Focus - metaknowledge

Examples
    Dynamic Systems
    Beliefs
    Boolean Logic
Example - Blocks World
Blocks World

A

B

C

D

E
External Actions

\[ u(a,b) \]

\[ u(d,e) \]
Symbols: $a, b, c, d, e$

Unary Predicates:
- \textit{clear} - blocks with no blocks on top
- \textit{table} - blocks on the table

Binary Predicates:
- \textit{on} - pairs of blocks in which first is on the second
- \textit{above} - pairs in which first block is above the second

$u(x,y)$ - means that $x$ is moved from $y$ to the table.
$s(x,y)$ - means that $x$ is moved from the table to $y$. 
Operations:
\[ u(x, y) \text{ means that } x \text{ is moved from } y \text{ to the table.} \]
\[ s(x, y) \text{ means that } x \text{ is moved from the table to } y. \]

Physics:
\[ \text{true(table}(X), \text{do}(u(X,Y),S)) \] 
\[ \quad \text{:- true(clear}(X),S) \& \text{ true(on}(X,Y),S) \]

\[ \text{true(clear}(Y), \text{do}(u(X,Y),S)) \] 
\[ \quad \text{:- true(clear}(X),S) \& \text{ true(on}(X,Y),S) \]

\[ \text{true}(P, \text{do}(A,S)) \] 
\[ \quad \text{:- add}(P,S) \]

\[ \text{true}(P, \text{do}(A,S)) \] 
\[ \quad \text{:- true}(P,S) \& \neg \text{del}(P, \text{do}(A,S)) \]
Operations:
\( u(x, y) \) means that \( x \) is moved from \( y \) to the table.
\( s(x, y) \) means that \( x \) is moved from the table to \( y \).

Physics:
\[
\text{true}(\text{table}(X), \text{do}(u(X, Y), S)) : - \\
\text{true}(\text{clear}(X), S) & \\
\text{true}(\text{on}(X, Y), S)
\]

\[
\text{true}(\text{clear}(Y), \text{do}(u(X, Y), S)) : - \\
\text{true}(\text{clear}(X), S) & \\
\text{true}(\text{on}(X, Y), S)
\]

\[
\text{true}(P, \text{do}(A, S)) : - \\
\text{add}(P, S)
\]

\[
\text{true}(P, \text{do}(A, S)) : - \text{true}(P, S) & \neg \text{del}(P, \text{do}(A, S))
\]
Operations:
\[ u(x, y) \] means that \( x \) is moved from \( y \) to the table.
\[ s(x, y) \] means that \( x \) is moved from the table to \( y \).

Physics:
\[
\text{add}(\text{table}(X), \text{do}(u(X, Y), S)) \leftarrow
\text{true}(\text{clear}(X), S) \&
\text{true}(\text{on}(X, Y), S)
\]

\[
\text{add}(\text{clear}(Y), \text{do}(u(X, Y), S)) \leftarrow
\text{true}(\text{clear}(X), S) \&
\text{true}(\text{on}(X, Y), S)
\]

\[
\text{del}(\text{on}(X, Y), \text{do}(u(X, Y), S)) \leftarrow
\text{true}(\text{clear}(X), S) \&
\text{true}(\text{on}(X, Y), S)
\]

\text{true}(P, \text{do}(A, S)) \leftarrow
Dataset:

```
clear(c)       on(c,a)       table(b)
clear(d)       on(a,b)       table(e)
on(d,e)
```

Transition Rule:

```
u(X,Y) & clear(X) & on(X,Y)  
   ==> ~on(X,Y) & table(X) & clear(Y)
```

Result of $u(c,a)$:

```
clear(a)       on(a,b)       table(b)
clear(c)       on(d,e)       table(c)
clear(d)       table(e)
```
Example - Boolean Logic
Basic idea: represent expressions in Boolean Logic as Logic Programming terms and write rules to define basic concepts of Boolean Logic.

NB: We can extend to defining Logic Programming within Logic Programming as well. The formalization is messier, and some nasty problems need to be handled (notably paradoxes).
Symbols (propositions)

$p, q, r$
Syntactic Metavocabulary

Object Constants (propositions)
  \( p, q, r \)

Function constants
  \( \text{not}(p) \quad \text{if}(p,q) \)
  \( \text{and}(p,q) \quad \text{iff}(p,q) \)
  \( \text{or}(p,q) \)
Object Constants (propositions)
  \( p, q, r \)

Function constants
  \( \text{not}(x) \)
  \( \text{if}(x,y) \)
  \( \text{and}(x,y) \)
  \( \text{iff}(x,y) \)
  \( \text{or}(x,y) \)

Relation Constants
  \( \text{proposition}(x) \)
  \( \text{implication}(x) \)
  \( \text{negation}(x) \)
  \( \text{biconditional}(x) \)
  \( \text{conjunction}(x) \)
  \( \text{sentence}(x) \)
  \( \text{disjunction}(x) \)
proposition(p)  
proposition(q)  
proposition(r)  

negation(not(x)) ⇔ sentence(x)  
conjunction(and(x,y)) ⇔ sentence(x) ∧ sentence(y)  
disjunction(or(x,y)) ⇔ sentence(x) ∧ sentence(y)  
implication(if(x,y)) ⇔ sentence(x) ∧ sentence(y)  
equivalence(iff(x,y)) ⇔ sentence(x) ∧ sentence(y)  

sentence(x) ⇔ 

proposition(x) ∨ negation(x) ∨ conjunction(x) ∨ 
disjunction(x) ∨ implication(x) ∨ biconditional(x)
Rules of Inference

