Deduction

Deductive systems

Deductive databases

Parsing as deduction

Explaining deductive reasoning

DEDUCTIVE INFERENCING

COMPUTATION

=

DEDUCTION

+

CONTROL

Kowalski

LAYERS OF A DEDUCTION SYSTEM

Four layers connecting deduction and control

1. **Logic – syntax and semantics of expressions**
2. **Calculus – syntactic derivations on formulas**
3. **Representation – state of formulas and derivations**
4. **Control – strategies and heuristics for selection**

Most common examples

1.	**First-order predicate logic**		
2.	**Gentzen calculi**	**Resolution**	**Theory resolution**
3.	**Tableau**	**Matrix**	**Clause graphs**
4.	**Special techniques**		

PROPERTIES OF CALCULI

Function

Syntactic **derivation to verify** ***semantic*** **validity (true under all interpretations)**

Structure

Set of ***axioms*** **(tautologies), minimal and** ***derivation rules***

Behavior

Forward chaining – deductive calculus

Backward chaining – test calculus

Assessing a calculus

Soundness **– all axioms and derivable formulas are valid**

Completeness **– all valid formulas are derivable (in a finite number of steps)**

Some fundamental insights

First-order predicate logic is ***not decidable*****, but** ***complete***

Higher-order predicate logic is ***not complete*** **(Gödel 1931)**

First-order predicate logic is the ***most expressive*** **and still** ***complete*** **logic (Lindström 1969)**

GENTZEN CALCULUS – NATURAL DEDUCTION

A positive deductive calculus

13 Derivation rules

$$\frac{F \quad G}{F \wedge G}$$

And-Intro.

$$\frac{F}{F \vee G} \qquad \frac{G}{F \vee G}$$

Or-Intro.

$$\frac{\begin{matrix}[F]\\ G\end{matrix}}{F \Rightarrow G}$$

Implication-Intro.

$$\frac{Fa}{\forall x Fx}$$

All-Intro.

$$\frac{Fa}{\exists x Fx}$$

Exist-Intro.

$$\frac{\begin{matrix}[F]\\ \Box\end{matrix}}{\neg F}$$

Negation-Intro.

$$\frac{F \wedge G}{F} \qquad \frac{F \wedge G}{G}$$

And-Elim.

$$\frac{F \vee G \quad \begin{matrix}[F]\\ H\end{matrix} \quad \begin{matrix}[G]\\ H\end{matrix}}{H}$$

Or-Elim.

$$\frac{F \quad F \Rightarrow G}{G}$$

Implication-Elim.

$$\frac{\forall x Fx}{Fa}$$

All-Elim.

$$\frac{\exists x Fx \quad \begin{matrix}[Fa]\\ H\end{matrix}}{H}$$

Exist-Elim.

$$\frac{F \quad \neg F}{\Box} \qquad \frac{\Box}{F}$$

Negation-Elim.

One axiom ($F \vee \neg F$) needed for obtaining completeness

RESOLUTION (Robinson 1965)

A negative test calculus

Uses formulas in form of clauses

(set of literals, implicitly disjoint)

- **One axiom, elementary contradiction**
  □ (empty clause)
- **One resolution rule**

	Simplest form:	***Resolution with substitution:***	
clause1:	$L, K_1, \ldots, K_n$	$L, K_1, \ldots, K_n$	
clause2:	$\neg L, M_1, \ldots, M_m$	$\neg L', M_1, \ldots, M_m$	$\sigma L = \sigma L'$
resolvent:	$K_1, \ldots, K_n, M_1, \ldots, M_m$	$\sigma K_1, \ldots, \sigma K_n, \sigma M_1, \ldots, \sigma M_m$	

Resolution has *refutation completeness*

RESOLUTION - AN EXAMPLE

A barber shaves a person if and only if

that person does not shave himself.

