Inference Rules and Complexity

David McAllester
AT&T Labs-Research

Outline

- bottom-up logic programming
  "Inferring Recursive Data Types" http://www.research.att.com/ dmac

- descriptive complexity theory
  Givan and McAllester KR92

- solving quasi-word problems (locality)
  McAllester JACM 93
  Givan and McAllester KR92
  Basin and Ganzinger LICS 96

Inference Rules

An *inference rule* is a first order Horn clause, i.e., it has the form

$$\frac{A_1, \ \ldots, \ A_2}{C}$$

where A_i and C are first order atoms (predicates applied to terms).

For example:

$$\frac{x < y, \quad y < z}{x < z} \qquad\qquad \frac{True(\Delta \vee \neg\Phi), \quad True(\Phi \vee \Psi)}{True(\Delta \vee \Psi)}$$

Bottom-Up Logic Programming

For $\boldsymbol{R}$ a set of rules, Σ a set of ground atoms, define

$$C_R(\Sigma)$$

to be the set of ground atoms derivable from Σ using rules in $\boldsymbol{R}$.

$\boldsymbol{R}$ is *terminating* if $C_R(\Sigma)$ is finite for any finite Σ.

Context Free Parsing

The initial data base contains a grammar (assertions of the form $A \to a$ and $A \to BC$) and an input string represented as an assertion of the form

$$\texttt{INPUT}(\texttt{CONS}(a_1,\ \texttt{CONS}(a_2,\ \ldots)))$$

The Cocke-Kasimi-Younger (CKY) parsing algorithm is defined by the following rules.

$\texttt{PARSES}(u,\ i,\ j)$ means that the substring from i to j parses as non-terminal u.

$$\frac{\begin{array}{l} U \to a \\ \texttt{INPUT(CONS(}a\texttt{, }j\texttt{))} \end{array}}{\texttt{PARSES(}U\texttt{, CONS(}a\texttt{, }j\texttt{), }j\texttt{)}}$$

$$\frac{\begin{array}{l} A \to BC \\ \texttt{PARSES(}B\texttt{, }i\texttt{, }j\texttt{)} \\ \texttt{PARSES(}C\texttt{, }j\texttt{, }k\texttt{)} \end{array}}{\texttt{PARSES(}A\texttt{, }i\texttt{, }k\texttt{)}}$$

$$\frac{\texttt{INPUT(CONS(X, Y))}}{\texttt{INPUT(Y)}}$$

Sophisticated Evaluation

Definition: A *prefix firing* of a rule

$$\frac{A_1, \; \ldots, \; A_n}{C}$$

from premise set Σ is a ground instance $A'_1, \ldots A'_j$ of the antecedents A_1, …, A_j where $j \leq n$ such that each A'_i is a member of $C_R(\Sigma)$.

Definition: A rule is *antecedent-bound* if every variable in the conclusion appears in some antecedent.

Let R be any antecedent-bound set of rules and Σ any set of ground atomic fomulas given in dag form.

Assume unit time hash table operations.

Theorem [McAllester "Inferring Recursive Types"]: For fixed R, $C_R(\Sigma)$ can be computed in time proportional to the (dag) size of Σ plus the number of prefix firings (of R from Σ).

The CKY algorithm takes $O(n^3)$ Time

There are only $O(n^3)$ prefix firings of

$$\frac{\begin{array}{l} A \to BC \\ \mathtt{PARSES}(B,\ i,\ j) \\ \mathtt{PARSES}(C,\ j,\ k) \end{array}}{\mathtt{PARSES}(A,\ i,\ k)}$$

Proving the Evaluation Theorem

Consider the following source to source transformations on bottom-up logic programs.

$$A_1,\ \ldots,\ P(t_1,\ \ldots,\ t_n),\ \ldots, A_n/C \quad \Rightarrow$$

$$P(t_1,\ \ldots,\ t_n)/Q(x_1,\ \ldots,\ x_m), \quad A_1,\ \ldots,\ Q(x_1,\ \ldots,\ x_m),\ \ldots, A_n/C$$

$$A_1,\ A_2,\ A_3,\ \ldots A_n/\ C \quad \Rightarrow$$

$$A_1,\ A_2/P(x_1,\ \ldots,\ x_n), \quad P(x_1,\ \ldots,\ x_n),\ A_3,\ \ldots,\ A_n/C$$

$$P(x_1,\ \ldots,\ x_n,\ y_1,\ \ldots,\ y_k)),\ Q(y_1,\ \ldots,\ y_k,\ z_1,\ \ldots,\ z_m)/C \quad \Rightarrow$$

$$P(x_1,\ \ldots,\ x_n,\ y_1,\ \ldots,\ y_k)/P'(f(x_1,\ \ldots,\ x_n),\ g(y_1,\ \ldots\ y_k))$$

$$Q(y_1,\ \ldots,\ y_k,\ z_1,\ \ldots,\ z_m)/Q'(g(y_1,\ \ldots,\ y_k),\ u(z_1,\ \ldots\ z_m))$$

$$P'(x,y),\ Q'(y,z)/R(x,y,z)$$

$$R(f(x_1,\ \ldots,\ x_n),\ g(y_1,\ \ldots,\ y_k),\ u(z_1,\ \ldots,\ z_m))/C$$

These transformations yield rules where the only multiple antecedent rules are of the form $P(x,\ y), Q(y,\ z)/R(x,\ y,\ z)$. The proof is easy for such rules.

