Automated Reasoning Systems

Introduction

Temur Kutsia

RISC, Johannes Kepler University of Linz, Austria
kutsia@risc.uni-linz.ac.at

What is Automated Reasoning

- Reasoning: The process of making inferences.
- Automated reasoning studies methods to automate the process of reasoning.
- Automated reasoning systems: Computer programs that implement automated reasoning methods to perform reasoning automatically (or semi-automatically).

Examples of Reasoning

- All men are mortal. Socrates is a man. Therefore Socrates is mortal.
- All fruit is tasty if it is not cooked. This apple not tasty. Therefore, it is cooked.

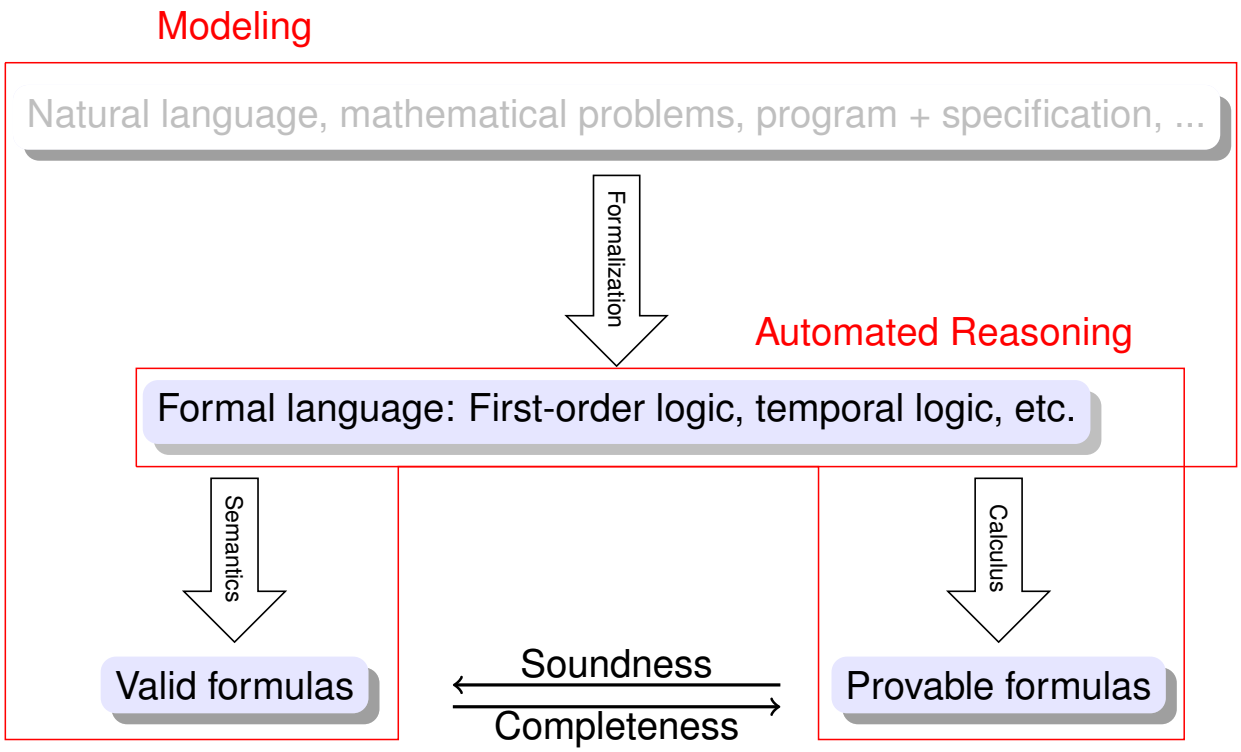
Are These Reasonings Correct?

- All that glistens is not gold. This pot does not glisten. Therefore, it is gold.
- All men are mortal. Socrates is not mortal. Therefore, Socrates is not a man.

Are These Statements True?

- There exists a person with the property that if he (or she) is a genius then everybody is a genius.
- In idempotent groups the group operation is commutative.

General Picture



Informal Example

Problem formulation (Chang and Lee, 1973):

> *Suppose that stock prices go down if the prime interest rate goes up. Suppose also that most people are unhappy when stock prices go down. Assume that prime interest rate does go up. Are most people unhappy?*

Formalization:

- P : prime interest rate goes up.
- S : stock prices go down.
- U : most people are unhappy.
- If the prime interest rate goes up, stock prices go down: $P \Rightarrow S$.
- If stock prices go down, most people are unhappy: $S \Rightarrow U$.