Formalization of the statement and normalization into disjunctive Normal form

shave(barber,x) ↔ ¬shave(x,x) transformed into:

(shave(barber,x) → ¬shave(x,x)) ^ (¬shave(x,x) → shave(barber,x))

(¬shave(barber,x) ∨ ¬shave(x,x)) ^ (shave(x,x) ∨ shave(barber,x))

Factorization prior to substitution:

σ = {x ← barber, y ← barber}

shave(x,x), shave(barber,x) |- shave(barber,barber)

¬shave(barber,y), ¬shave(y,y) |- ¬shave(barber,barber)

□

THEORY RESOLUTION

Extension of the resolution principle to a domain theory

No interpretation makes both L and ¬L true

- **in resolution only for *syntactic* complementarity**
- **extended to (multiple) *semantic* complementarity**

An example

clause1: a<b, K

clause2: b<c, M

clause3: c<a, N

---

resolvent: K, M, N

REPRESENTATION

Efficiency through reduction rules

- ***logical* simplifications**

  (tautologies replaced by true, contradictions by false, replacements similar to elimination rules in ND calculus)

- **simplifications with '*useless*' clauses**

  (e.g.: isolation rule – clause with 'unique' literal)

Design of reduction rules 'creative', needs justification

Consequences of representation

State transition instead of calculus

- ***Initial* states are sets of clauses**
- ***Intermediate* states are e.g., clauses graphs**
- ***Terminal* states – with or without empty clause**

CONTROL

Efficiency through syntactically-based strategies

- ***Restriction*** **strategies**

  (e.g., unit resolution – complete for Horn clauses, input resolution, set-of-support, …)

- ***Ordering*** **strategies**

  (e.g.: saturation level, with unit preference, fairness, unit resolution, …)

An example prover - OTTER

- **Elaborate implementation, indexing techniques**
- **User categorises clauses into (ordinary) clauses, axioms, and set of support, specifies options**
- **Makes resolution with axioms and 'best' clause from set-of-support**

DEDUCTIVE DATABASES

Components

- **Tables (*existensional* database)**
- **Rules (*intensional* database)**

Rules define intensional database on the base of the extensional one

Procedure

- **Elementary relations expressed in tables**
- **General, recursive relations expressed in rules**
- **Query evaluation through resolution derivation**

Benefit

- **Combines capabilities of relational databases and rule-based systems**
- **Strong reduction of redundancy and number of tables**

Problem: Control over chained application of rules

REPRESENTATION IN A RELATIONAL DATABASE

Example database relation

PARENT	
CNAME	PNAME
Smith John Jr	Smith John
Smith John Jr	Smith Mary
Rogers Charles	Rogers Linda
Rogers Linda	Jones David
Rogers Linda	Jones Mary
Smith Mary	Ford Albert
Cramer Steven	Cramer William

Relational calculus query to find the parent(s) of Charles Rogers

- **GET(X) : PARENT(Rogers Charles, X)**

REPRESENTATION WITH TWO TABLES

Example database relation

PARENT	
CNAME	PNAME
Smith John Jr	Smith John
Smith John Jr	Smith Mary
Rogers Charles	Rogers Linda
Rogers Linda	Jones David
Rogers Linda	Jones Mary
Smith Mary	Ford Albert
Cramer Steven	Cramer William

GRANDPARENT	
CNAME	PNAME
Rogers Charles	Jones David
Rogers Charles	Jones Mary
Smith John Jr	Ford Albert

Relational calculus query to find the grandparent(s) of Charles Rogers

- **GET(X) : GRANDPARENT(Rogers Charles, X)**

REPRESENTATION IN A DEDUCTIVE DATABASE

Extensional database

PARENT	
CNAME	PNAME
Smith John Jr	Smith John
Smith John Jr	Smith Mary
Rogers Charles	Rogers Linda
Rogers Linda	Jones David
Rogers Linda	Jones Mary
Smith Mary	Ford Albert
Cramer Steven	Cramer William