λ-calculus Operational Semantics

$$e ::= x \mid 0 \mid (e_1\ e_2) \mid \lambda x.e$$

$$\mathtt{EVAL}(e,\ \emptyset)$$

$$\frac{\mathtt{EVAL}(\lambda x.e,\ \rho)}{\langle \lambda x.e,\ \rho\rangle \to \langle \lambda x.e,\ \rho\rangle}$$

$$\frac{\mathtt{EVAL}(x,\ \rho)}{\langle x,\ \rho\rangle \to \rho(x)}$$

$$\frac{\mathtt{EVAL}(0,\ \rho)}{\langle 0,\ \rho\rangle \to 0}$$

$$\frac{\mathtt{EVAL}((f\ w),\ \rho) \quad \langle f,\ \rho\rangle \to 0}{\mathtt{ERROR}}$$

$$\frac{\mathtt{EVAL}((f\ w),\ \rho)}{\mathtt{EVAL}(f,\ \rho), \quad \mathtt{EVAL}(w,\ \rho)}$$

$$\frac{\mathtt{EVAL}((f\ w),\ \rho) \quad \langle f,\ \rho\rangle \to \langle \lambda x.u,\ \sigma\rangle \quad \langle w,\ \rho\rangle \to v}{\langle (f\ w),\ \rho\rangle \to \langle u,\ \sigma[x := v]\rangle}$$

$$\frac{\langle u,\ \rho\rangle \to \langle w,\ \sigma\rangle}{\mathtt{EVAL}(w,\ \sigma)}$$

$$\frac{\langle u,\ \rho\rangle \to \langle w,\ \sigma\rangle \quad \langle w,\ \sigma\rangle \to v}{\langle u,\ \rho\rangle \to v}$$

Set Based Abstraction

$$\frac{\begin{array}{l}\mathtt{EVAL}(u,\ \rho)\\ \mathtt{EVAL}(u,\ \gamma)\\ \langle u,\ \rho\rangle \to v\end{array}}{\langle u,\ \gamma\rangle \to v}$$

Set Based Analysis

$$\frac{\texttt{EVAL}((f\ w))}{\texttt{EVAL}(f),\quad \texttt{EVAL}(w)}$$

$$\frac{\texttt{EVAL}(\lambda x.e)}{\lambda x.e \to \lambda x.e}$$

$$\frac{\texttt{EVAL}(0)}{0 \to 0}$$

$$\frac{\texttt{EVAL}((f\ w)),\quad f \to 0}{\texttt{ERROR}}$$

$$\frac{\begin{array}{l}\texttt{EVAL}((f\ w))\\ f \to \lambda x.u\\ w \to v\end{array}}{x \to v,\quad (f\ w) \to u}$$

$$\frac{u \to w}{\texttt{EVAL}(w)}$$

$$\frac{\begin{array}{l}u \to w\\ w \to v\end{array}}{u \to v}$$

Explicit Locality

Definition: A rule is *explicity local* if every term in the conclusion is contained in some antecedent.

explicitly local rules:

$$\frac{\begin{array}{l} U \rightarrow a \\ \mathtt{INPUT(CONS(}a\mathtt{,}\ j\mathtt{))} \end{array}}{\mathtt{PARSES(}U\mathtt{,\ CONS(}a\mathtt{,}\ j\mathtt{),}\ j\mathtt{)}}$$

$$\frac{\begin{array}{l} A \rightarrow BC \\ \mathtt{PARSES(}B\mathtt{,}\ i\mathtt{,}\ j\mathtt{)} \\ \mathtt{PARSES(}C\mathtt{,}\ j\mathtt{,}\ k\mathtt{)} \end{array}}{\mathtt{PARSES(}A\mathtt{,}\ i\mathtt{,}\ k\mathtt{)}}$$

$$\frac{\mathtt{INPUT(CONS(X,\ Y))}}{\mathtt{INPUT(Y)}}$$

rules not explicitly local:

$$\frac{\mathtt{NATNUM(X)}}{\mathtt{NATNUM(SUCCESSOR(X))}}$$

Capturing P

Definition: R *accepts* e if $\texttt{INPUT}(e) \vdash_R \texttt{Accept}(e)$.

