Concurrent Algorithms in Linear Logic Programming

Robert J. Simmons (Work with Frank Pfenning)
February 14, 2008
Concurrent Algorithms in Linear Logic Programming

Robert J. Simmons (Work with Frank Pfenning)
February 14, 2008

(available from http://www.cs.cmu.edu/~fp)
Inference Rules

In Meld, Datalog, XSB:
“c :- a, b.”
Inference Rules

\[ \begin{align*}
a & \quad \text{true} \\
b & \quad \text{true} \\
c & \quad \text{true}
\end{align*} \]
Inference Rules

\[ \text{path}(X,Y) \quad \text{true} \]
\[ \text{path}(Y,Z) \quad \text{true} \]
\[ \text{path}(X,Z) \quad \text{true} \]
Inference Rules

\[ \text{path}(X,Y) \]
\[ \text{car\_is\_at}(C,X) \]
\[ \hline \]
\[ \text{car\_is\_at}(C,Y) \]
Inference Rules

\[
\text{path}(X,Y) \rightarrow \text{true}
\]

\[
\text{car\_is\_at}(C,X)
\]

\[
\text{car\_is\_at}(C,Y)
\]
Inference Rules

\[
\text{path}(X,Y) \text{ true} \\
\text{car_is_at}(C,X) \ ? \\
\underline{\text{car_is_at}(C,Y)}
\]
Inference Rules

path(X,Y) \text{true}

car\_is\_at(C,X) ?

\hline
car\_is\_at(C,Y) ?
Inference Rules

\[
\text{path}(X,Y) \quad \text{persistent} \\
\text{car\_is\_at}(C,X) \quad ? \\
\hline \\
\text{car\_is\_at}(C,Y) \quad ?
\]
Inference Rules

\[ \text{path}(X,Y) \quad \text{persistent} \]
\[ \text{car\_is\_at}(C,X) \quad \text{ephemeral} \]

\[ \frac{}{\text{car\_is\_at}(C,Y)} ? \]
Inference Rules

\[ \text{path}(X,Y) \quad \text{persistent} \]
\[ \text{car_is_at}(C,X) \quad \text{ephemeral} \]
\[ \hline \]
\[ \text{car_is_at}(C,Y) \quad \text{ephemeral} \]
Inference Rules

path(X, Y) \textit{persistent}

\[\text{car\_is\_at}(C, X) \textit{ephemeral}\]

\[\text{car\_is\_at}(C, Y) \textit{ephemeral}\]

(Linear Logic!)
Persistent propositions describe permanent truth the system

path(a,c), path(c,a), path(a,b), path(b,c), path(c,b), path(b,a)
Ephemeral propositions describe current state of system.
Ephemeral rules describe potential changes to system.
Ephemeral rules describe potential changes to system.
Ephemeral rules describe potential changes to system
unvisited(root)  

visited(root)  

edge(X,Y)  
visited(X)  
unvisited(Y)  
visited(Y)  
tree(X,Y)
Ephemeral propositions are “mutual exclusion locks”

\[
\begin{align*}
\text{unvisited}(\text{root}) & \quad \text{visited}(\text{root}) \\
\text{unvisited}(Y) & \quad \text{visited}(Y) \\
\text{edge}(X,Y) & \quad \text{tree}(X,Y)
\end{align*}
\]
\[
\text{item}(X) \\
\text{list}(L) \\
\overline{\text{list}(\text{cons}(X,L))}
\]
Ephemeral propositions are “state objects”

\[
\text{item}(X) \\
\text{list}(L) \\
\frac{}{\text{list}(\text{cons}(X,L))}
\]
Logical account of concurrency

Ephemeral propositions are “mutual exclusion locks” “state objects” ...what else?

Represent locality? (@modal logic)
Global synchronization? (temporal logic)
Declarative model of programming can allow more optimizations

Executable specifications!

Huge success with purely-persistent Datalog programming (Whaley et. al)

Spiral too, it seems!

Both are domain-specific (whether de facto or de jure)