Intensional database

∀X∀Y∀Z PARENT(X,Y)

&

PARENT(Y,Z)

→

GRANDPARENT(X,Z)

Relational calculus query to find the grandparent(s) of Charles Rogers

- **GET(X) : GRANDPARENT(Rogers Charles, X)**

REPRESENTATION AS LOGICAL FORMULAS

PARENT(Smith John Jr, Smith John)

PARENT(Smith John Jr, Smith Mary)

PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Linda, Jones David)

PARENT(Rogers Linda, Jones Mary)

PARENT(Smith Mary, Ford Albert)

PARENT(Cramer Steven, Cramer William)

$$\forall X \forall Y \forall Z\ \text{PARENT}(X,Y)\ \&\ \text{PARENT}(Y,Z) \rightarrow \text{GRANDPARENT}(X,Z)$$

PROOF OF ASSERTIONS WITH THE DATABASE

a) **“Is Rogers Charles a parent of Rogers Linda?”**

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"
¬PARENT(Rogers Charles, Rogers Linda)

PROOF OF ASSERTIONS WITH THE DATABASE

a) **"Is Rogers Charles a parent of Rogers Linda?"**

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

PROOF OF ASSERTIONS WITH THE DATABASE

a) **"Is Rogers Charles a parent of Rogers Linda?"**

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) **"Is Rogers Charles a grandparent of Jones Mary?"**

PROOF OF ASSERTIONS WITH THE DATABASE

a) **"Is Rogers Charles a parent of Rogers Linda?"**

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) **"Is Rogers Charles a grandparent of Jones Mary?"**

¬GRANDPARENT(Rogers Charles, Jones Mary)

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) "Is Rogers Charles a grandparent of Jones Mary?"

¬GRANDPARENT(Rogers Charles, Jones Mary)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) "Is Rogers Charles a grandparent of Jones Mary?"

¬GRANDPARENT(Rogers Charles, Jones Mary)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, Jones Mary)

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) "Is Rogers Charles a grandparent of Jones Mary?"

¬GRANDPARENT(Rogers Charles, Jones Mary)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, Jones Mary)

PARENT(Rogers Charles, Rogers Linda)

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) "Is Rogers Charles a grandparent of Jones Mary?"

¬GRANDPARENT(Rogers Charles, Jones Mary)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, Jones Mary)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, Jones Mary)

PROOF OF ASSERTIONS WITH THE DATABASE

a) "Is Rogers Charles a parent of Rogers Linda?"

¬PARENT(Rogers Charles, Rogers Linda)

PARENT(Rogers Charles, Rogers Linda)

[]

b) "Is Rogers Charles a grandparent of Jones Mary?"

¬GRANDPARENT(Rogers Charles, Jones Mary)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, Jones Mary)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, Jones Mary)

PARENT(Rogers Linda, Jones Mary)

[]

REPRESENTATION WITH TWO TABLES

Extensional database

PARENT	
CNAME	PNAME
Smith John Jr	Smith John
Smith John Jr	Smith Mary
Rogers Charles	Rogers Linda
Rogers Linda	Jones David
Rogers Linda	Jones Mary
Smith Mary	Ford Albert
Cramer Steven	Cramer William

MALE
NAME
Smith John Jr
Smith John
Rogers Charles
Jones David
Ford Albert
Cramer Steven
Cramer William

Intensional database

$\forall X \forall Y$ **PARENT(X,Y) & MALE(Y) → FATHER(X,Y)**

$\forall X \forall Y \forall Z$ **PARENT(X,Y) & PARENT(Y,Z) → GRANDPARENT(X,Z)**

$\forall X \forall Y \forall Z$ **GRANDPARENT(X,Y) & FATHER(Z,Y) → GRANDFATHER(X,Y)**

ANSWERING A QUERY WITH THE DATABASE (1)

“Who are the grandparents of Rogers Charles?”

ANSWERING A QUERY WITH THE DATABASE (1)

“Who are the grandparents of Rogers Charles?”

