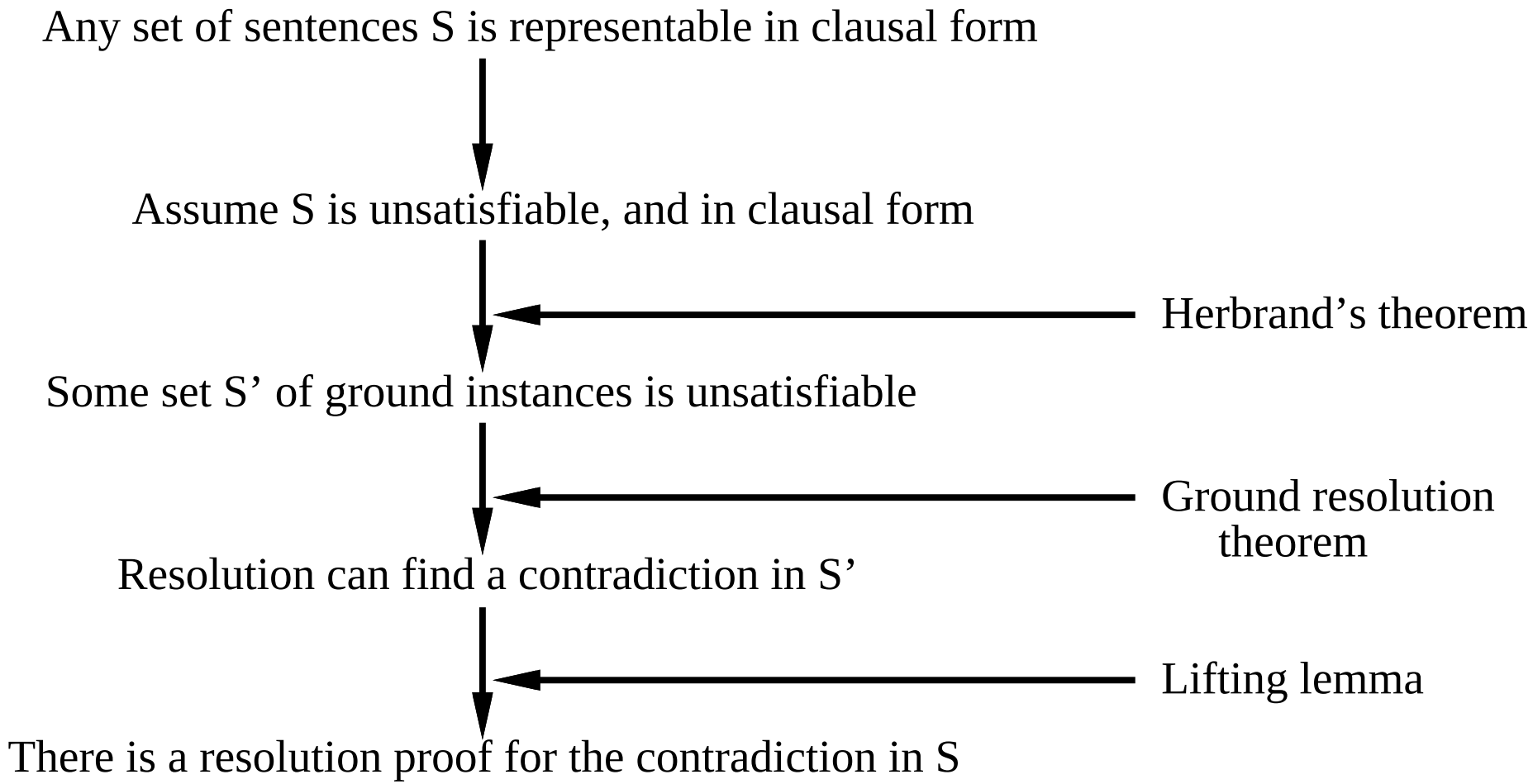
Completeness

Completeness Proof



Herbrand's Universe and Base

- Herbrand Universe: Given S set of clause, then H_S —**Herbrand Universe of S**— is the set of all ground terms for S
  Example:

$$\begin{aligned} S &= \{\neg P(x, f(x, A)) \vee \neg Q(x, A) \neg R(x, B)\} \\ H_S &= \{A, B, f(A, A), f(A, B), f(B, A), f(A, f(A, A)), \ldots\} \end{aligned}$$

- Herbrand base for S: Given S and H_S, then $H_S(S)$ is the set of all ground clauses for S wrt. H_S — the Herbrand base for S—
  Example:

$$\begin{aligned} \{\ & \neg P(A, f(A, A)) \vee \neg Q(A, A) \neg R(A, B), \\ & \neg P(B, f(B, A)) \vee \neg Q(B, A) \neg R(B, B), \\ & \neg P(f(A, A), f(f(A, A), A)) \vee \neg Q(f(A, A), A) \neg R(f(A, A), B), \\ & \neg P(f(A, B), f(f(A, B), A)) \vee \neg Q(f(A, B), A) \neg R(f(A, B), B), \\ & \neg P(f(B, A), f(f(B, A), A)) \vee \neg Q(f(B, A), A) \neg R(f(B, A), B), \\ & \neg P(f(A, f(A, A)), f(f(A, f(A, A)), A)) \vee \neg Q(f(A, f(A, A)), A) \neg R(f(A, f(A, A)), B), \end{aligned}$$

Herbrand's Theorem

[Herbrand, 1930]

> If a set S of clauses is unsatisfiable, then there exists a finite subset of $H_S(S)$ that is also unsatisfiable.

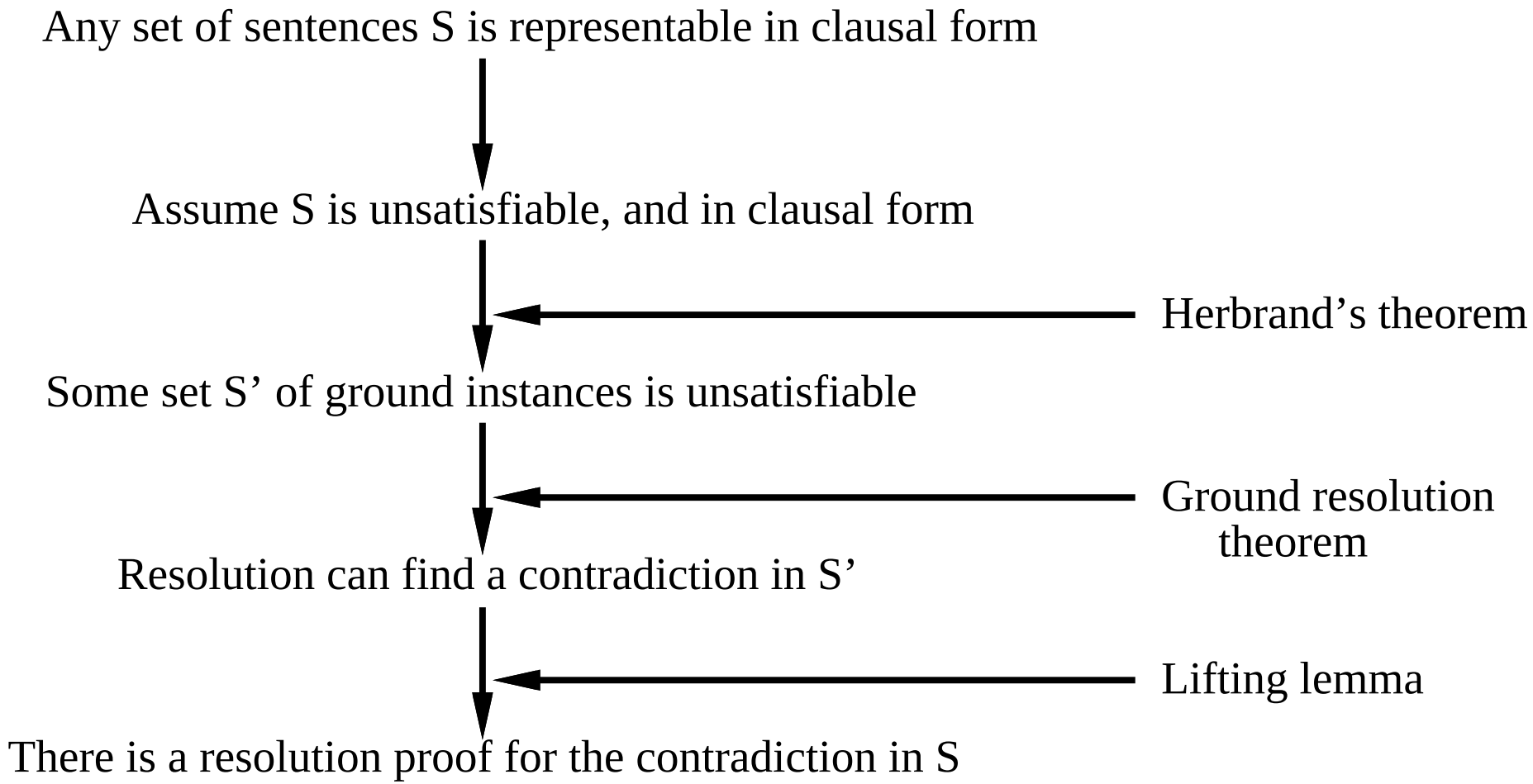
Lifting Lemma

Let C_1 and C_2 be two clauses, and C_1' and C_2' be two ground instances of C_1 and C_2.
If C' is a resolvent (prop. logic) of C_1' and C_2', then there exists a clause C, such that

- C is a resolvent of C_1 and C_2.
- C' is a ground instance of C

Resolvent = 1 resolution step + arbitrary many factorization steps

Completeness Proof



Dealing with Equality

How to deal with Equality?