We should show that if $P \Rightarrow S$, $S \Rightarrow U$, and P hold, then U holds as well.

Informal Example

- We should show that if $P \Rightarrow S$, $S \Rightarrow U$, and P hold, then U holds as well.
- That means, $((P \Rightarrow S) \wedge (S \Rightarrow U) \wedge P) \Rightarrow U$ is valid.

Semantically:

P	S	U	$P \Rightarrow S$	$S \Rightarrow U$	$((P \Rightarrow S) \wedge (S \Rightarrow U) \wedge P) \Rightarrow U$
true	*true*	*true*	*true*	*true*	*true*
true	*true*	*false*	*true*	*false*	*true*
true	*false*	*true*	*false*	*true*	*true*
true	*false*	*false*	*false*	*true*	*true*
false	*true*	*true*	*true*	*true*	*true*
false	*false*	*true*	*true*	*true*	*true*
false	*true*	*false*	*true*	*false*	*true*
false	*false*	*false*	*true*	*true*	*true*

Informal Example

- We should show that if $P \Rightarrow S$, $S \Rightarrow U$, and P hold, then U holds as well.

To solve this problem by (automated) reasoning, we need to have a corresponding calculus.

- Assume we have such a calculus that is sound and complete.
- Then we should prove U in this calculus from the assumptions $P \Rightarrow S$, $S \Rightarrow U$, and P.
- (Assume the calculus has the modus ponens rule: A and $A \Rightarrow B$ imply B)

Proof:

1. $P \Rightarrow S$ (Assumption)
2. $S \Rightarrow U$ (Assumption)
3. P (Assumption)
4. S (From 3 and 1 by modus ponens)
5. U (From 4 and 2 by modus ponens)

Informal Example

- In the example we used propositional logic.
- Often we need more powerful logics.
- For instance, we need first-order logic to express this:
  - All men are mortal. Socrates is a man. Therefore Socrates is mortal.
  - $\forall x.man(x) \Rightarrow mortal(x)$: All men are mortal.
  - $man(socrates)$: Socrates is a man.
  - $mortal(socrates)$: Socrates is mortal.

First-Order Logic

- Syntax
- Semantics
- Inference system

Syntax

- Alphabet
- Terms
- Formulas

Alphabet

A first-order alphabet consists of the following sets of symbols:

- A countable set of variables $\mathcal{V}$.
- For each $n \geq 0$, a set of n-ary function symbols $\mathcal{F}^n$. Elements of $\mathcal{F}^0$ are called constants.
- For each $n \geq 0$, a set of n-ary predicate symbols $\mathcal{P}^n$.
- Logical connectives $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$.
- Quantifiers $\exists$, $\forall$.

Notation:

- x, y, z for variables.
- f, g for function symbols.
- a, b, c for constants.
- p, q for predicate symbols.

Terms

Definition

- A variable is a term.
- If $t_1, \ldots, t_n$ are terms and $f \in \mathcal{F}^n$, then $f(t_1, \ldots, t_n)$ is a term.

Notation:

- s, t, r for terms.

Formulas

Definition

- If $t_1, \ldots, t_n$ are terms and $p \in \mathcal{F}^n$, then $p(t_1, \ldots, t_n)$ is a formula. It is called an atomic formula or an atom.
- If A is a formula, $\neg A$ is a formula.
- If A and B are formulas, then $A \vee B$, $A \wedge B$, $A \Rightarrow B$, and $A \Leftrightarrow B$ are formulas.
- If A is a formula, then $\exists x.A$ and $\forall x.A$ are formulas.

Notation:

- A, B for formulas.

Example

Translating English sentences into first-order logic formulas:

1. For each natural number there exists exactly one immediate successor natural number.

$$\forall x.(\exists y.(y \doteq succ(x) \wedge \forall z.(z \doteq succ(x) \Rightarrow y \doteq z)))$$

2. There is no natural number whose immediate successor is 0.

$$\neg\exists x.0 \doteq succ(x)$$

3. For each nonzero natural number there exists exactly one immediate predecessor natural number.

$$\forall x.(\neg(x \doteq 0) \Rightarrow \exists y.(y \doteq pred(x) \wedge \forall z.(z \doteq pred(x) \Rightarrow y \doteq z)))$$

Assume:

- $succ, pred$ unary function symbols.
- $\doteq$ binary predicate symbol.