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

ANSWERING A QUERY WITH THE DATABASE (1)

"Who are the grandparents of Rogers Charles?"

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

ANSWERING A QUERY WITH THE DATABASE (1)

“Who are the grandparents of Rogers Charles?”

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, V)

ANSWERING A QUERY WITH THE DATABASE (1)

"Who are the grandparents of Rogers Charles?"

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, V)

PARENT(Rogers Charles, Rogers Linda)

ANSWERING A QUERY WITH THE DATABASE (1)

"Who are the grandparents of Rogers Charles?"

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, V)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, V)

ANSWERING A QUERY WITH THE DATABASE (1)

"Who are the grandparents of Rogers Charles?"

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, V)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, V)

PARENT(Rogers Linda, Jones Mary)

[]

ANSWERING A QUERY WITH THE DATABASE (1)

“Who are the grandparents of Rogers Charles?”

Finding instantiations for the variable in the query representation

¬GRANDPARENT(Rogers Charles, V)

¬PARENT(X,Y) ∨ ¬ PARENT(Y,Z) ∨ GRANDPARENT(X,Z)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, V)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, V)

PARENT(Rogers Linda, Jones Mary)

PARENT(Rogers Linda, Jones David)

[] []

ANSWERING A QUERY WITH THE DATABASE (2)

“Who is the grandfather of Rogers Charles?”

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

ANSWERING A QUERY WITH THE DATABASE (2)

“Who is the grandfather of Rogers Charles?”

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

ANSWERING A QUERY WITH THE DATABASE (2)

“Who is the grandfather of Rogers Charles?”

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

ANSWERING A QUERY WITH THE DATABASE (2)

“Who is the grandfather of Rogers Charles?”

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

¬PARENT(Rogers Linda, X) ∨ ¬MALE(X)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

¬PARENT(Rogers Linda, X) ∨ ¬MALE(X)

PARENT(Rogers Linda, Jones David)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

¬PARENT(Rogers Linda, X) ∨ ¬MALE(X)

PARENT(Rogers Linda, Jones David)

¬MALE(Jones David)

ANSWERING A QUERY WITH THE DATABASE (2)

"Who is the grandfather of Rogers Charles?"

¬GRANDFATHER(Rogers Charles, X)

¬GRANDPARENT(V,Y) ∨ ¬ FATHER(Z,Y) ∨ GRANDFATHER(V, Y)

¬GRANDPARENT(Rogers Charles,X) ∨ ¬FATHER(Z, X)

¬PARENT(V,Y) ∨ ¬ PARENT(Y,W) ∨ GRANDPARENT(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬FATHER(Z, X)

¬PARENT(V,W) ∨ ¬ MALE(W) ∨ FATHER(V, W)

¬PARENT(Rogers Charles,Y) ∨ ¬PARENT(Y, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

PARENT(Rogers Charles, Rogers Linda)

¬PARENT(Rogers Linda, X) ∨ ¬PARENT(Z, X) ∨ ¬MALE(X)

¬PARENT(Rogers Linda, X) ∨ ¬MALE(X)

PARENT(Rogers Linda, Jones David)

¬MALE(Jones David)

MALE(Jones David)

[]

THE STATE OF DEDUCTIVE DATABASES

Problems with deductive databases

- **No commercial deductive database (one company producing DDBMS had to close)**
- **Prolog implementations are much faster than deductive databases**

Use of ideas of deductive databases

- **Prolog is used successfully in industry**
- **Constraint logic programming is very successful in industry**
- **Ideas of deductive databases used in extensions to standard relational databases**
- **Answer set programming as a new logic programming formalism**

PARSING AS DEDUCTION

Connection between parsing and deduction

- **Axiomatization of context-free grammars in definite clauses (subset of first-order logic)**
- **Identification of context-free parsing with deduction for a restricted class of definite clauses**
- **Extension to larger classes of definite clauses by replacing atomic grammar symbols by complex ones matched by unification – constraints specified by an argument**
- **Further extended to unification grammars**