- What do you need to deal with equality by using only resolution?

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \; x = x$

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$
  - $\forall x, y, z \;\; (x = y \wedge y = z) \Rightarrow x = z$

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$
  - $\forall x, y, z \;\; (x = y \wedge y = z) \Rightarrow x = z$
  - For each function symbol f or arity n that occurs in KB, add

$$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; .$$
$$(x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$
  - $\forall x, y, z \;\; (x = y \wedge y = z) \Rightarrow x = z$
  - For each function symbol f or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$
  - For each predicate symbol P or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow [P(x_1, \ldots, x_n) \Leftrightarrow P(y_1, \ldots, y_n)]$$

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$
  - $\forall x, y, z \;\; (x = y \wedge y = z) \Rightarrow x = z$
  - For each function symbol f or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$
  - For each predicate symbol P or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow [P(x_1, \ldots, x_n) \Leftrightarrow P(y_1, \ldots, y_n)]$$
- Problem: This creates a huge search space!!

How to deal with Equality?

- What do you need to deal with equality by using only resolution?
- We have to add the axioms:
  - $\forall x \;\; x = x$
  - $\forall x, y \;\; x = y \Rightarrow y = x$
  - $\forall x, y, z \;\; (x = y \wedge y = z) \Rightarrow x = z$
  - For each function symbol f or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$
  - For each predicate symbol P or arity n that occurs in KB, add

    $$\forall x_1, \ldots, x_n, y_1, \ldots y_n \;\; . \\ (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Rightarrow [P(x_1, \ldots, x_n) \Leftrightarrow P(y_1, \ldots, y_n)]$$
- Problem: This creates a huge search space!!
- You want to make equality a primitive concept and therefore have specialised rules for it.

Paramodulation

- The paramodulation rule:

$$\frac{s = t \vee l_1 \vee \ldots l_n \qquad\qquad m_1 \vee \ldots \vee m_i[u] \vee \ldots \vee m_n}{(l_1 \vee \ldots l_n \vee m_1 \vee \ldots \vee m_i[t] \vee \ldots \vee m_n)\theta}$$

  where θ is the unifier of s and u.

- Example:

$$\frac{\begin{array}{l}\neg(K = \mathrm{sort}(L)) \vee \mathrm{sort}([x|L]) = \mathrm{insert\_sort}(x, K) \\ \mathrm{Sorted}(\mathrm{sort}(y)) \vee \neg(y = \mathrm{nil})\end{array}}{\neg(K = \mathrm{sort}(L)) \vee \mathrm{Sorted}(\mathrm{insert\_sort}(x, K)) \vee \neg([x|L] = \mathrm{nil})}$$

  where $\theta = [y/[x|L]]$

Strategies

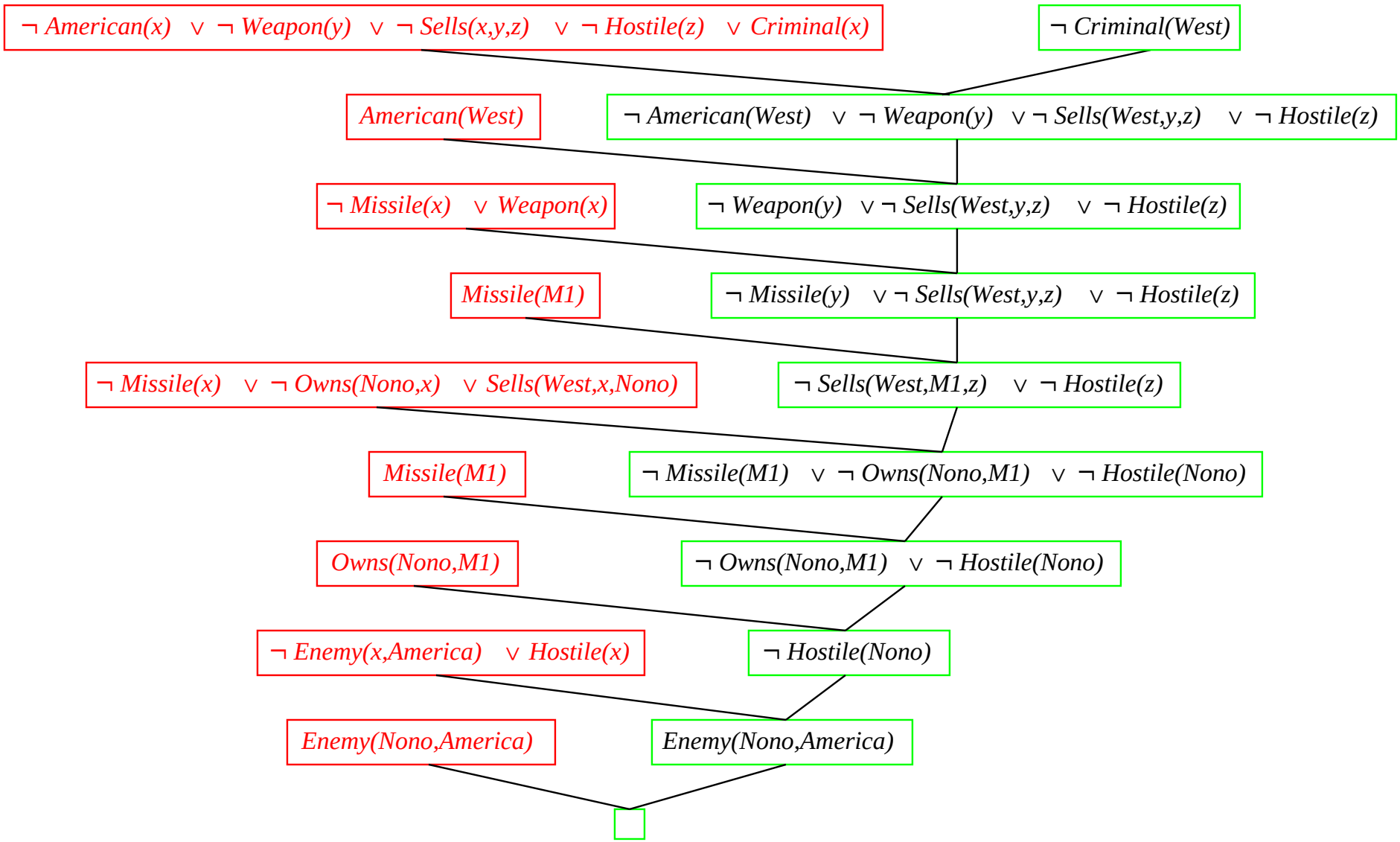
Strategies

- Unit preference: Only allow resolutions if at least one of the clauses is a unit clause — contains only one literal
  - Idea: Using unit clauses will allows result in clauses with strictly less literals
  - Incomplete in general
  - Complete for Horn clauses (each clause contains at most one positive literal)
- Set of support:
  - Procedure
    - Select a subset SoS of the set of clauses
    - Allow only resolution steps where one partner is from SoS
    - Resolvent is again in SoS
  - Not complete for every choice of SoS
  - Common approach: Take initial SoS be clauses obtained from negated conjtecture

Strategies contd

- Input resolution: Only allow resolutions between C and C' if at least one of C or C' is an input clause
  - Incomplete in general
  - Complete for Horn clauses (each clause contains at most one positive literal)
  - Variant Linear resolution: Also allow resolution between C and C' if C is an ancestor of C'
  - Linear resolution is complete
- Subsumption: Eliminate all redundant clauses.
  - A clause C is redundant, if there exists a clause C' which is more general than P.
  - $P(x)$ is more general than $P(A)$
  - $P(x)$ is more general than $P(x) \vee Q(y, x)$

Input Resolution Proof



Theorem Provers

Theorem Provers

- Automated theorem provers: OTTER [McCune92]
- Interactive proof assistants: ΩMEGA [Siekmann et.al since 1990]

OTTER

- **Set of support** SoS
- Set of usable axioms.
  - Background knowledge of the problem (i.e. KB)
  - Boundaries between usable and SoS can be initially defined by the user
- Set of rewrites $u = v$,
  - always applied from left to right
  - used for simplification: $x + 0 = x$.
- Set of parameters and clauses that define the strategy

Otter

```
procedure OTTER(sos, usable)
    inputs:  sos, a set of support---clauses defining the problem (a global variable)
             usable, background knowledge potentially relevant to the problem
    repeat
          clause ← the lightest member of sos
           move clause from sos to usable
          PROCESS(INFER(clause, usable), sos)
    until sos = [] or  a refutation has been found
```

```
function INFER(clause, usable) returns clauses
    resolve clause with each member of usable
    return the resulting clauses after applying FILTER
```

```
procedure PROCESS(clauses, sos)
    for each clause in clauses do
          clause ← SIMPLIFY(clause)
          merge identical literals
          discard clause if it is a tautology
          sos ← [clause | sos]
          if clause has no literals then a refutation has been found
          if clause has one literal then look for unit refutation
    end
```

