Computational Logic

Logic Programming

Cezary Kaliszyk **Georg Moser**

Institute of Computer Science @ UIBK

Winter 2015

Grounders and Solvers

Comparison

Prolog	ASP
• proof search	• model search
• Turing complete	• finite domain
• control	• specification language
• efficiency	• generality

Summary of Last Lecture

Answer Set Programming

- novel approach to modelling and solving search and optimisation problems
- ¬ programming, but a specification language
- ¬ Turing complete
- purely declarative
- restricted to finite models

Example ((part of) 8-queens problem)

```
:- not (1 = count(Y : queen(X,Y))), row(X)
```

- expresses that exactly one queen appears in every row and column
- is read as a rule: "if X is a row, 1 = count(Y : queen(X,Y)) holds"

Outline of the Lecture

Monotone Logic Programs

introduction, basic constructs, unification, database and recursive programming, termination

Incomplete Data Structures and Constraints

incomplete data structures, definite clause grammars, constraint logic programming, answer set programming

Full Prolog

semantics, correctness proofs, meta-logical predicates, cuts nondeterministic programming, efficient programs, complexity

Theory of Logic Programs

Definitions

- goal clause

$$\leftarrow B_1, \ldots, B_n$$

consists of sequence $B_1, \ldots, B_n$ of goals

- empty goal clause $\leftarrow$ is denoted by $\square$
- resolvent of goal clause $\leftarrow B_1, \ldots, B_i, \ldots, B_m$ and rule $A \leftarrow A_1, \ldots, A_n$ is goal clause

$$\leftarrow B_1\theta, \ldots, B_{i-1}\theta, A_1\theta, \ldots, A_n\theta, B_{i+1}\theta \ldots, B_m\theta$$

provided B_i (selected goal) and A unify with most general unifier θ

NB: see week 2 for the most general unifier

Selective Linear Definite Clause Resolution

Definitions

- SLD-derivation of logic program P and goal clause G consists of
  1. maximal sequence $G_0, G_1, G_2, \ldots$ of goal clauses
  2. sequence $C_0, C_1, C_2, \ldots$ of variants of rules in P
  3. sequence $\theta_0, \theta_1, \theta_2, \ldots$ of substitutions

  such that
  - $G_0 = G$
  - G_{i+1} is resolvent of G_i and C_i with mgu θ_i
  - C_i has no variables in common with $G, C_0, \ldots, C_{i-1}$
- SLD refutation is finite SLD derivation ending in $\square$
- computed answer substitution of SLD refutation of P and G with substitutions $\theta_0, \theta_1, \ldots, \theta_m$ is restriction of $\theta_0\theta_1 \cdots \theta_m$ to variables in G

Example

```
plus(0,X,X).
plus(s(X),Y,s(Z)) :- plus(X,Y,Z).
times(0,X,0).
times(s(X),Y,Z) :- times(X,Y,U), plus(U,Y,Z).
```

```
:- times(X,X,Y)
```

SLD-refutation

G_0: `:- times(X,X,Y)`
C_0: `times(s(X0),Y0,Z0) :- times(X0,Y0,U0), plus(U0,Y0,Z0)`
θ_0: `X ↦ s(X0), Y0 ↦ s(X0), Z0 ↦ Y`

G_1: `:- times(X0,s(X0),U0), plus(U0,s(X0),Y)`
C_1: `times(0,X1,0).`
θ_1: `X0 ↦ 0, X1 ↦ s(0), U0 ↦ 0`

G_2: `:- plus(0,s(0),Y)`
C_2: `plus(0,X2,X2).`
θ_2: `X2 ↦ s(0), Y ↦ s(0)`

G_3: $\square$ computed answer substitution: `X ↦ s(0), Y ↦ s(0)`

Definition

- a selection function selects the next goal G in goal clause, where resolution is attempted
- Prolog's selection function proceeds left to right

Theorem

$\forall$ *logic programs P and goal clause G*
$\forall$ *computed answer substitutions θ*
$\forall$ *selection functions $\mathcal{S}$*
$\exists$ *computed answer substitution θ' using $\mathcal{S}$*
such that θ' is at least as general as θ (with respect to variables in G)

Search or SLD Trees

Definition

a search tree (aka SLD tree) of a goal G is a tree T such that

- the root of T is labelled with G
- the nodes of T are labelled with conjunctions of goals, where one goal is selected (wrt a selection function)
- $\exists$ edge from node N for each clause, whose head unifies with the selected goal
- edges are labelled with (partial) answer substitutions
- leaves are success nodes, if the empty goal (denoted by $\Box$) has been reached or failure nodes otherwise