Parsing algorithms

- ***Offline* – constraints *after* context-free parsing**
- ***Online* – constraints *during* context-free parsing (considered here)**

DEFINITE CLAUSES

Definitions – definite clauses

$P \Leftarrow Q_1$ **& … &** Q_n

- P **is true if** Q_1 **and … and** Q_n **are true**
- P **is *positive* literal or *head***
- $Q_1 \ldots Q_n$ **are *negative* literals or *body***
- **Literals have the form** $p(t_1,\ldots,t_k)$ **with predicate** p **(arity** k**) and** t_i **as arguments (terms)**

Definitions – a program

- **A set of clauses (input clauses) is a *program***
- **A program defines the relations between the predicates appearing in the heads of the clauses**
- **A goal statement** $\Leftarrow P$ **requires finding provable instances of** P

DEFINITE CLAUSES

Definite clause grammars

Context-free rule $X \rightarrow \alpha_1 \dots \alpha_n$

translated to definite clause $X(S_0,S_n) \Leftarrow \alpha_1(S_0,S_1)\ \&\ \dots\ \&\ \alpha_n(S_{n-1},S_n)$

with variables S_i being string arguments (positions in the input string)

- **Generalization by adding predicate arguments to string arguments**

Deduction in definite clauses

Resolution

(1) $B \Leftarrow A_1\ \&\ \dots\ \&\ A_m$

(2) $C \Leftarrow D_1\ \&\ \dots\ \&\ D_i\ \&\ \dots\ \&\ D_n$

if B and D_i are unifiable by substitution σ, infer

(3) $\sigma[C \Leftarrow D_1\ \&\ \dots\ \&\ D_{i-1}\ \&\ A_1\ \&\ \dots\ \&\ A_m\ \dots\ \&\ D_{i+1}\ \dots\ \&\ D_n]$

Clause (3) is a derived clause, resolvent from (1) and (2)

EARLEY DEDUCTION

Definitions

- **Definite clauses divided into *program* and *state***
- **Program – set of *input* clauses, fixed**
- **State – set of *derived* clauses, nonunit clauses with one negative literal selected (initially the goal)**

Inference rules

- ***Instantiation* – selected literal of some clause unifies with a positive literal of a *nonunit* clause C in the program, deriving the instantiation $\sigma[C]$**

  (σ is the most general unifier of the two literals)
- ***Reduction* – selected literal of some clause unifies with a *unit* clause in the program or in the current state, deriving the clause $\sigma[C']$**

  C' is C minus the selected literal (σ is the most general unifier of the two literals)

Techniques

- **Mixed top-down bottom-up mechanism**
- **Blockage of derived clauses subsumed by the state**
- **Handling gaps, dependencies by extra arguments**

EXAMPLE DEDUCTION PROOF

Context free grammar			*Definite clause program*			
S	**→**	**NP VP**	**s(S0,S)**	**⇐**	**np(S0,S1) & vp(S1,S)**	**(20)**
NP	**→**	**Det N**	**np(S0,S)**	**⇐**	**det(S0,S1) & n(S1,S)**	**(21)**
Det	**→**	**NP Gen**	**det(S0,S)**	**⇐**	**np(S0,S1) & gen(S1,S)**	**(22)**
Det	**→**	**Art**	**det(S0,S)**	**⇐**	**art(S0,S)**	**(23)**
Det	**→**	**A**	**det(S0,S)**			**(24)**
VP	**→**	**V NP**	**vp(S0,S)**	**⇐**	**v(S0,S1) & np(S1,S)**	**(25)**