Theorem [Givan and McAllester KR92]: A dag expression language over a finite signature is polynomial time recognizable if and only if it is the language accepted by an explicitly local rule set.

Capturing coNP, PSPACE, and TIME $2^{O(n)}$

For "explicitly local" programs we have the following.

Language Feature	Complexity CLass
Disjunction	CoNP
Disjunction + Negation by Failure	PSPACE
ACI matching	$2^{O(n)}$ Time

Appropriate definitions can be found in the slides for a UMass talk
http://www.research.att.com/ dmac/rules-slides.html

Quasi-Word Problems

The word problem for a set of equations E is the problem of determining membership (of an equation) in the thoery of E.

The quasi-word problem for a rule set R is the problem of determining membership (of a rule) in the theory of R.

Equivalently, the quasy-word problem for R is the problem of determining $\Sigma \ \vdash_R \ \Phi$ for arbitrary Σ and Φ.

Consider the theory of lattices with a monotone operation:

$$x \leq x$$

$$x \leq y,\ y \leq z \Rightarrow x \leq z$$

$$\begin{array}{rcl} x & \leq & x \vee y \\ y & \leq & x \vee y \\ x \leq z,\ y \leq z & \Rightarrow & x \vee y \leq z \end{array} \qquad \begin{array}{rcl} x \wedge y & \leq & x \\ x \wedge y & \leq & y \\ z \leq x,\ z \leq y & \Rightarrow & z \leq x \wedge y \end{array}$$

$$x \leq y \Rightarrow f(x) \leq f(y)$$

The quasi-word problem for lattices with monotone operations is cubic time decidable.

Implicit Locality

Definition: R is *implicitly local* if $\Sigma \vdash_R \Phi$ implies that there exists a "local" derivation, i.e., one in which every term in the derivation occurs in either Φ or Σ.

The term "local" will mean implicitly local.

Proving Implicit Locality

Consider

$$x \leq y,\ y \leq z \Rightarrow x \leq z$$
$$x \leq y \Rightarrow f(x) \leq f(y)$$

Anth question of whether $\Sigma\ \vdash_R\ \Phi$.

Consider the assertions which become provable when we add $f(t)$.

Every new assertion is one of the following four types.

$$x \leq t \Rightarrow f(x) \leq f(t)$$
$$t \leq x \Rightarrow f(t) \leq f(x)$$

$$x \leq t,\ \ y \leq f(x)\ \ \Rightarrow y \leq f(t)$$
$$t \leq x,\ \ f(x) \leq y\ \ \Rightarrow f(t) \leq y$$

From these "invariants" it can be shown that no "feedback inferences" can occur when a new term is added.

Locality Completion

$$x \leq y,\ y \leq z \Rightarrow x \leq z$$
$$x \leq y \Rightarrow f(x,\ z) \leq f(y,\ z)$$
$$x \leq y \Rightarrow f(z,\ x) \leq f(z,\ y)$$

R is not local.

$$a \leq a',\ b \leq b' \ \vdash_R \ f(a,\ b) \leq f(a',\ b')$$

The automated attempt to prove locality generates this counter-example. Suppose $f(t, w)$ is new.

$$x \leq t \Rightarrow f(x,\ w) \leq f(t,\ w)$$
$$w \leq x \Rightarrow f(t,\ w) \leq f(t,\ y)$$

$$x \leq t,\ \ w \leq x \Rightarrow f(x,\ w) \leq f(t,\ y)$$

When a counter example is found, add it to the rule set and continue.

Theorem: The (possibly infinite) rule set produced by this process is (implicitly) local.

Ordered Resolution

Consider classical binary resolution.

$$\frac{\Delta \vee A, \quad \neg A \vee \Gamma}{\Delta \vee \Gamma}$$

Fix a well founded total order on atomic formulas.

The above reaolution is called *ordered* if A is the maximum atom in both clauses.

Completeness Theorem [Robinson?]: If an (infinite) set of ground clauses is unsatisfiable then the empty clause can be derived by ordered resolution.

Proof: Suppose that the set Σ of all clauses derivable by ordered resolution does not contain the empty clause. Build a truth assignment by transfinite induction on the total order. At each step there exists a truth value which does not violate any clause — if $\neg A$ violates a clause $\Delta \vee A$ and A violates a clause $\neg A \vee \Gamma$ then Σ contains the earlier clause $\Delta \vee \Gamma$ but no such clause was violated earlier.

Saturation

Again fix a well founded total order on ground atoms.

A clause Δ is said to *precede* an atom A if every atomic formula in Δ is less than A.

