\}$
$\{Missile(M)\}$	$\{\neg Missile(M), \neg Owns(Nono, M), \neg Hostile(Nono)\}$
$\{Owns(Nono, M)\}$	$\{\neg Owns(Nono, M), \neg Hostile(Nono)\}$
$\{\neg Enemy(x_4, America), Hostile(x_4)\}$	$\{\neg Hostile(Nono)\}$
	$\{\frac{x_4}{Nono}\}$
$\{Enemy(Nono, America)\}$	$\{\neg Enemy(Nono, America)\}$
	$\square$

Curiosity Killed the Cat?

From [Russell and Norvig (2010)]:

> *Everyone who loves all animals is loved by someone.*
> *Anyone who kills an animal is loved by noone.*
> *Jack loves all animals.*
> *Cats are animals.*
> *Either Jack or curiosity killed the cat (whose name is "Garfield").*

$\rightarrow$ Prove that curiosity killed the cat.

Convention: In what follows, for better readability we will sometimes write implications $P \wedge Q \wedge R \rightarrow S$ instead of clauses $\neg P \vee \neg Q \vee \neg R \vee S$.

Curiosity Killed the Cat? Clauses

Everyone who loves all animals is loved by someone:
$\forall x[\forall y(Animal(y) \rightarrow Loves(x,y)) \rightarrow \exists z Loves(z,x)]$
SNF & Clauses: $Animal(f(x_1)) \vee Loves(g(x_1), x_1)$; and
$\neg Loves(x_2, f(x_2)) \vee Loves(g(x_2), x_2)$

Anyone who kills an animal is loved by noone:
$\forall x[\exists y(Animal(y) \wedge Kills(x,y)) \rightarrow \forall z \neg Loves(z,x)]$
Clause: $\neg Animal(y_3) \vee \neg Kills(x_3, y_3) \vee \neg Loves(z_3, x_3)$

Jack loves all animals:
Clause: $Animal(x_4) \rightarrow Loves(Jack, x_4)$

Cats are animals:
Clause: $Cat(x_5) \rightarrow Animal(x_5)$

Either Jack or curiosity killed the cat (whose name is "Garfield"):
Clauses: $Kills(Jack, Garfield) \vee Kills(Curiosity, Garfield)$; and $Cat(Garfield)$

Curiosity Killed the Cat! PL1 Resolution Proof

$\{Cat(Garfield)\}$ $\quad$ $\{\neg Cat(x_5), Animal(x_5)\}$

$\{\frac{x_5}{Garfield}\}$

$\{Animal(Garfield)\}$ $\quad$ $\{\neg Animal(y_3), \neg Kills(x_3, y_3), \neg Loves(z_3, x_3)\}$

$\{\frac{y_3}{Garfield}\}$

$\{\neg Kills(x_3, Garfield), \neg Loves(z_3, x_3)\}$ $\quad$ $\{Kills(Jack, Garfield), Kills(Curiosity, Garfield)\}$ $\quad$ $\{\neg Kills(Curiosity, Garfield)\}$

$\{Kills(Jack, Garfield)\}$

$\{\frac{x_3}{Jack}\}$

$\{\neg Loves(z_3, Jack)\}$ $\quad$ $\{\neg Loves(x_2, f(x_2)), Loves(g(x_2), x_2)\}$ $\quad$ $\{\neg Animal(x_4), Loves(Jack, x_4)\}$

$\{\frac{x_2}{Jack}, \frac{x_4}{f(Jack)}\}$

$\{Loves(g(Jack), Jack), \neg Animal(f(Jack))\}$ $\quad$ $\{Animal(f(x_1)), Loves(g(x_1), x_1)\}$

$\{\frac{x_1}{Jack}\}$

$\{\frac{z_3}{g(Jack)}\}$

$\{Loves(g(Jack), Jack)\}$

$\square$

Summary

- The Herbrand universe is the set of all ground terms that can be built from the symbols used in a set θ of PL1 formulas. The (propositional-logic) Herbrand expansion instantiates the formulas with these terms, and is satisfiable iff θ is.
- For unsatisfiable θ, we can always find an unsatisfiable finite subset of the Herbrand expansion.
- PL1 resolution reasons directly about PL1 formulas (in clausal normal form) It relies on unification to compare PL1 terms.
- Binary PL1 resolution is like propositional resolution with unification. It is not refutation-complete.
- To obtain a complete PL1 resolution calculus, we can either allow to unify sets of resolution literals (full PL1 resolution), or to unify literals within clauses (factoring).
- The set of satisfiable PL1 formulas is not recursively enumerable. Thus, neither the reduction to propositional logic, nor PL1 resolution, guarantee to terminate in finite time.

Topics We Didn't Cover Here

PL1 is very expressive, but: (some people just can't get enough)

- Second-Order Logic: Quantification over predicates.

$$\forall x, y[Equals(x, y) \leftrightarrow [\forall p(p(x) \leftrightarrow p(y))]]$$

  "We define x and y to be "Equal" iff their behavior with respect to all predicates is identical."

- Temporal Logic: Quantification over future behaviors.

$$\mathbf{AG}[\varphi \implies \mathbf{EF}\psi]$$

  "For **A**ll futures, we **G**lobally have that, if $s \models \varphi$, then there **E**xists a future from s on which **F**inally we have ψ."

And what else? There's of course also *lots* of algorithmic stuff *within* PL1 that we didn't cover.

→ If you want to know all about this, take the "Automated Reasoning" courses.

Reading

- *Chapter 9: Inference in First-Order Logic*, Section 9.1, 9.2, and 9.5 [Russell and Norvig (2010)].

  Content: What I cover in "Reduction to Propositional Reasoning" is distributed in RN over Section 9.1, which gives a very brief sketch of the idea, and Section 9.5.4 which contains a summary of Herbrand's results.

  Section 9.2.2 contains a much less formal/detailed account of what I cover in "Substitutions, and Unification".

  Section 9.5.2 briefly outlines (in half a page!) what I cover in "Predicate Logic Resolution". Section 9.5.3 pretty much coincides with my "On Criminals and Cats".

  Sections 9.3 and 9.4 describe "forward chaining" and "backward chaining", relevant to Databases and Logic Programming. Nice background reading! The same applies to a few gems here and there, such as a summary of Gödel's incompleteness theorem.

  → Overall: As usual, lacks rigor, but covers a great breadth of subjects and provides nice complementary reading. Can't replace the lecture.

References I

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (Third Edition)*. Prentice-Hall, Englewood Cliffs, NJ, 2010.