Lexical categories

of the sentence

$_0$Agatha$_1$'s$_2$husband$_3$hit$_4$Ulrich$_5$

represented by the unit clauses

n(0,1) gen(1,2) n(2,3) v(3,4) n(4,5) **(26)**

goal statement ans ⇐ s(0,5) provable, if (26) is a sentence

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	**⇐**	**np(S0,S1) & vp(S1,S)**		**(20)**
np(S0,S)	**⇐**	**det(S0,S1) & n(S1,S)**		**(21)**
det(S0,S)	**⇐**	**np(S0,S1) & gen(S1,S)**		**(22)**
det(S0,S)	**⇐**	**art(S0,S)**		**(23)**
det(S0,S)				**(24)**
vp(S0,S)	**⇐**	**v(S0,S1) & np(S1,S)**		**(25)**
ans	**⇐**	**s(0,5)**	**goal statement**	**(33)**

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	$\Leftarrow$	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	$\Leftarrow$	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	$\Leftarrow$	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	$\Leftarrow$	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	$\Leftarrow$	v(S0,S1) & np(S1,S)		(25)
ans	$\Leftarrow$	s(0,5)	goal statement	(33)
s(0,5)	$\Leftarrow$	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	$\Leftarrow$	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	$\Leftarrow$	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	$\Leftarrow$	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	$\Leftarrow$	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	$\Leftarrow$	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	$\Leftarrow$	v(S0,S1) & np(S1,S)		(25)
ans	$\Leftarrow$	s(0,5)	goal statement	(33)
s(0,5)	$\Leftarrow$	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	$\Leftarrow$	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	$\Leftarrow$	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	$\Leftarrow$	art(0,S)	(35) instantiates (23)	(37)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)
vp(1,5)	⇐	v(1,S1) & np(S1,5)	(40) instantiates (25)	(41)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)
vp(1,5)	⇐	v(1,S1) & np(S1,5)	(40) instantiates (25)	(41)
det(0,S)	⇐	gen(1,5)	(39) reduces (36)	(42)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	**⇐**	**np(S0,S1) & vp(S1,S)**		**(20)**
np(S0,S)	**⇐**	**det(S0,S1) & n(S1,S)**		**(21)**
det(S0,S)	**⇐**	**np(S0,S1) & gen(S1,S)**		**(22)**
det(S0,S)	**⇐**	**art(S0,S)**		**(23)**
det(S0,S)				**(24)**
vp(S0,S)	**⇐**	**v(S0,S1) & np(S1,S)**		**(25)**
ans	**⇐**	**s(0,5)**	**goal statement**	**(33)**
s(0,5)	**⇐**	**np(0,S1) & vp(S1,5)**	**(33) instantiates (20)**	**(34)**
np(0,S)	**⇐**	**det(0,S1) & n(S1,S)**	**(34) instantiates (21)**	**(35)**
det(0,S)	**⇐**	**np(0,S1) & gen(S1,S)**	**(35) instantiates (22)**	**(36)**
det(0,S)	**⇐**	**art(0,S)**	**(35) instantiates (23)**	**(37)**
np(0,S)	**⇐**	**n(0,S)**	**(24) reduces (35)**	**(38)**
np(0,1)			**(27) reduces (38)**	**(39)**
s(0,5)	**⇐**	**vp(1,5)**	**(39) reduces (34)**	**(40)**
vp(1,5)	**⇐**	**v(1,S1) & np(S1,5)**	**(40) instantiates (25)**	**(41)**
det(0,S)	**⇐**	**gen(1,5)**	**(39) reduces (36)**	**(42)**
det(0,2)			**(28) reduces (42)**	**(43)**

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)
vp(1,5)	⇐	v(1,S1) & np(S1,5)	(40) instantiates (25)	(41)
det(0,S)	⇐	gen(1,5)	(39) reduces (36)	(42)
det(0,2)			(28) reduces (42)	(43)
np(0,S)	⇐	n(2,5)	(43) reduces (35)	(44)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)
vp(1,5)	⇐	v(1,S1) & np(S1,5)	(40) instantiates (25)	(41)
det(0,S)	⇐	gen(1,5)	(39) reduces (36)	(42)
det(0,2)			(28) reduces (42)	(43)
np(0,S)	⇐	n(2,5)	(43) reduces (35)	(44)
np(0,3)			(29) reduces (44)	(45)