Further Issues

- Nonsense inferences blow up search space

  $\Longrightarrow$ **Sorts, Types, Dependent Types, . . .**

Further Issues

- Nonsense inferences blow up search space

  $\Longrightarrow$ **Sorts, Types, Dependent Types, . . .**

- Some formulas introduce are bad for proof search

  $\Longrightarrow$ **Integrate theory reasoning into unification algorithm**

Further Issues

- Nonsense inferences blow up search space
  $\Longrightarrow$ **Sorts, Types, Dependent Types, . . .**
- Some formulas introduce are bad for proof search
  $\Longrightarrow$ **Integrate theory reasoning into unification algorithm**
- FOL sufficient to represent about Wumpus-like worlds, but still too restricted
  Example: For induction axioms need at least second-order logic
  $\Longrightarrow$ **Lecture Introduction to CL, Semantics of Higher-Order logics**

Further Issues

- Nonsense inferences blow up search space
  $\Longrightarrow$ **Sorts, Types, Dependent Types, . . .**
- Some formulas introduce are bad for proof search
  $\Longrightarrow$ **Integrate theory reasoning into unification algorithm**
- FOL sufficient to represent about Wumpus-like worlds, but still too restricted
  Example: For induction axioms need at least second-order logic
  $\Longrightarrow$ **Lecture Introduction to CL, Semantics of Higher-Order logics**
- Other calculi, more restrictions of the search space:
  $\Longrightarrow$ **Lecture Automated Reasoning**

Further Issues

- Nonsense inferences blow up search space
  $\Longrightarrow$ **Sorts, Types, Dependent Types, . . .**
- Some formulas introduce are bad for proof search
  $\Longrightarrow$ **Integrate theory reasoning into unification algorithm**
- FOL sufficient to represent about Wumpus-like worlds, but still too restricted
  Example: For induction axioms need at least second-order logic
  $\Longrightarrow$ **Lecture Introduction to CL, Semantics of Higher-Order logics**
- Other calculi, more restrictions of the search space:
  $\Longrightarrow$ **Lecture Automated Reasoning**
- Efficient procedures for fragments
  $\Longrightarrow$ **rewriting, BDDs, decision procedures, . . .**

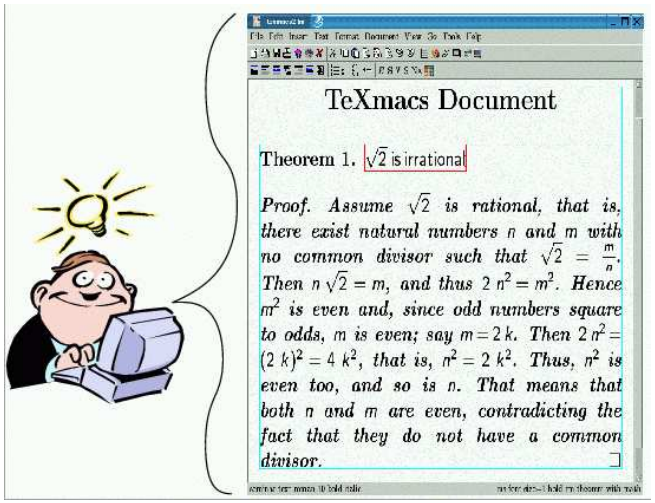
Even More Further Issues

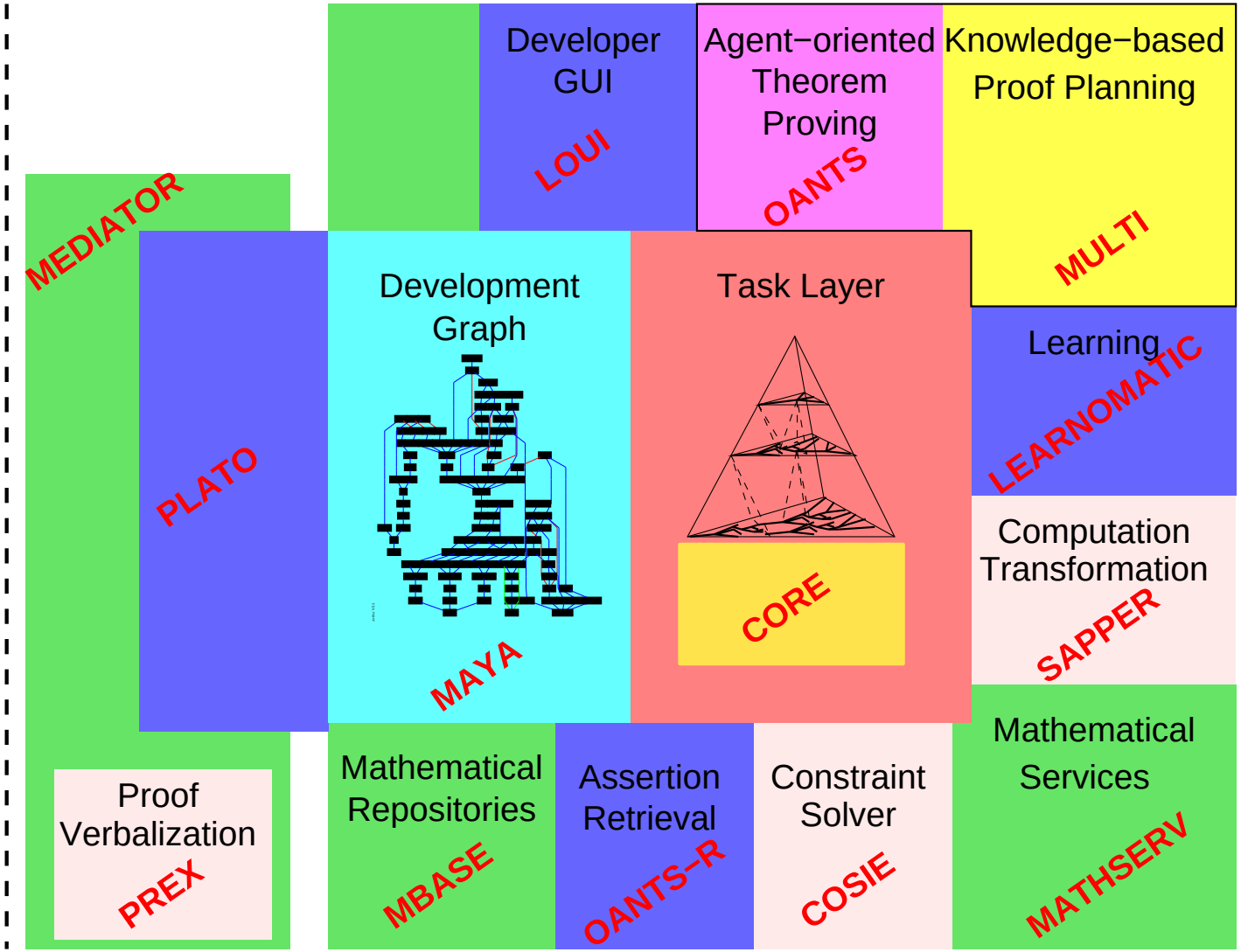
Even if we have all that, we still

only can prove conjectures from a knowledge base.

- Sufficient for the Wumpus world agent, but. . . (Compiler for a PL)
- need much more for doing software verification or mathematics (IDE for a PL)
- What is missing for doing software verification or mathematics?

DIALOG


TeXmacs Document
Theorem 1. √2 is irrational
Proof. Assume √2 is rational, that is, there exist natural numbers n and m with no common divisor such that √2 = m/n. Then n√2 = m, and thus 2n² = m². Hence m² is even and, since odd numbers square to odds, m is even; say m = 2k. Then 2n² = (2k)² = 4k², that is, n² = 2k². Thus, n² is even too, and so is n. That means that both n and m are even, contradicting the fact that they do not have a common divisor.


Developer GUI
LOUI
Agent−oriented Theorem Proving
OANTS
Knowledge−based Proof Planning
MULTI
MEDIATOR
PLATO
Development Graph
MAYA
Task Layer
CORE
Learning
LEARNOMATIC
Computation Transformation
SAPPER
Proof Verbalization
PREX
Mathematical Repositories
MBASE
Assertion Retrieval
OANTS−R
Constraint Solver
COSIE
Mathematical Services
MATHSERV

Integrating Texteditors and Proof Assistants

example.tm

File Edit Insert Text Format Document View Go Tools Help

Theory [Distributivity in Simple Sets]

Context :

We refer to the definitions and axioms of the theory ↪[Simple Sets] .

Theorem [Distributivity of intersection] :

It holds that $\forall A, B, C.\ (A \cap (B \cup C)) = ((A \cap B) \cup (A \cap C))$.

Proof :

We prove the equality of these sets by proving the following two subset relations :

(1) $((A \cap (B \cup C)) \subset ((A \cap B) \cup (A \cap C)))$

(2) $(((A \cap B) \cup (A \cap C)) \subset (A \cap (B \cup C)))$

We start by proving the first subgoal.

By the →[Definition of subset] we need to prove

$\forall x.\ x \in (A \cap (B \cup C)) \Rightarrow x \in ((A \cap B) \cup (A \cap C))$.

Assume $x \in (A \cap (B \cup C))$.

article plato menus text roman 10 start

UNIVERSITÄT DES SAARLANDES

Integrating Texteditors and Proof Assistants

example.tm

File Edit Insert Text Format Document View Go Tools Help

Theory [Distributivity in Simple Sets]

Context :

We refer to the definitions and axioms of the theory ↪ [Simple Sets] .