A clause set Σ is called *implicitly saturated* under ordered resolution if for every clause Δ derivable by ordered resolution on atom A we have that Δ *follows from* the clauses in Σ that precede A.

Implicit Completeness Theorem [Bachmair and Ganzinger 90] If Σ is implicitly saturated and does not contain the empty clause then Σ is satisfiable.

Proof: Same as before.

Relevance

Again fix a well founded total order on atoms.

A clause Δ is said to be *relevant* to an atom A if every atom in Δ is less than or equal to A.

Relevance Theorem[Basin and Ganzinger 96]: If Σ is implicitly saturated with respect to ordered resolution then for any Boolean formula Ψ we have that if $\Sigma \models \Psi$ then $\Sigma' \models \Psi$ where Σ' is the set of clauses in Σ relevant to some atom in Ψ.

Proof: By the proofs of the preceding theorems any model of Σ' can be extended to a model of Σ. Hence if $\Sigma' \not\models \Psi$ then $\Sigma \not\models \Psi$.

Saturation Implies Locality

A set of (ground) clauses Σ is said to be implicitly saturated with respect to a class of orders if it is implicitly saturated under every order in that class.

A total well founded order $\prec$ on atoms is called *subterm compatible* if there exists a total well founded order on terms such that every term is greater than its proper subterms and for any atoms A and B if the largest term in A is greater than the largest term in B then $A \succ B$

Theorem[Basin and Ganzinger 96] If R is a rule set such that the set of ground instances of rules in R is implicity saturated over the class of all subterm compatible orders then R is implicitly local.

Proof: Let R satisfies these conditions. We must show that if $\Sigma \vdash_R \Phi$ then there exists a local proof of Φ from Σ. An atom A will be called local to Σ and Φ if all terms in A appear in Σ or Φ. There exists a subterm compatible ordering where all local atoms are smaller than all nonlocal atoms. The relevance theorem now implies that we need only consider local instances of R.

Saturation is Decidable

We now fix a class of allowed well founded partial orders on atoms.

A resultion derivation of a nonground clause Δ by resolution on a nonground atom A is called *redundant* in a nonground clause set Σ if for all ground substitutions σ and all orders in the allowed class such that $\Delta\sigma$ precedes $A\sigma$ we have that $\Delta\sigma$ follows from ground instances of clauses in Σ that precede $A\sigma$.

Theorem [McAllester 97]: For the class of subterm compatible orders, redundancy is decidable.

Proof Idea: We do a case analysis over the sets of terms in A and Δ which can all be simultaneously equal to each other and equal the maximum term in the ordered resolution step. In each case there is a ground substitution and an ordering which minimizes the set of clauses preceding A. If this minimal set of clauses preceding A implies Δ in each case then the resolution is redundant.

Corollary: It is possible to determine whether a finite set Σ of nonground clauses is saturated over the class of subterm-compatible orderings.

Locality Does Not Imply Saturation

Here we use "saturation" for saturation over the class of subterm compatible orders.

Locality is undecidable [Givan and McAllester KR92].

Saturation is decidable and implies locality.

Therefore locality must not imply saturation.

An Example

The following is local.

$$x \leq y,\ y \leq z \Rightarrow x \leq z$$
$$x \leq y \Rightarrow f(x) \leq f(y)$$

The proof of locality uses the following "invariants"

$$\begin{aligned} x \leq z,\ \ y \leq f(x) \ &\Rightarrow y \leq f(z) \\ z \leq x,\ \ f(x) \leq y \ &\Rightarrow f(z) \leq y \end{aligned}$$

But saturation yields (at least) the full rule set

$$x \leq y,\ y \leq z \Rightarrow x \leq z$$
$$x \leq y \Rightarrow f(x) \leq f(y)$$
$$\begin{aligned} x \leq z,\ \ y \leq f(x) \ &\Rightarrow y \leq f(z) \\ z \leq x,\ \ f(x) \leq y \ &\Rightarrow f(z) \leq y \end{aligned}$$

In Locality completion the invariant set can grow arbitrarily large without generating a new rule in the local set — locality is undecidable.

But in saturation each new "invariant" is a violation of saturation and the set being saturated is enlarged.

Main Points

- The sophisticated evaluation theorem gives the order of run time for CKY parsing and control flow analysis (and a host of other algorithms).

- Practical programs, such as CKY parsing and flow analysis, are explicitly local.

- All polynomial time problems can be solved with explicitly local rule sets.

- Algorithms for proving implicit locality can solve natural mathematical quasi-word problems such as the quasi-word problem for lattices with monotone operations.

- Ordered resolution methods (saturation) provide a clean theoretical approach to implicit locality.

- It seems possible that combining insights from the two approaches will yield further improvements in both implicit locality testing and ordered resolution algorithms.