EXAMPLE DEDUCTION PROOF (1)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
ans	⇐	s(0,5)	goal statement	(33)
s(0,5)	⇐	np(0,S1) & vp(S1,5)	(33) instantiates (20)	(34)
np(0,S)	⇐	det(0,S1) & n(S1,S)	(34) instantiates (21)	(35)
det(0,S)	⇐	np(0,S1) & gen(S1,S)	(35) instantiates (22)	(36)
det(0,S)	⇐	art(0,S)	(35) instantiates (23)	(37)
np(0,S)	⇐	n(0,S)	(24) reduces (35)	(38)
np(0,1)			(27) reduces (38)	(39)
s(0,5)	⇐	vp(1,5)	(39) reduces (34)	(40)
vp(1,5)	⇐	v(1,S1) & np(S1,5)	(40) instantiates (25)	(41)
det(0,S)	⇐	gen(1,5)	(39) reduces (36)	(42)
det(0,2)			(28) reduces (42)	(43)
np(0,S)	⇐	n(2,5)	(43) reduces (35)	(44)
np(0,3)			(29) reduces (44)	(45)
s(0,5)	⇐	vp(3,5)	(45) reduces (34)	(46)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	**⇐**	**np(S0,S1) & vp(S1,S)**		**(20)**
np(S0,S)	**⇐**	**det(S0,S1) & n(S1,S)**		**(21)**
det(S0,S)	**⇐**	**np(S0,S1) & gen(S1,S)**		**(22)**
det(S0,S)	**⇐**	**art(S0,S)**		**(23)**
det(S0,S)				**(24)**
vp(S0,S)	**⇐**	**v(S0,S1) & np(S1,S)**		**(25)**
det(0,S)	**⇐**	**gen(3,S)**	**(45) reduces (36)**	**(47)**

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	**⇐**	**np(S0,S1) & vp(S1,S)**		**(20)**
np(S0,S)	**⇐**	**det(S0,S1) & n(S1,S)**		**(21)**
det(S0,S)	**⇐**	**np(S0,S1) & gen(S1,S)**		**(22)**
det(S0,S)	**⇐**	**art(S0,S)**		**(23)**
det(S0,S)				**(24)**
vp(S0,S)	**⇐**	**v(S0,S1) & np(S1,S)**		**(25)**
det(0,S)	**⇐**	**gen(3,S)**	**(45) reduces (36)**	**(47)**
vp(3,5)	**⇐**	**v(3,S1) & np(S1,5)**	**(46) instantiates (25)**	**(48)**

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)
np(4,S)	⇐	n(4,S)	(24) reduces (53)	(55)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)
np(4,S)	⇐	n(4,S)	(24) reduces (53)	(55)
np(4,5)			(31) reduces (54)	(56)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	**⇐**	**np(S0,S1) & vp(S1,S)**		**(20)**
np(S0,S)	**⇐**	**det(S0,S1) & n(S1,S)**		**(21)**
det(S0,S)	**⇐**	**np(S0,S1) & gen(S1,S)**		**(22)**
det(S0,S)	**⇐**	**art(S0,S)**		**(23)**
det(S0,S)				**(24)**
vp(S0,S)	**⇐**	**v(S0,S1) & np(S1,S)**		**(25)**
det(0,S)	**⇐**	**gen(3,S)**	**(45) reduces (36)**	**(47)**
vp(3,5)	**⇐**	**v(3,S1) & np(S1,5)**	**(46) instantiates (25)**	**(48)**
vp(3,5)	**⇐**	**np(4,5)**	**(30) reduces (48)**	**(49)**
np(4,5)	**⇐**	**det(4,S1) & n(S1,5)**	**(49) instantiates (21)**	**(50)**
det(4,S)	**⇐**	**np(4,S1) & gen(S1,S)**	**(50) instantiates (22)**	**(51)**
det(4,S)	**⇐**	**art(4,S)**	**(50) instantiates (23)**	**(52)**
np(4,S)	**⇐**	**det(4,S1) & n(S1,S)**	**(51) instantiates (21)**	**(53)**
np(4,5)	**⇐**	**n(4,5)**	**(24) reduces (50)**	**(54)**
np(4,S)	**⇐**	**n(4,S)**	**(24) reduces (53)**	**(55)**
np(4,5)			**(31) reduces (54)**	**(56)**
vp(3,5)			**(56) reduces (49)**	**(57)**