Theorem [Distributivity of intersection] :

It holds that $\forall A, B, C.\ (A \cap (B \cup C)) = ((A \cap B) \cup (A \cap C))$.

Proof :

We prove the equality of these sets by proving the following two subset relations :

(1) $((A \cap (B \cup C)) \subset ((A \cap B) \cup (A \cap C)))$

(2) $(((A \cap B) \cup (A \cap C)) \subset (A \cap (B \cup C)))$

We start by proving the first subgoal.

By the → [Definition of subset] we need to prove

$\forall x.\ x \in (A \cap (B \cup C)) \Rightarrow x \in ((A \cap B) \cup (A \cap C))$.

$x \in (A \cap (B \cup C))$

Assume .

Apply Definition of subset
Apply Definition of set=
Apply Definition of union -
Apply Definition of union +
Apply Definition of intersection - : Arguments: Compute Results
Apply Definition of intersection +

article plato menus text roman 10 start

Integrating Texteditors and Proof Assistants

example.tm

File Edit Insert Text Format Document View Go Tools Help

Theory [Distributivity in Simple Sets]

Context :

We refer to the definitions and axioms of the theory ↪[Simple Sets] .

Theorem [Distributivity of intersection] :

It holds that $\forall A, B, C.\ (A \cap (B \cup C)) = ((A \cap B) \cup (A \cap C))$.

Proof :

We prove the equality of these sets by proving the following two subset relations :

(1) $((A \cap (B \cup C)) \subset ((A \cap B) \cup (A \cap C)))$

(2) $(((A \cap B) \cup (A \cap C)) \subset (A \cap (B \cup C)))$

We start by proving the first subgoal.

By the →[Definition of subset] we need to prove

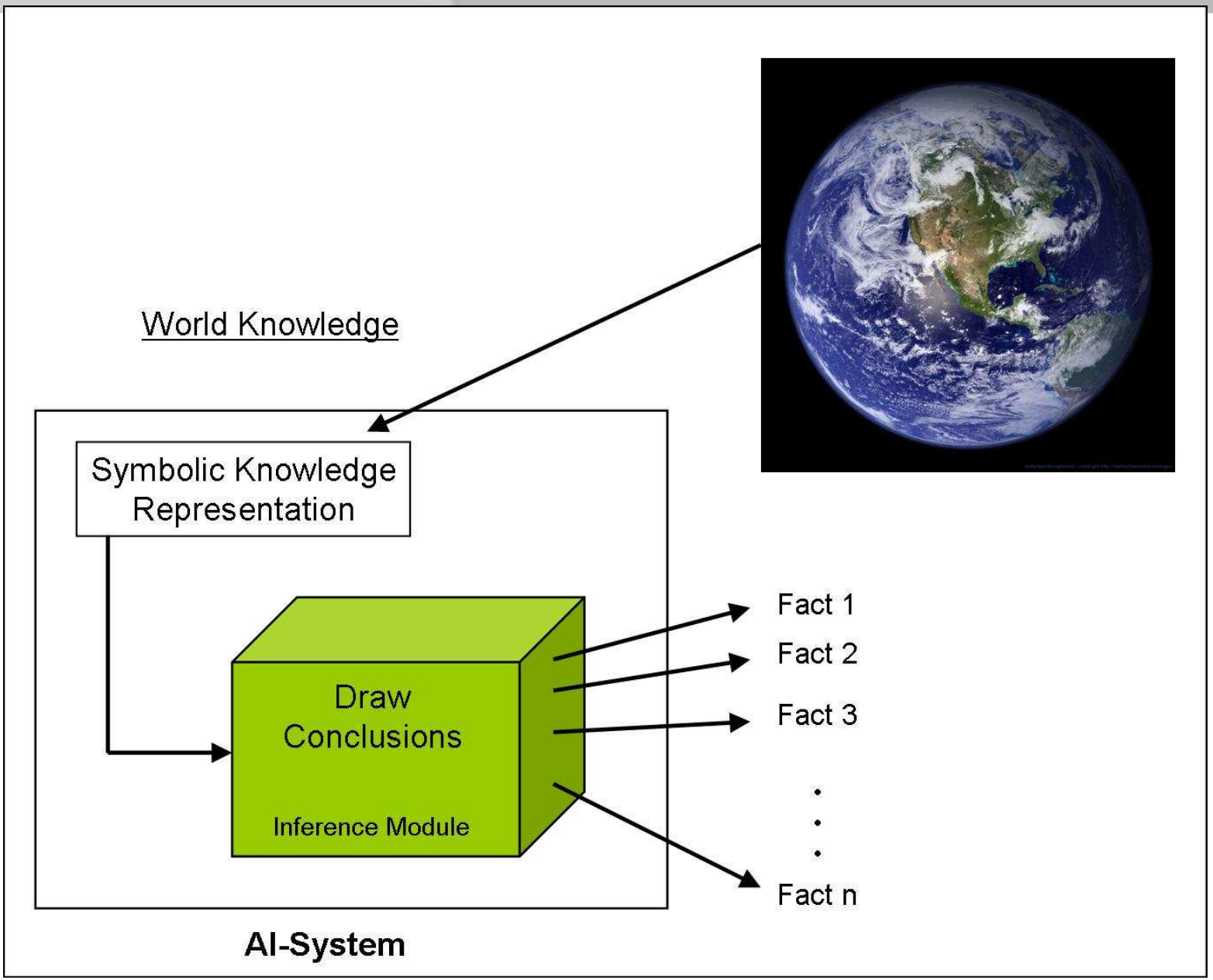
$\forall x.\ x \in (A \cap (B \cup C)) \Rightarrow x \in ((A \cap B) \cup (A \cap C))$.

Assume $x \in (A \cap (B \cup C))$.

It follows that $x \in A \wedge x \in (B \cup C)$.

article plato menus text roman 10 start

Fast First-Order Unification


World Knowledge
Symbolic Knowledge Representation
Draw Conclusions
Inference Module
Fact 1
Fact 2
Fact 3
Fact n
AI-System

A Common Operation in (almost all) AI-Systems:

1.

- Automated Theorem Proving [x]
  - "Built-in Equations"

2.

- Logic Programming
  - Features
  - Data Structures
- Pattern Invocated Procedures [x]
- Natural Language Processing
  - Unification Grammars
- Vision [x]

3.

- Knowledge Representation
- Expert Systems

Matching of Descriptions

- **Expert Systems:**
  OPS-5; Rete algorithm — Forgy 1981; 1982
- **Language Processing:**
  Feature Unification — Ait-Kaci 1984; Smolka, Ait-Kaci 1987
- **Knowledge Representation:**
  KL-ONE; Feature Unification — WINO since 1989
- **Vision:**
  Graph-Matching — Rastall; 1969
- **Pattern Directed Programming Languages:**
  Planner; Matchless — Hewitt 1972; Stickel 1975

Matching of Descriptions (cont.)

- **Term Rewriting Systems:**
  T-Unification — Peterson; Stickel 1981
- **Deduction Systems:**
  Modal-, Temporal, Epistimologische Operatoren in ◊-Unification
  Ohlbach 1988; 1989
- **Logical Programming:**
  Unification, WAM — D. Warren 1983

The Race for the Fastest (the Earliest) Unification Algorithm

- Exponential: $\mathcal{O}(2^n)$ Robinson
- Quadratic: $\mathcal{O}(n^2)$ Robinson
- Logarithmic: $\mathcal{O}(n \log n)$ Martelli-Montanari
- Linear: $\mathcal{O}(n)$ Paterson-Wegman
- Quasi-linear: $\longrightarrow$ today

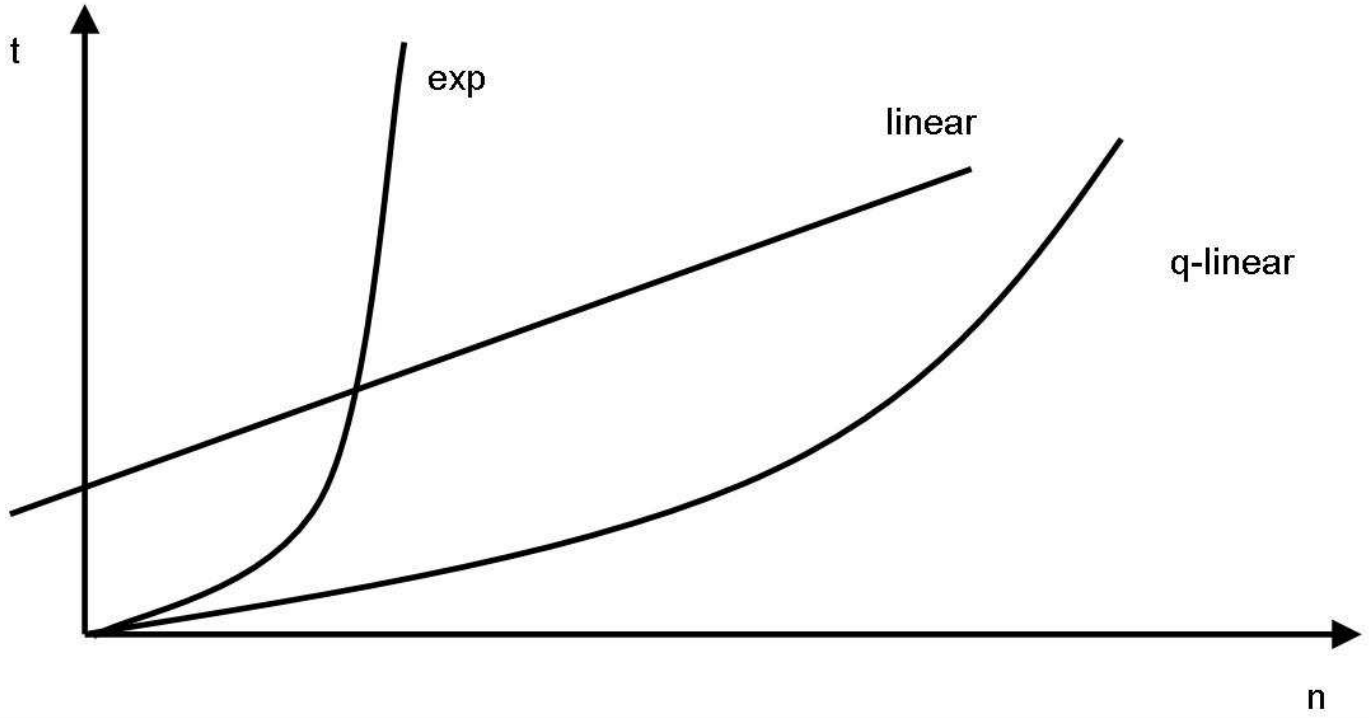
QUADRATIC (instead of exponential)
UNIFICATION
$\Uparrow$
n^2 instead of 2^n