Semantics

- Meaning of a first-order language consists of an universe and an appropriate meaning of each symbol.
- This pair is called structure.
- Structure fixes interpretation of function and predicate symbols.
- Meaning of variables is determined by a variable assignment.
- Interpretation of terms and formulas.

Structure

- Structure: a pair (D, I).
- D is a nonempty universe, the domain of interpretation.
- I is an interpretation function defined on D that fixes the meaning of each symbol associating
  - to each $f \in \mathcal{F}^n$ an n-ary function $f_I : D^n \to D$, (in particular, $c_I \in D$ for each constant c)
  - to each $p \in \mathcal{P}^n$ different from $\doteq$, an n-ary relation p_I on D.

Variable Assignment

- A structure $\mathcal{S} = (D, I)$ is given.
- Variable assignment $\sigma_{\mathcal{S}}$ maps each $x \in \mathcal{V}$ into an element of D: $\sigma_{\mathcal{S}}(x) \in D$.
- Given a variable x, an assignment $\vartheta_{\mathcal{S}}$ is called an x-variant of $\sigma_{\mathcal{S}}$ iff $\vartheta_{\mathcal{S}}(y) = \sigma_{\mathcal{S}}(y)$ for all $y \neq x$.

Interpretation of Terms

- A structure $\mathcal{S} = (D, I)$ and a variable assignment $\sigma_{\mathcal{S}}$ are given.
- Value of a term t under $\mathcal{S}$ and $\sigma_{\mathcal{S}}$, $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t)$:
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(x) = \sigma_{\mathcal{S}}(x)$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(f(t_1, \ldots, t_n)) = f_I(Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t_1), \ldots, Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t_n))$.

Interpretation of Formulas

- A structure $\mathcal{S} = (D, I)$ and a variable assignment $\sigma_{\mathcal{S}}$ are given.
- Value of an atomic formula under $\mathcal{S}$ and $\sigma_{\mathcal{S}}$ is one of *true*, *false*:
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(s \doteq t) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(s) = Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t)$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(p(t_1, \ldots, t_n)) = true$ iff $(Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t_1), \ldots, Val_{\mathcal{S},\sigma_{\mathcal{S}}}(t_n)) \in p_I$.

Interpretation of Formulas

- A structure $\mathcal{S} = (D, I)$ and a variable assignment $\sigma_{\mathcal{S}}$ are given.
- Values of compound formulas under $\mathcal{S}$ and $\sigma_{\mathcal{S}}$ are also either *true* or *false*:
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(\neg A) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A) = false$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A \vee B) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A) = true$ or $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(B) = true$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A \wedge B) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A) = true$ and $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(B) = true$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A \Rightarrow B) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A) = false$ or $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(B) = true$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A \Leftrightarrow B) = true$ iff $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(A) = Val_{\mathcal{S},\sigma_{\mathcal{S}}}(B)$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(\exists x.A) = true$ iff $Val_{\mathcal{S},\vartheta_{\mathcal{S}}}(A) = true$ for some x-variant $\vartheta_{\mathcal{S}}$ of $\sigma_{\mathcal{S}}$.
  - $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(\forall x.A) = true$ iff $Val_{\mathcal{S},\vartheta_{\mathcal{S}}}(A) = true$ for all x-variants $\vartheta_{\mathcal{S}}$ of $\sigma_{\mathcal{S}}$.

Interpretation of Formulas

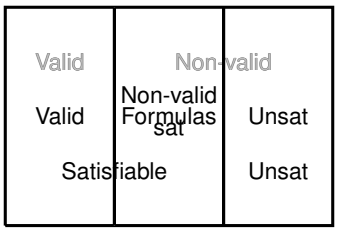
- A structure $\mathcal{S} = (D, I)$ is given.
- The value of a formula A under $\mathcal{S}$ is either *true* or *false*:
  - $Val_{\mathcal{S}}(A) = true$ iff $Val_{\mathcal{S}}, \sigma_{\mathcal{S}}(A) = true$ for all $\sigma_{\mathcal{S}}$.
- $\mathcal{S}$ is called a model of A iff $Val_{\mathcal{S}}(A) = true$.
- Written $\models_{\mathcal{S}} A$.