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)
np(4,S)	⇐	n(4,S)	(24) reduces (53)	(55)
np(4,5)			(31) reduces (54)	(56)
vp(3,5)			(56) reduces (49)	(57)
det(4,5)	⇐	gen(5,S)	(56) reduces (51)	(58)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)
np(4,S)	⇐	n(4,S)	(24) reduces (53)	(55)
np(4,5)			(31) reduces (54)	(56)
vp(3,5)			(56) reduces (49)	(57)
det(4,5)	⇐	gen(5,S)	(56) reduces (51)	(58)
s(0,5)			(57) reduces (46)	(59)

EXAMPLE DEDUCTION PROOF (2)

s(S0,S)	⇐	np(S0,S1) & vp(S1,S)		(20)
np(S0,S)	⇐	det(S0,S1) & n(S1,S)		(21)
det(S0,S)	⇐	np(S0,S1) & gen(S1,S)		(22)
det(S0,S)	⇐	art(S0,S)		(23)
det(S0,S)				(24)
vp(S0,S)	⇐	v(S0,S1) & np(S1,S)		(25)
det(0,S)	⇐	gen(3,S)	(45) reduces (36)	(47)
vp(3,5)	⇐	v(3,S1) & np(S1,5)	(46) instantiates (25)	(48)
vp(3,5)	⇐	np(4,5)	(30) reduces (48)	(49)
np(4,5)	⇐	det(4,S1) & n(S1,5)	(49) instantiates (21)	(50)
det(4,S)	⇐	np(4,S1) & gen(S1,S)	(50) instantiates (22)	(51)
det(4,S)	⇐	art(4,S)	(50) instantiates (23)	(52)
np(4,S)	⇐	det(4,S1) & n(S1,S)	(51) instantiates (21)	(53)
np(4,5)	⇐	n(4,5)	(24) reduces (50)	(54)
np(4,S)	⇐	n(4,S)	(24) reduces (53)	(55)
np(4,5)			(31) reduces (54)	(56)
vp(3,5)			(56) reduces (49)	(57)
det(4,5)	⇐	gen(5,S)	(56) reduces (51)	(58)
s(0,5)			(57) reduces (46)	(59)
ans			(59) reduces (33)	(60)

Basics of Answer Set Programming

Declarative programming oriented towards difficult search problems

Stable model (answer set) semantics of logic programming

Answer set is a model (an interpretation) of a logical program

Answer Set Programming has its roots in

- **(deductive) databases**
- **logic programming (with negation)**
- **(logic-based) knowledge representation and**
- **(nonmonotonic) reasoning constraint solving**

  (in particular, SATisfiability testing)

KR's shift of paradigm

***Theorem Proving* based approach**

(e. g. Prolog)

- **Provide a representation of the problem**
- **A solution is given by a *derivation of a query***

***Model Generation* based approach**

(eg. SATisfiability testing)

- **Provide a representation of the problem**
- **A solution is given by *a model of the representation***

General assessment

Technical perspective

various extensions

e.g., distinction weak/strong negation
non-monotonic reasoning enabled

very effective implementations, tools

a research area of its own

Application perspective

Wide area of application

(everything that can be formulated as a logical program)