UNIVERSITÄT DES SAARLANDES

Robinson‘s Algorithm is exponential

→ linear

→ quasi linear algorithms



Unification of Free Terms

Unsorted Signatures

	1920	Emil Post	Unification
	1930	J. Herbrand	Unification Algorithm
	1960	D. Prawitz	Most General Unifier
	1963	M. Davis	’Linked Conjunktions’
	1964	J. Guard	Unification
→	1965	A. Robinson	Most General Unifier
	1970	D. Knuth	Unification in T. R.
	1971	A. Robinson	Fast Implementation

Unification of Free Terms (cont.)

Unsorted Signatures

	1973	L. D. Baxter	Fast Unification
	1975	M. Venturini-Zille	Fast Unification
→	1976	Martelli-Montanari	(Linear)
	1976	G. Huet	Almost linear
→	1977	Paterson-Wegman	!Linear!
	1982	Kapur et al.	Improvement
	1983	Bidoit and Corbin	Improvement
	1987	Escalade + Ghallap	Improvement

The Martelli-Montanari-Algorithm

(Decomposition)

Multi-Equations

- $x_1 = x_2 = \ldots = x_n = t_1 = t_2 = \ldots = t_n$
- represented as:

$$\text{M: } \{\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_n\}\} = \{V = T\}$$

where $x_1, \ldots, x_n$ variables and $t_1, \ldots, t_n$ terms.

- Unifier for that representation:

$$\boxed{\begin{array}{l}\sigma \text{ solves } \{\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_n\}\}, \\ \text{iff} \\ \sigma\, x_1 = \ldots = \sigma\, x_n = \sigma\, t_1 = \ldots = \sigma\, t_n\end{array}}$$

- Example: $\{\{x,y\} = \{f(a), f(z)\}\}$

Martelli-Montanari Algorithm

- MM-Input: $G = \{s = t\}$.
- **Init($s = t$)**:

  $$\{\}, \{\{x\} = \{s, t\}, \{x_1\} = \{\}, \ldots \{x_n\} = \{\}\}$$

  where x_i are all variables of s, t and x is a new variable
- **Occurs-in-check:**

  $$\frac{T, \{x_1, \ldots, x_n\} = \{t\} \star U}{\bot}$$

  If some x_i occurs in t
- **Clash**:

  $$\frac{T, V = \{f(s_1, \ldots, s_n), g(t_1, \ldots, t_m)\} \cup M \star U}{\bot}$$

Martelli-Montanari Algorithm

- **Decomposition:**

$$\frac{T, V = \{f(s_1^1, \ldots, s_n^1), \ldots, f(s_1^k, \ldots, t_n^k)\} \star U}{[T, V = \{k(x_1, \ldots, x_n)\}], \{\text{Norm}(\{x_i\} = \{s_1^i, \ldots, s_n^i\}) \mid 1 \leq i \leq n\} \cup \sigma(U)}$$

for $0 \leq n \leq M$, the x_i are new and $\sigma = \{x \leftarrow f(x_1, \ldots, x_n) \mid \forall x \in V\}$.

$$\begin{aligned} \text{Norm}(V = T) &\equiv V \cup \text{Vars}(T) = T \setminus \text{Vars}(T) \\ \text{Vars}(x \star T) &\equiv x \star \text{Vars}(T) (x \text{ variable}) \\ \text{Vars}(s \star T) &\equiv \text{Vars}(T) (s \text{ not a variable}) \end{aligned}$$

- **Merge:**

$$\frac{T, \{V_1 = T_1, V_2 = T_2\} \cup U}{\{V_1 \cup V_2 = T_1 \cup T_2\}}$$

if $V_1 \cap V_2 \neq \{\,\}$, i.e some $x \in V_1$ and $x \in V_2$.

Output of M.-M-Algorithm

- A list of multiequations of the form:

$$[V_1 = \{t_1\}, \ldots, V_n = \{t_n\}]$$

- Compute the most general unifier from left to right!
- Example: $[\{x\} = \{g(y)\}, \{y\} = \{h(z)\}]$
  mgu: $\sigma = \{x \leftarrow g(h(a)), y \leftarrow h(z)\}$

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$$

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$

$\rightarrow \underbrace{[x_0 \leftarrow f(x_1, x_2)]}_{\sigma_1},$ (Decomposition)

$\{x_1 = f(x, g(a, y)) = f(h(y), g(y, a)), x_2 = g(x, h(y)) = g(z, z)\}$

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$

$\rightarrow \underbrace{[x_0 \leftarrow f(x_1, x_2)]}_{\sigma_1},$ (Decomposition)

$\{x_1 = f(x, g(a, y)) = f(h(y), g(y, a)), x_2 = g(x, h(y)) = g(z, z)\}$

$\rightarrow \underbrace{[x_3 \leftarrow f(x_4, x_5), x_6 = g(x_7, x_8)] \circ \sigma_1}_{\sigma_2},$ (2 $\times$ Decomposition)

$\{x_4 = x = h(y), x_5 = g(a, y) = g(y, a), x_7 = x = z, x_8 = z = h(y)\}$

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$

$\rightarrow \underbrace{[x_0 \leftarrow f(x_1, x_2)]}_{\sigma_1},$ (Decomposition)

$\{x_1 = f(x, g(a, y)) = f(h(y), g(y, a)), x_2 = g(x, h(y)) = g(z, z)\}$

$\rightarrow \underbrace{[x_3 \leftarrow f(x_4, x_5), x_6 = g(x_7, x_8)] \circ \sigma_1}_{\sigma_2},$ ($2 \times$ Decomposition)

$\{x_4 = x = h(y), x_5 = g(a, y) = g(y, a), x_7 = x = z, x_8 = z = h(y)\}$

$\rightarrow \sigma_2, \{x = z = x_7 = x_8 = x_4 = h(y), x_5 = g(a, y) = g(y, a)\}$ ($2 \times$ Merge)

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$

$\rightarrow \underbrace{[x_0 \leftarrow f(x_1, x_2)]}_{\sigma_1},$ (Decomposition)

$\{x_1 = f(x, g(a, y)) = f(h(y), g(y, a)), x_2 = g(x, h(y)) = g(z, z)\}$

$\rightarrow \underbrace{[x_3 \leftarrow f(x_4, x_5), x_6 = g(x_7, x_8)] \circ \sigma_1}_{\sigma_2},$ (2 $\times$ Decomposition)

$\{x_4 = x = h(y), x_5 = g(a, y) = g(y, a), x_7 = x = z, x_8 = z = h(y)\}$

$\rightarrow \sigma_2, \{x = z = x_7 = x_8 = x_4 = h(y), x_5 = g(a, y) = g(y, a)\}$ (2 $\times$ Merge)

$\rightarrow \underbrace{[\{x, z, x_7, x_8, x_4\} \leftarrow h(y)] \circ \sigma_2}_{\sigma_3}, \{a = y, y = a\}$ (2 $\times$ Decomposition)

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$[], \{k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))\}$

$\rightarrow \underbrace{[x_0 \leftarrow f(x_1, x_2)]}_{\sigma_1},$ (Decomposition)

$\{x_1 = f(x, g(a, y)) = f(h(y), g(y, a)), x_2 = g(x, h(y)) = g(z, z)\}$

$\rightarrow \underbrace{[x_3 \leftarrow f(x_4, x_5), x_6 = g(x_7, x_8)] \circ \sigma_1}_{\sigma_2},$ (2 $\times$ Decomposition)

$\{x_4 = x = h(y), x_5 = g(a, y) = g(y, a), x_7 = x = z, x_8 = z = h(y)\}$

$\rightarrow \sigma_2, \{x = z = x_7 = x_8 = x_4 = h(y), x_5 = g(a, y) = g(y, a)\}$ (2 $\times$ Merge)

$\rightarrow \underbrace{[\{x, z, x_7, x_8, x_4\} \leftarrow h(y)] \circ \sigma_2}_{\sigma_3}, \{a = y, y = a\}$ (2 $\times$ Decomposition)

$\rightarrow \sigma_3, \{y = a = a\}$ (Merge)

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$\rightarrow \sigma_3, \{y = a = a\}$ (Merge)

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$\rightarrow \sigma_3, \{y = a = a\}$ (Merge)

$\rightarrow [y \leftarrow a] \circ \sigma_3, \{\}$ (Decomposition)

Example: Unification using M.-M.