Example

- Formula: $\forall x.(p(x) \Rightarrow q(f(x), a))$
- Define $\mathcal{S} = (D, I)$ as
  - $D = \{1, 2\}$,
  - $a_I = 1$,
  - $f_I(1) = 2, f_I(2) = 1$,
  - $p_I = \{2\}$,
  - $q_I = \{(1, 1), (1, 2), (2, 2)\}$.
- If $\sigma_{\mathcal{S}}(x) = 1$, then $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(\forall x.(p(x) \Rightarrow q(f(x), a))) = true$.
- If $\sigma_{\mathcal{S}}(x) = 2$, then $Val_{\mathcal{S},\sigma_{\mathcal{S}}}(\forall x.(p(x) \Rightarrow q(f(x), a))) = true$.
- Hence, $\models_{\mathcal{S}} A$.

Validity, Unsatisfiability

- A formula A is valid, if $\models_{\mathcal{S}} A$ for all $\mathcal{S}$.
- Written $\models A$.
- A formula A is unsatisfiable, if $\models_{\mathcal{S}} A$ for no $\mathcal{S}$.
- If A is valid, then $\neg A$ is unsatisfiable and vice versa.



Inference System

- Several inference systems exist for first-order logic.
- Today: The Resolution Calculus

The Resolution Calculus

- Operates on the clausal fragment of first-order logic
- Clause: A formula of the form $\forall x_1. \cdots . \forall x_n.(L_1 \vee \cdots \vee L_k)$, where
  - each L_i is a literal, i.e., either an atomic formula or its negation,
  - $L_1 \vee \cdots \vee L_k$ contains no variables other than $x_1, \ldots, x_n$
- Every first-order formula can be reduced to a set of clauses.
- The reduction preserves unsatisfiability.
- Clauses are often written without quantifier prefix: $L_1 \vee \cdots \vee L_k$.

The Resolution Calculus

Two inference rules: Binary resolution and factoring.

- Binary resolution:

$$\frac{L_1 \vee A \qquad \neg L_2 \vee B}{(A \vee B)\sigma}$$

  where σ is a most general unifier of L_1 and L_2.
- Factoring:

$$\frac{L_1 \vee L_2 \vee A}{(L_1 \vee A)\sigma}$$

  where σ is a most general unifier of L_1 and L_2.

Unification

- Process of equation solving.
- Used in the inference rules to make two literals identical.
- Unification algorithm computes a most general unifier for solvable systems equations
- For unsolvable ones, it reports failure.

Example

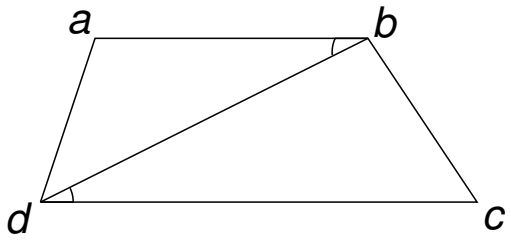
- $\{x \mapsto a, y \mapsto f(a)\}$ is a most general unifier of $p(x, f(x))$ and $p(a, y)$.
- $p(x)$ and $p(f(x))$ do not have a unifier as well as $p(x)$ and $q(y)$.

Proving by Resolution

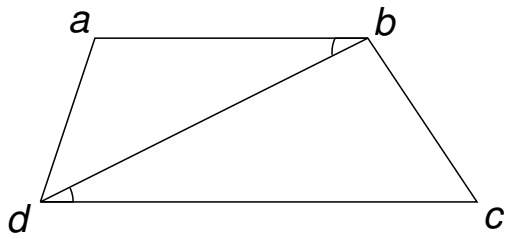
- Given a set of clauses S and a hypothesis H, to prove H from S by resolution one should
  1. Negate the hypothesis;
  2. Add the negated hypothesis to S and start derivation, trying to obtain the contradiction;
  3. In the derivation, use binary resolution and factoring rules to generate new clauses, add them to S;
  4. If the empty clause appears, stop: Contradiction found, H is proved;
  5. If no step can be made and the empty clause is not found, then H can not be proved.
- Binary resolution + factoring is a refutationally complete inference system: A set of clauses is unsatisfiable iff these rules lead to the contradiction.

Example. Proving by Resolution

Given a trapezoid, prove that the interior alternate angles obtained by the intersection of a diagonal with the bases are equal:



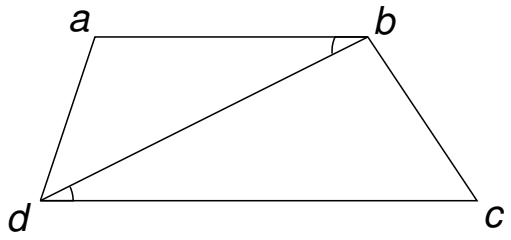
Example. Proving by Resolution



Formalization. Notation:

- $trap(x_1, x_2, x_3, x_4)$: Trapezoid with left upper vertex x_1, right upper vertex x_2, right lower vertex x_3, left lower vertex x_4.
- $par(x_1, x_2, y_1, y_2)$: The line x_1x_2 is parallel to the line y_1y_2.
- $eq(x_1, x_2, x_3, y_1, y_2, y_3)$: The angles $x_1x_2x_3$ and $y_1y_2y_3$ are equal.