And Introduction:

$$\forall x. (\text{sentence}(x) \land \text{sentence}(y) \Rightarrow ai(x,y,\text{and}(x,y)))$$

And Elimination:

$$\forall x. (\text{sentence}(x) \land \text{sentence}(y) \Rightarrow ae(\text{and}(x,y),x))$$
$$\forall x. (\text{sentence}(x) \land \text{sentence}(y) \Rightarrow ae(\text{and}(x,y),y))$$
Validity of Axiom Scemata:

\[
\text{valid}(\text{or}(x,\neg x)) \iff \text{sentence}(x)
\]

Soundness:

\[
\text{proves}(x,y) \iff \text{entails}(x,y)
\]

Deduction Theorem:

\[
\text{proves}(\text{and}(x,y),z) \iff \text{proves}(x,\text{implies}(y,z))
\]
Example - Natural Language Processing
Good Sentences:
   Mary likes Pat.
   Mary likes Pat and Quincy.
   Pat and Quincy like Mary.

Bad Sentences:
   Mary Pat likes.
   Likes and Mary Pat Quincy.
Backus Naur Form (BNF)

<sentence> ::= <np> <vp>
<np> ::= <noun>
<np> ::= <noun> "and" <noun>
<vp> ::= <verb> <np>
<noun> ::= "mary" | "pat" | "quincy"
<verb> ::= "like" | "likes"
sentence(Z) :- np(X) & vp(Y) & append(X,Y,Z)
sentence(Z) :- np(X) & vp(Y) & append(X,Y,Z)

np(X) :- noun(x)
np(W) :-
    noun(X) & noun(Y) &
    append(X,and,Y) & append(Z,Y,W)
Logical Grammar

\[
\text{sentence}(Z) :- \ np(X) \ & \ \text{vp}(Y) \ & \ \text{append}(X,Y,Z)
\]

\[
\text{np}(X) :- \ \text{noun}(x)
\]

\[
\text{np}(W) :-
\quad \text{noun}(X) \ & \ \text{noun}(Y) \ &
\quad \text{append}(X,\text{and},Y) \ & \ \text{append}(Z,Y,W)
\]

\[
\text{vp}(Z) :- \ \text{verb}(X) \ & \ \text{np}(Y) \ & \ \text{append}(X,Y,Z)
\]
sentence(Z) :- np(X) & vp(Y) & append(X,Y,Z)

np(X) :- noun(x)
np(W) :-
  noun(X) & noun(Y) &
  append(X,and,Y) & append(Z,Y,W)

vp(Z) :- verb(X) & np(Y) & append(X,Y,Z)

noun(mary)
noun(pat)
noun(quincyn)

Logical Grammar

sentence(Z) :- np(X) & vp(Y) & append(X,Y,Z)

np(X) :- noun(x)
np(W) :-
    noun(X) & noun(Y) &
    append(X,and,Y) & append(Z,Y,W)

vp(Z) :- verb(X) & np(Y) & append(X,Y,Z)

noun(mary)
noun(pat)
noun(quincy)

verb(like)
verb(likes)
Examples

Sentences:
√ Mary likes Pat.
√ Mary likes Pat and Quincy.
√ Pat and Quincy like Mary.

Not Sentences:
× Mary Pat likes.
× Likes and Mary Pat Quincy.
Sentences:
  Mary likes Pat.
  Mary likes Pat and Quincy.
  Pat and Quincy like Mary.

Allowed but not sentences in natural English:
  Mary like Pat.
  Pat and Quincy likes Mary.

How can we enforce subject-verb number agreement?
Augmented Logical Grammar

sentence(Z) :- np(X,W) & vp(Y,W) & append(X,Y,Z)

np(X,0) :- noun(x)
np(W,1) :-
   noun(X) & noun(Y) &
   append(X,and,Y) & append(Z,Y,W)

vp(Z,W) :- verb(X,W) & np(Y) & append(X,Y,Z)

noun(mary)
noun(pat)
noun(quincy)

verb(like,1)
verb(likes,0)
Augmented Logical Grammar

\[
\text{sentence}(Z) \leftarrow \text{np}(X, W) \land \text{vp}(Y, W) \land \text{append}(X, Y, Z)
\]

\[
\text{np}(X, 0) \leftarrow \text{noun}(x)
\]

\[
\text{np}(W, 1) \leftarrow
\quad \text{noun}(X) \land \text{noun}(Y) \land
\quad \text{append}(X, \text{and}, Y) \land \text{append}(Z, Y, W)
\]

\[
\text{vp}(Z, W) \leftarrow \text{verb}(X, W) \land \text{np}(Y) \land \text{append}(X, Y, Z)
\]

noun(mary)
noun(pat)
noun(quincy)

verb(like, 1)
verb(likes, 0)
Augmented Logical Grammar

sentence(Z) :- np(X,W) & vp(Y,W) & append(X,Y,Z)

np(X,0) :- noun(x)
np(W,1) :- noun(X) & noun(Y) & append(X,and,Y) & append(Z,Y,W)

vp(Z,W) :- verb(X,W) & np(Y) & append(X,Y,Z)

noun(mary)
noun(pat)
noun(quincy)

verb(like,1)
verb(likes,0)
Augmented Logical Grammar

sentence(Z) :- np(X,W) & vp(Y,W) & append(X,Y,Z)

np(X,0) :- noun(x)
np(W,1) :- noun(X) & noun(Y) & append(X,and,Y) & append(Z,Y,W)

vp(Z,W) :- verb(X,W) & np(Y) & append(X,Y,Z)

noun(mary)
noun(pat)
noun(quincy)

verb(like,1)
verb(likes,0)