Use representation: $x_1 = \ldots = x_n = t_1 = \ldots = t_m$ for $\{x_1, \ldots, x_n\} = \{t_1, \ldots, t_m\}$

$\rightarrow \sigma_3, \{y = a = a\}$ (Merge)

$\rightarrow [y \leftarrow a] \circ \sigma_3, \{\}$ (Decomposition)

$\Downarrow$ Extraction

Most general unifier: $\{x \leftarrow h(a), z \leftarrow h(a), y \leftarrow a\}$

$$k(f(x, g(a, y)), g(x, h(y))) = k(f(h(y), g(y, a)), g(z, z))$$

Theorems about M.-M.'s algorithm

- The Martelli-Montanari unification algorithm is:
  - Sound and
  - Complete

  Proof: (invariant)

  Every rule contains $\mathbb{U}(\mathrm{G})$ (as in the usual rule syntax)
- The algorithm terminates
  (standard proof procedure for many unification algorithms)

Multisets and Multiset Ordering

- Example Multisets:

$$\begin{aligned} \{3,3,3,4,0,0\} &= \{3,0,0,3,4,3\} \\ &\neq \{0,3,4\} \end{aligned}$$

- Definition: for a partially ordered set $(S, <)$ the corresponding multiset order is defined as:
  $M \gg M'$ iff
  1. $M' = (M - X) \cup Y$ for multisets X, Y
  2. $\forall y \in Y$ there exists $x \in X$ with $X > Y$
- Example:

$$\begin{aligned} \{3,3,4,0\} &\gg \{3,4\} \\ &\gg \{3,2,2,1,1,1,4,0\} \\ &\gg \{3,3,3,3,2,2\} \end{aligned}$$

Theorem

A wellfounded order on a set $\mathbb{S}$ defines a wellfounded order on the set of multisets on $\mathbb{S}$.

Termination of Martelli-Montanari

- Complexity Mesure $\mu = (\mu_1, \mu_2)$
  $\mu_1 :=$ Multiset of depth of all (sub-)terms
  $\mu_2 :=$ Number of multisets

  $\mu = (\mu_1, \mu_2)$ is lexicograpically well founded
- **Theorem 1:** The Martelli-Montanari algorithm terminates.
- **Proof:** For every rule $\frac{T,U}{T',U'}$ we have:

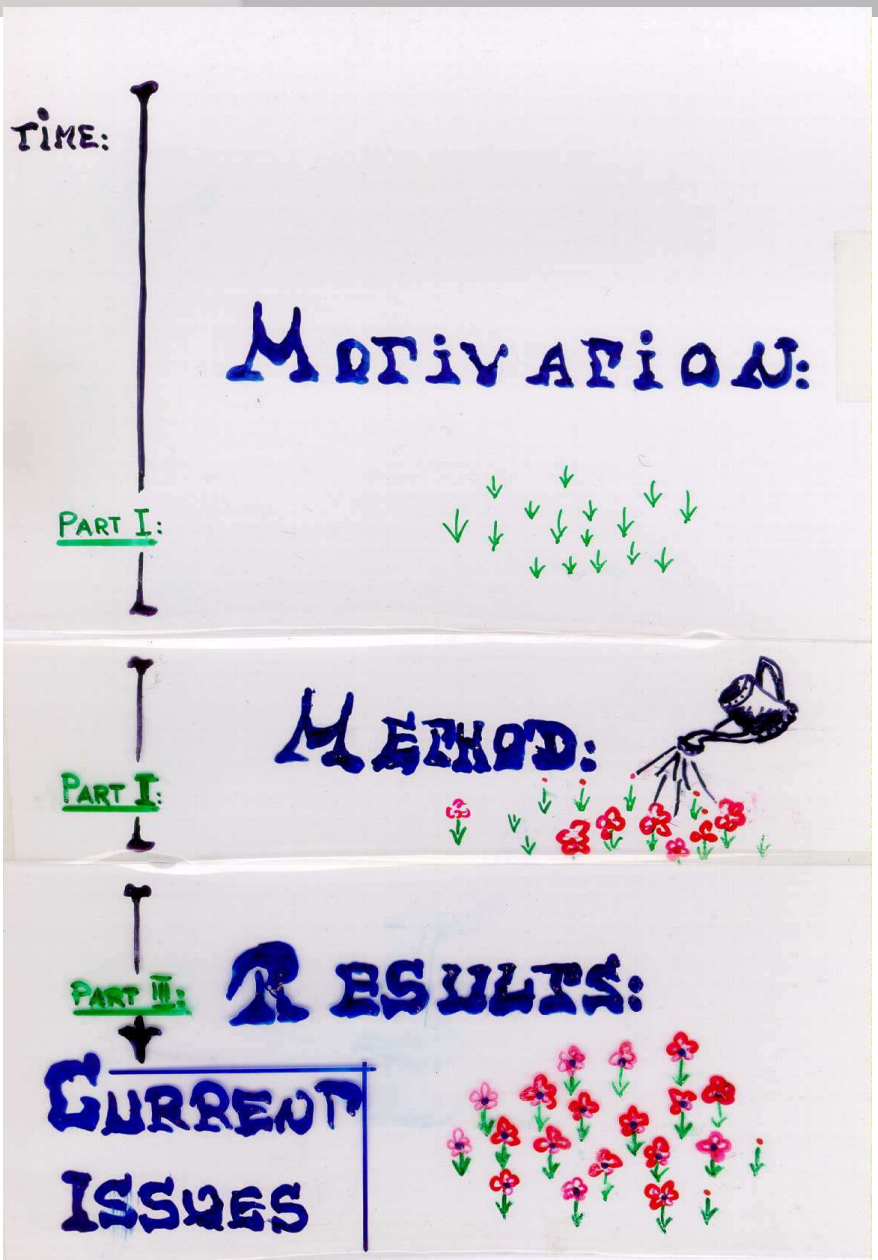
$$(\mu_1, \mu_2)_{T',U'} < (\mu_1, \mu_2)_{T,U}$$

■

- **Theorem 2:** The Martelli-Montanari algorithm is quasi-linear, i. e.

$$n \cdot \log(n)$$

First-Order Unification Theory


TIME:
MOTIVATION:
PART I:
METHOD:
PART I:
PART III:
RESULTS:
CURRENT ISSUES

Part I:

Motivation

Theorie-Unification

- Equality-reasoning difficult despite paramodulation rule
- **Enhacement:** Build specifc equalities into unification procedure
- **Examples:** $g(x, y) = g(y, x)$ (commutativity)
  - First possiblility

Theorie-Unification

- Equality-reasoning difficult despite paramodulation rule
- **Enhacement:** Build specifc equalities into unification procedure
- **Examples:** $g(x, y) = g(y, x)$ (commutativity)
  - First possiblility

$$\frac{P(g(a, b)) \quad \neg P(g(b, a))}{\text{contradiction}}$$

Theorie-Unification

- Equality-reasoning difficult despite paramodulation rule
- **Enhacement:** Build specifc equalities into unification procedure
- **Examples:** $g(x, y) = g(y, x)$ (commutativity)
  - First possiblility

$$\frac{P(g(a, b)) \quad \neg P(g(b, a))}{\text{contradiction}}$$

  - or

Theorie-Unification

- Equality-reasoning difficult despite paramodulation rule
- **Enhacement:** Build specifc equalities into unification procedure
- **Examples:** $g(x, y) = g(y, x)$ (commutativity)
  - First possiblility

$$\frac{P(g(a, b)) \quad \neg P(g(b, a))}{\text{contradiction}}$$

  - or

$$\frac{P(g(x, y)), Q(x, y) \quad \neg P(g(a, b))}{Q(a, b) \quad Q(b, a)}$$

Theorieunifikation (cont.)

- Third possibility

Theorieunifikation (cont.)

- Third possibility

$$\frac{}{Q(h(a),a),R(a,h(a))}$$
$$Q(h(y),y),R(y,h(y))$$

Theorieunifikation (cont.)

- Third possibility

$$\frac{\begin{array}{rl} \mathsf{P}(\mathsf{y}, & \mathsf{k}(\mathsf{f}(\,\mathsf{x}, \quad \mathsf{g}(\mathsf{a},\mathsf{y})), \; \mathsf{g}(\mathsf{x},\mathsf{h}(\mathsf{y})))), \quad \mathsf{Q}(\mathsf{x},\mathsf{y}) \\ \neg\mathsf{P}(\mathsf{y}', & \mathsf{k}(\mathsf{f}(\mathsf{h}(\mathsf{y}')), \mathsf{g}(\mathsf{y}',\mathsf{a})), \; \mathsf{g}(\mathsf{z},\mathsf{z}))), \quad \mathsf{R}(\mathsf{y}',\mathsf{z}) \end{array}}{\begin{array}{c} \mathsf{Q}(\mathsf{h}(\mathsf{a}),\mathsf{a}),\mathsf{R}(\mathsf{a},\mathsf{h}(\mathsf{a})) \\ \mathsf{Q}(\mathsf{h}(\mathsf{y}),\mathsf{y}),\mathsf{R}(\mathsf{y},\mathsf{h}(\mathsf{y})) \end{array}}$$

Theorieunifikation (cont.)