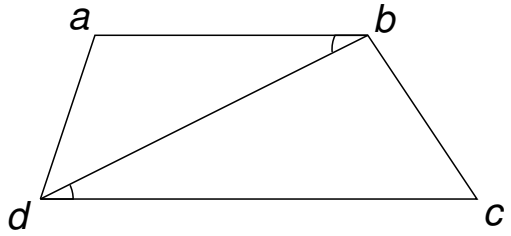
Example. Proving by Resolution



Formalization. Assumptions and hypothesis:

1. $\forall x_1.\forall x_2.\forall x_3.\forall x_4.trap(x_1, x_2, x_3, x_4) \Rightarrow par(x_1, x_2, x_3, x_4)$.
2. $\forall x_1.\forall x_2.\forall y_1.\forall y_2.par(x_1, x_2, y_1, y_2) \Rightarrow eq(x_1, x_2, y_2, y_1, y_2, x_2)$.
3. $trap(a, b, c, d)$.
4. $eq(a, b, d, c, d, b)$ (Hypothesis)

Example. Proving by Resolution



Formalization. Proof from clausified assumptions and the negated hypothesis:

1. $\neg trap(x_1, x_2, x_3, x_4) \vee par(x_1, x_2, x_3, x_4)$
2. $\neg par(x_1, x_2, y_1, y_2) \vee eq(x_1, x_2, y_2, y_1, y_2, x_2)$.
3. $trap(a, b, c, d)$.
4. $\neg eq(a, b, d, c, d, b)$
5. $\neg par(a, b, c, d)$ (Resolvent of 2 and 4)
6. $\neg trap(a, b, c, d)$ (Resolvent of 5 and 1)
7. $\square$ (Resolvent of 6 and 3, contradiction found.)

Another Example. Proving by Resolution

Factoring is important!

Show that the given set of clauses (1-3) is unsatisfiable:

1. $\neg p(x, y) \vee q(x, y)$.
2. $p(x, y) \vee q(y, x)$.
3. $\neg q(a, a) \vee \neg q(b, b)$
4. $q(x_1, y_1) \vee q(y_1, x_1)$. (Resolvent of 1 and 2)
5. $q(x_1, x_1)$ (Factor of 4)
6. $\neg q(b, b)$ (Resolvent of 5 and 3)
7. $\square$ (Resolvent of 5 and 6, contradiction found.)

Proving by Resolution

- Unrestricted application of the inference rules might lead to search space explosion.
- Most of the generated clauses are redundant.
- How to avoid generating them?
- Resolution strategies.

Ordered Resolution

- One of most efficient resolution strategies.
- Assumes a partial ordering on terms and literals.
- Ordered inference:
  - A subset of the literals is marked as maximal
  - (If the clause is ground, i.e, without variables, the order is total, and the greatest literal is marked as maximal)
  - The inference rules may be restricted in some cases so that they apply only to maximal literals.

Set-of-Support Strategy

- Task: Prove by resolution + factoring that the set of clauses S is unsatisfiable.
- Split S into two sets: U and SOS.
- Names: U for usable, SOS for set-of-support.
- Strategy: Do not allow resolution between clauses in U. Put new clauses in SOS.
- Complete if U is satisfiable.

How to Deal With Equality

- Roughly: Replace equals by equals.
- Rule: Paramodulation

$$\frac{s = t \vee C_1 \qquad L[r] \vee C_2}{(L[t] \vee C_1 \vee C_2)\sigma}$$

  where $L[r]$ is a literal in which the term r occurs and σ is a most general unifier of r and s.
- Unrestricted replacement leads to search space explosion.
- Restrictions: r can not be a variable, $s\sigma\vartheta \succ t\sigma\vartheta$ for some ϑ where $\succ$ is the given ordering, etc.

Resolution Theorem Provers

- Quite some state-of-the-art theorem provers are based on resolution.
- Among them Prover9 (the successor of Otter) and Vampire (most successful prover in the CASC competitions).
- Next lecture: Prover9.