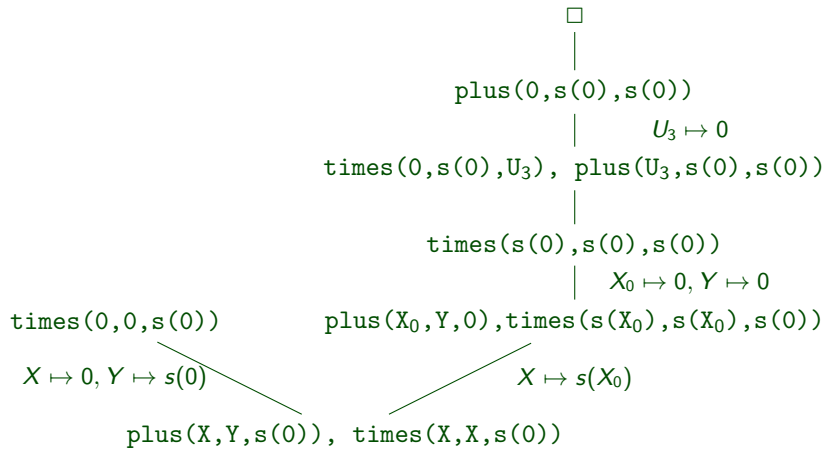
Remark

a search tree captures all SLD derivations wrt a given selection function

Example (cont'd)

```
plus(0,X,X).
plus(s(X),Y,s(Z)) :- plus(X,Y,Z).

times(0,X,0).
times(s(X),Y,Z) :- times(X,Y,U),
                   plus(U,Y,Z).
```



NB: search trees are a tree representation of SLD-derivations

Proof Trees

Definitions

- a proof tree for a program P and a goal G is a tree, whose nodes are goals and whose edges represent reduction of goals
- the root is the query G
- the edges are labelled with (partial) answer substitutions
- a proof tree for a conjunction of goals $G_1, \ldots, G_n$ is the set of all proof trees for G_i

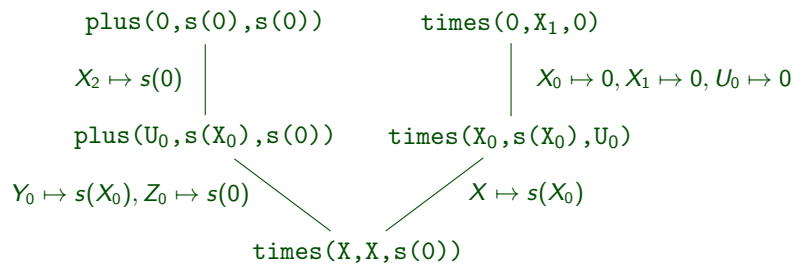
Remark

a proof tree is a different representation of one successful solution represented by a search tree

Example (cont'd)

```
plus(0,X,X).
plus(s(X),Y,s(Z)) :- plus(X,Y,Z).

times(0,X,0).
times(s(X),Y,Z) :- times(X,Y,U),
                   plus(U,Y,Z).
```



Semantics

Definitions

- the Herbrand universe for a program P is the set of all closed terms built from constants and function symbols appearing in the program
- the Herbrand base is the set of all ground goals formed from predicates in P and terms in the Herbrand universe
- an interpretation is a subset of the Herbrand base
- an interpretation I is a model if it is closed under rules:

  $$\forall \text{ rules } A :- B_1, \ldots, B_n: \quad \text{if } B_1, \ldots, B_n \in I, \text{ then } A \in I$$

- the minimal model of P is the intersection of all models

Theorem

the minimal model is unique

Declarative, Operational, and Denotational Semantics

Definition

- the declarative semantics of P (aka its meaning) is the minimal model of P
- we also say that the meaning of a logic program P, is the set of (ground unit) goals deducible from P

Definitions

the operational semantics describes the meaning of a program procedurally

Definition

the denotational semantics assign meanings to programs based on associating with the program a function over the domain computed by the program

Correctness and Completeness

Definition

the intended meaning of a logic program is a set of ground facts

Definition

a program P is called

- correct with respect to the intended meaning M, if the meaning of P is a subset of M
- complete if the intended meaning M is a subset of the meaning of P