- Third possibility

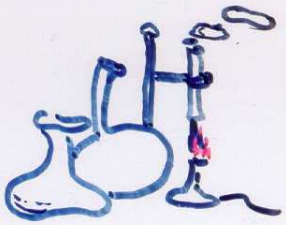
$$\frac{\begin{array}{rl} P(y, & k(f(\,x, \quad g(a,\,y)),\; g(x,h(y)))), \quad Q(x,y) \\ \neg P(y', & k(f(h(y'),\, g(y',a)),\; g(z,z))), \qquad R(y',z) \end{array}}{\begin{array}{c} Q(h(a),a), R(a,h(a)) \\ Q(h(y),y), R(y,h(y)) \end{array}}$$

- Requirements to obtain new resolvents:
  Algorithm for Unification for commutativity

The Unification Laboratory

The Unification Laboratory

Experiment 1:



Given:

U1: $f(g(a,y),x)$ and $f(x,g(a,z))$

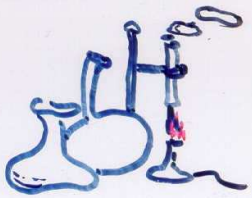
Solution:

$\sigma_1 = \{x \mapsto g(a,y), z \mapsto y\}$

Unification under Commutativity

IFICATION LABORATORY

RIMENT 1:



N:

U1: $f(g(a,y),x)$ = $f(x,g(a,z))$

SOLUTION:

$\sigma_1 = \{x \mapsto g(a,y), z \mapsto y\}$

EXPERIMENT 2:

GIVEN: U1 as before

AND C: $f(x,y) = f(y,x)$

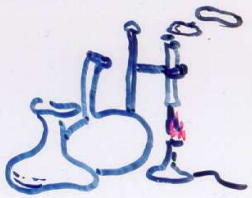
SOLUTION:

σ_1 AND $\sigma_2 = \{z \mapsto y\}$

Unification under Associativity

IFICATION LABORATORY

RIMENT 1:



N:

U1: $f(g(a,y),x)$ and $f(x,g(a,z))$

SOLUTION:

$\sigma_1 = \{x \mapsto g(a,y), z \mapsto y\}$

EXPERIMENT 3:

GIVEN: U1 as before

AND A: $f(x,f(y,z)) = f(f(x,y),z)$

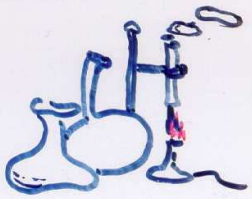
SOLUTION:

σ_1 AND $\sigma_3 = \{x \mapsto f(g(a,y),g(a,y)), z \mapsto y\}$

Unification under Assoc. and Commutativity

IFICATION LABORATORY

RIMENT 1:



N:

U1: $\dfrac{f(g(a,y),x)}{f(x,g(a,z))}$

SOLUTION: $\sigma_1 = \{x \mapsto g(a,y),\ z \mapsto y\}$

EXPERIMENT 2:

GIVEN: U1 as before

AND $\mathbb{C}$: $f(x,y) = f(y,x)$

AND $\mathbb{A}$: $f(x,f(y,z)) = f(f(x,y),z)$

SOLUTION:

What Are We Doing ?

Excursion 1:

Solving Equations

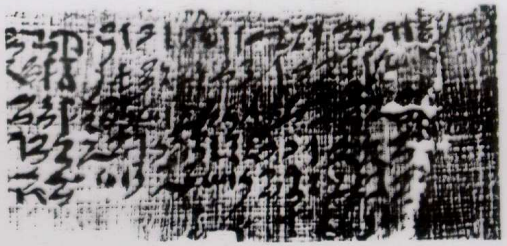
$$3x^2 + 4y = 16$$
$$x + y = 3$$


Solving Of Equations

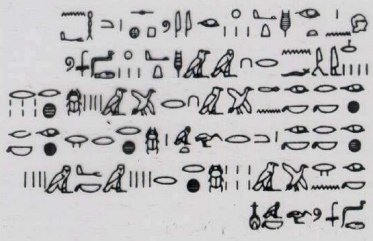
- Diophantos of Alexandria (~ 250 B.C.)
- Babylonian Mathematicians:

Stone Plate with Cuneiform Characters;
about 2000 to 3000 B.C.:

2500 B.C.



Papyrus Text, ~ 1800 B.C.
(demotian language)



Form der Berechnung eines Haufens, gerechnet 1½ mal zusammen mit 4. Er ist gekommen bis 10. Der Haufe nun nennt sich?

$1\frac{1}{2}x + 4 = 10$


Cubic Equations: $a_1x^3 + a_2x^2 + a_3x + a_4 = 0$

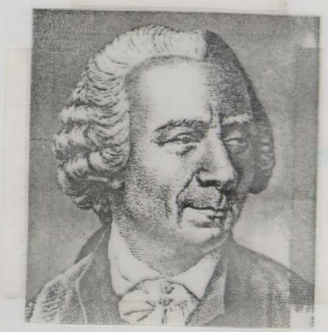
Nicolo Tartaglia
(1500 - 1557)

Geronimo Cardano
(1501 - 1576)

Equations Of The 4th Degree:

L. Ferrari (1522 - 1565)

Algebraic Equations and Trauscendental Equations:



Leonard Euler
(1707 - 1783)

François Vieta
(1540 - 1603)

"aequare"

The Whetstone Of Witte

Robert Recorde
; 1557

Howbeit, for eaſe alteratiõ of equations. I will propounde a fewe exãples, bicauſe the extraction of their rootes, maie the more aptly bee wrought. And to auoide the tediouſe repetition of theſe woordes: is equalle to: I will ſette as I doe often in woorke vſe, a paire of paralleles, or Gemowe lines of one lengthe, thus: =====, bicauſe noe. 2. thynges, can be moare equalle. And now marke theſe nombers.

1. 14.z. + .15.q. ==== 71.q.
2. 20.z. —— .18.q. ==== .102.q.
3. 26.z. + 102. ==== 9.z. — 102. + 213.q.

Aus dem „Wetzstein des Witzes", 1557, des Engländers R. Recorde. Erstes Auftreten des Gleichheitszeichens. Unmittelbar über den Formeln die Begründung für die Wahl dieses Symbols.

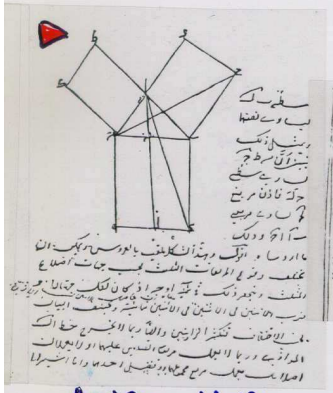
René Descartes
(1596 - 1650)

"x", "y", "z"

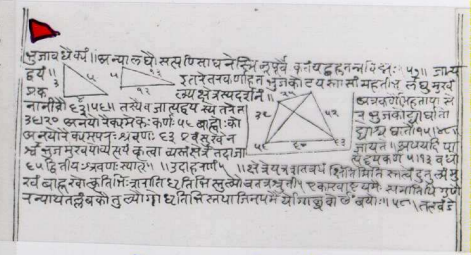

BUT: How To Solve it ?

- Euclid (300 B.C.), Elemente, Book X

However: based on Geometric



Arabian Maths



Indian Maths

Adam Riese

- Has Every Equation:

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 = 0$$

A Solution ?

- René Descartes (1596 - 1650)
- Jean D'Alembert (1717 - 1783)
- C. F. Gauss (1777 - 1855)

Main Theorem of Algebra

- 1799

- Can every solution be expressed as a radical ?

Niels Henrik Abel (1802 - 1829)

E. Galois (1811-1832)

There Are Still A Few Things To Do !!

Pierre De Fermat (1601 - 1665)

$$X^n + Y^n = Z^n$$

For $n > 2$:

$\exists\, x, y, z$?

D. Hilbert:

Is it decidable if

$P(x) = 0$

has a solution in $\mathbb{N}$?

Paris, 1900

D. Hilbert (1862 - 1943)


PART 2.
METHOD

THEORY UNIFICATION

- TERMINOLOGY
- E- UNIFICATION:
  - C- UNIFICATION

    $f(x,y) = f(y,x)$
  - A- UNIFICATION

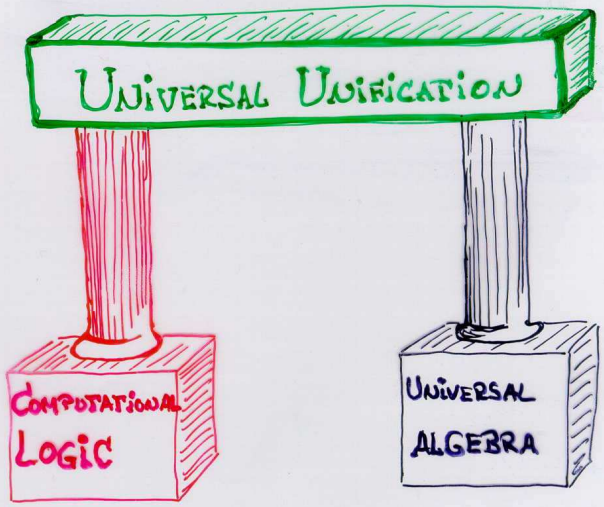
    $f(x, f(y,z)) = f(f(x,y),z)$
  - AC- UNIFICATION

T- UNIFICATION

The Classical Setting: Method

However, To Generalize
One Needs Experience..

