1999

p59/10
LCP

Logic & Proof 1

How do Prolog clauses differ from the clauses used by general-purpose resolution theorem provers? [2 marks].

Describe the series of resolutions that is performed by a Prolog interpreter when it is supplied with a program and a set of clauses. To illustrate your answer, explain how the following program executes when presented with the goal $\leftarrow Q(f(f(f(f(a)))))$:

$$Q(a) \leftarrow \tag{1}$$
$$Q(f(a)) \leftarrow \tag{2}$$
$$Q(f(f(x))) \leftarrow Q(f(x)), Q(x) \tag{3}$$

[7 marks]

Considering the program and the goal $\leftarrow Q(f(f(f(f(a)))))$ now as a set of clauses, derive the empty clause using general resolution. (Full credit requires finding the shortest derivation.) [6 marks]

Consider the set of clauses consisting of the program given above and the goal $\leftarrow Q(\underbrace{f(\cdots f(}_{n}(a))\cdots))$. Let $p(n)$ be the number of steps executed by a Prolog interpreter when given those clauses. Let $r(n)$ be the minimum number of steps required to derive the empty clause from those clauses using general resolution. Compare the approximate growth rates of $p(n)$ and $r(n)$ as n increases, and explain any difference you find. [5 marks]

Model Answer

(Much of this is from the notes.) A general clause is a set of literals, representing a disjunction, where a *literal* is either an atom or a negated atom. Each clause of a Prolog program must have precisely one positive literal, which is called the *head*; any other literals must be negative. The Prolog interpreter manages the outstanding set of goals as another clause, the *goal clause*, which consists entirely of negative literals.

The Prolog interpreter considers both the outstanding goals and program clauses from left to right. It attempts to unify the leftmost goal with the head of each program clause in turn. If unification succeeds, it resolves the two clauses (the goal clause and the program clause), yielding a new goal clause. In effect, the goal has been replaced by subgoals given by the program clause. These are recursively tackled until no subgoals are left. If the interpreter reaches a dead end, meaning that a goal cannot be resolved with any program clause, then it backtracks, undoing the most recent resolution and attempting to find another unifiable clause.

The program executes as follows. The initial goal clause is $\leftarrow Q(f(f(f(f(a)))))$.

Resolving with clause (3) yields the new goal clause $\leftarrow Q(f(f(f(a)))), Q(f(f(a)))$

Resolving with clause (3) yields the new goal clause $\leftarrow Q(f(f(a))), Q(f(a)), Q(f(f(a)))$

Resolving with clause (3) yields the new goal clause $\leftarrow Q(f(a)), Q(a), Q(f(a)), Q(f(f(a)))$

Resolving with clause (2) yields the new goal clause $\leftarrow Q(a), Q(f(a)), Q(f(f(a)))$

Resolving with clause (1) yields the new goal clause $\leftarrow Q(f(a)), Q(f(f(a)))$

Resolving with clause (2) yields the new goal clause $\leftarrow Q(f(f(a)))$

A further three resolutions yield the empty clause. The Prolog execution takes nine steps. There are no choice points. Note that Prolog does not delete repeated literals.

With general resolution, things are more interesting. We can resolve clause (3) with itself, after first renaming the variable x in one copy, taking $Q(f(f(x)))$ and $\neg Q(f(x'))$ as the complementary literals:

$$\begin{gathered} Q(f(f(x'))) \leftarrow Q(f(x')), Q(x') \\ Q(f(f(x))) \leftarrow Q(f(x)), Q(x) \end{gathered}$$

Now x' is instantiated to $f(x)$ and the resulting clause is

$$Q(f(f(f(x)))) \leftarrow Q(f(x)), Q(f(x)), Q(x)$$

or after eliminating repeated literals

$$Q(f(f(f(x)))) \leftarrow Q(f(x)), Q(x)$$

Resolving the negative literal $Q(f(x))$ with clause (3) similarly yields

$$Q(f(f(f(f(x))))) \leftarrow Q(f(x)), Q(x)$$

This can be resolved with the goal clause, yielding

$$\leftarrow Q(f(a)), Q(a)$$

Two further resolutions with clauses (2) and (1) yield the empty clause. The proof has taken five steps. [Any student who surpasses this will get extra marks!]

Turning to $p(n)$ and $r(n)$, it is not hard to see that $p(n)$ is exponential in n. Writing $\tilde{n}$ for $\underbrace{f(\cdots f(}_{n}(a))\cdots)$, the Prolog goal $Q(n \tilde{+} 2)$ gives rise to new subgoals $Q(n \tilde{+} 1)$ and $Q(\tilde{n})$, which require more than twice as many steps as $Q(\tilde{n})$ alone. So $p(n+2) > 2p(n)$, and $p(n)$ is bounded below by $2^{n/2}$. (More precisely, it is bounded below by the nth Fibonacci number.) General resolution does much better. Even discounting the trick used in the previous solution, two resolutions starting from clause (3) take $Q(\tilde{0}) \leftarrow$ and $Q(\tilde{1}) \leftarrow$ to $Q(\tilde{2}) \leftarrow$. Two further resolutions from the last two clauses and (3) yield $Q(\tilde{3}) \leftarrow$, and so forth. So $r(n)$ is at worst linear in n. The difference is explained by Prolog's strict evaluation order, which prevents the bottom-up derivation just described.