Grätzer,
Universal
Algebra



EQUATIONAL LOGIC

1. FREE ALGEBRA OF TERMS
2. EQUATIONAL THEORY:

$T := \{ s_1 = t_1 ; s_2 = t_2 ; \ldots ; s_n = t_n \}$

$s_i, t_i \in$ TERM (FOPC)

G. Birkhoff, "On the Structure of Abstract Algebras" (1935)

A. Tarski, "Equational Logic and Equational Theories of Algebra" (1968)

W. Taylor, "Equational Logic" (1979)

3. COMPUTATIONAL LOGIC:
   - Term Rewriting
   - Paramodulation

Equational Theories

- Definition:
  Let E be a set of Σ-Equations.

  $$E \models s = t \quad \text{iff} \quad \mathcal{A} \models E \text{ then } \mathcal{A} \models s = t$$

  i. e. all Σ-models of E are also models of $s = t$.

- Definition:
  E is an equational theory over Σ:

  $$s = t \in E \quad \text{if and only if} \quad E \models s = t$$

  i. e. $E := \{s = t | E \models s = t\}$

- Definition:
  Every (finite) subset $AX \subseteq E$ with

  $$E := \{s = t | AX \models s = t\}$$

is a (finite) axiomatisation of E

Term Rewriting Systems

- Rewriting step:

  $s \longrightarrow_{\pi,l=r,\sigma} t$ (or $s \longrightarrow_E t$ for short)

  if π position in s,
  σ substitution,
  $l = r$ / $r = l$ axiom in E,
  $\sigma(l) = \sigma(s_{|\pi})$, and
  $t = s\,|_{\pi \leftarrow \sigma r}$.

- Theorem: $s \longleftrightarrow_{E^*} t \Leftrightarrow s =_E t$
- Normalforms:

  $$\text{Noetherian} + \text{Confluent} \quad \Longrightarrow \quad \text{Normal Form}$$

- Confluence and Noetherian properties give decision procedure

  $s =_T t$ iff $\|s\| = \|t\|$ for s, t terms

Birkhoff Theorem

- Birkhoff's Calculus for Equational Logic
  - $E \vdash t = t$
  - $E \vdash s = t$, if $E \vdash t = s$
  - $E \vdash r = t$, if $E \vdash r = s$ and $E \vdash s = t$
  - $E \vdash f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$, if $E \vdash s_1 = t_1, \ldots, s_n = t_n$
  - $E \vdash \sigma s = \sigma t$, if $s = t \in E$ and $\sigma \in \Sigma$

  With $=_E$ we denote the smallest relation on $\mathcal{T}_\Sigma$, that is closed under the upper rules (the smallest substitution invariant congruence relation).
- Completeness Theorem:

$$E \vDash s = t \;\Leftrightarrow\; E \vdash s = t \;\Leftrightarrow\; s =_E t$$

Definitions

- Theory Unification Problem: $\langle s = t \rangle_T$:

$$\boxed{\exists \sigma \in \Sigma . \sigma s =_T \sigma t \text{ for } T \in \mathcal{T}_=}$$

- Theory Matching Problem: $\langle s \leq t \rangle_T$:

$$\boxed{\exists \mu \in \Sigma . \mu s =_T t \text{ for } T \in \mathcal{T}_=}$$

Most General Set of Unifiers

- For $\langle s = t \rangle_T$
  - Σ: set of substitutions
  - $\mathbb{U}_T(s = t)$: set of unifiers
  - $\mu\mathbb{U}_T(s = t)$: minimal set of unifiers
- Correctness: $\forall \sigma \in \mu\mathbb{U}_T(s = t).\sigma s =_T \sigma t$
- Completeness: $\forall \delta \in \mathbb{U}_T(s = t).\exists \sigma \in \mu\mathbb{U}_T(s = t)$ such that $\sigma \sqsubseteq \delta$
- Minimality: $\forall \tau, \sigma \in \mu\Sigma.\sigma \not\sqsubseteq \tau$

Universal Unification: A Typehierarchy

- For $\langle s = t \rangle_T$
  - Σ: set of substitutions
  - $\mathbb{U}_T(s = t)$: set of unifiers
  - $\mu\mathbb{U}_T(s = t)$: minimal set of unifiers
- Typehierarchy:

(i)	$\mathcal{T} \in \mathcal{U}_1$	is **unitary** if	$\lvert\mu\mathbb{U}_T\rvert \leq 1$
(ii)	$\mathcal{T} \in \mathcal{U}_\omega$	is **finitary** if	$\lvert\mu\mathbb{U}_T\rvert \in \mathbb{N}$
(iii)	$\mathcal{T} \in \mathcal{U}_\infty$	is **infinitary** if	$\exists\langle s = t \rangle_T$ such that $\lvert\mu\mathbb{U}_T\rvert \in \infty$
(iv)	$\mathcal{T} \in \mathcal{U}_0$	**nullary** **type zero**	otherwise

E-Unification

- E-Unification-Problem Γ:

$$\langle s_i = t_i : \ 1 \leq i \leq n \rangle_E$$

- E-Unification-Sets:

$$\mathbb{U}_E(\Gamma) = \{\sigma \in \Sigma . \sigma s_i =_E \sigma t_i, \ 1 \leq i \leq n\}$$

Example: $\Gamma = \{f(g(a, y), x) = f(x, g(a, z))\}$

$$\mathbb{U}_C(\Gamma) := \{x \leftarrow g(a, y), y \leftarrow z\}, \qquad \{y \leftarrow z\}\}$$

- E-Equal on V: $\sigma =_E \tau[V] \Leftrightarrow \sigma x =_E \tau x, \forall x \in V$
- E-Instance on V: $\sigma \geq_E \tau[V] \Leftrightarrow \exists \rho . \sigma =_E \rho \circ \tau[V]$
- E-Equivalent on V: $\sigma \equiv_E \tau[V] \Leftrightarrow \sigma \geq_E \tau[V] \ \& \ \tau \geq_E \sigma[V]$

Properties

- **Lemma:**
  $\forall \sigma, \tau : \tau \geq_{\mathsf{E}} \sigma[\mathsf{V}]$ if $\sigma \in \mathbb{U}_{\mathsf{E}}(\Gamma)$ then $\tau \in \mathbb{U}_{\mathsf{E}}(\Gamma)$
- **Theorem:**
  1. $\geq_{\mathsf{E}} [\mathsf{V}]$ is a quasi-ordering
  2. $\equiv_{\mathsf{E}} [\mathsf{V}]$ is an equivalence relation

Basis $\mu\mathbb{U}$ of a Quasi-Ordered Set $(\mathbb{U}, >)$

1. $\mu\mathbb{U} \subseteq \mathbb{U}$
2. $\forall \delta \in \mathbb{U}\, \exists \sigma \in \mu\mathbb{U} :\ \delta \geq \sigma$
3. $\forall \sigma, \tau \in \mu\mathbb{U} :\ \sigma \geq \tau \ \Rightarrow\ \sigma = \tau$

Theorem: The basis $\mu\mathbb{U}$ is unique

Basis $\mu\mathbb{U}_{\mathsf{E}}$ of a E-Unification-Problem Γ

(1) $\mu\mathbb{U}_{\mathsf{E}}(\Gamma) \subseteq \mathbb{U}_{\mathsf{E}}(\Gamma)$ (Correctness)

(2) $\forall \delta \in \mathbb{U}_{\mathsf{E}}(\Gamma)\, \exists \sigma \in \mu\mathbb{U}_{\mathsf{E}}(\Gamma) :\ \delta \geq_{\mathsf{E}} \sigma\, [\mathsf{V}]$ (Completeness)

(3) $\forall \sigma, \delta \in \mu\mathbb{U}_{\mathsf{E}}(\Gamma) :\ \sigma \geq_{\mathsf{E}} \delta\, [\mathsf{V}] \ \Rightarrow\ \sigma \equiv_{\mathsf{E}} \delta$ (Minimality)

Corrollary:

1. The set of most general E-Unifiers is unique (modulo E-Equivalence).
2. The sets of MGUs have same cardinality.
3. Replacement of elements in $\mu\mathbb{U}$ by E-Equivalent ones gives again a basis.

Some special theories

Theory	Type	Unification Decidable?	$\mu\mathbb{U}_{\mathsf{T}}?$
$\{\}$	1	Yes	Yes
A	∞	Yes	Yes
C	ω	Yes	Yes
I	ω	Yes	Yes
A+C	ω	Yes	Yes
A+I	0	Yes	No
C+I	ω	Yes	Yes
A+C+I	ω	Yes	Yes

Some special theories (2)

Theory	Type	Unification Decidable?	$\mu\mathbb{U}_{\mathsf{T}}?$
D	∞	?	Yes
D+A	∞	No	Yes
D+C	∞	?	Yes
D+A+C	∞	No	Yes
D+A+I	?	Yes	No
Hom	1	Yes	Yes
Hom+A	∞	No	Yes
Applications			

Summary

- You can make unification faster using special procedures and special representations
- You can unify modulo a theory, but pay a price (decidable/undecidable, more/infinitely many unifiers)
- Unification theory develops means to study the decidability and (minimal) solution sets.