Logic Programming
Dynamic Systems

Michael Genesereth
Computer Science Department
Stanford University
Programme

Closed Systems

Reactive Systems

Mixed Initiative Systems
The Game of Life
Rules of the Game

(1) Any live cell with two or three live neighbors lives on to the next generation.

(2) Any live cell with fewer than two live neighbors dies (as if caused by underpopulation).

(3) Any live cell with more than three live neighbors dies (as if by overpopulation).

(4) Any dead cell with exactly three live neighbors becomes a live cell (as if by reproduction).
Symbols: \( c_{11}, c_{12}, \ldots \)

Unary Predicates:
- on - cell is live
- cell - true of cells

Binary Predicates:
- neighbor - cells are neighbors
Any live cell with fewer than two live neighbors dies.

update ::
  on(Y) & evaluate(countofall(X, neighbor(X,Y)&on(X)), 0)
  ==> ~on(Y)

update ::
  on(Y) & evaluate(countofall(X, neighbor(X,Y)&on(X)), 1)
  ==> ~on(Y)
Any live cell with more than three live neighbors dies.

update ::
    on(Y) &
    evaluate(min(countofall(X, neighbor(X, Y) & on(X)), 4), 4)
==> ~on(Y)
Transition Rules

Any *dead* cell with *exactly three* live neighbors becomes live.

**update ::**

```
cell(Y) & ~on(Y) &
evaluate(countofall(X,neighbor(X,Y)&on(X)),3),3)
==> on(Y)
```
Example

http://logicprogramming.stanford.edu/examples/gameoflife.html
Blocks World
$u(a,b)$

$u(d,e)$
Change Axioms

Unstacking:

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

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

Stacking:

\[ \text{tr(on}(X,Y), \text{do}(s(X,Y),S)) :\] \\
\[\text{tr(clear}(X),S) \& \text{tr(table}(X),S) \& \text{tr(clear}(Y),S) \]
Frame Axioms for Unstacking

\[ \text{tr(} \text{clear}(U), \text{do}(u(X,Y),S)\text{)} \] :
\[ \text{tr(} \text{clear}(X),S\text{)} \land \text{tr(} \text{on}(X,Y),S\text{)} \land \text{tr(} \text{clear}(U),S\text{)} \]

\[ \text{tr(} \text{table}(U), \text{do}(u(X,Y),S)\text{)} \] :
\[ \text{tr(} \text{clear}(X),S\text{)} \land \text{tr(} \text{on}(X,Y),S\text{)} \land \text{tr(} \text{table}(U),S\text{)} \]

\[ \text{tr(} \text{on}(U,V), \text{do}(u(X,Y),S)\text{)} \] :
\[ \text{tr(} \text{clear}(X),S\text{)} \land \text{tr(} \text{on}(X,Y),S\text{)} \land \text{tr(} \text{on}(U,V),S\text{)} \land \text{distinct}(U,X) \]

\[ \text{tr(} \text{on}(U,V), \text{do}(u(X,Y),S)\text{)} \] :
\[ \text{tr(} \text{clear}(X),S\text{)} \land \text{tr(} \text{on}(X,Y),S\text{)} \land \text{tr(} \text{on}(U,V),S\text{)} \land \text{distinct}(V,Y) \]
Frame Axioms for Stacking

\begin{align*}
\text{tr}(\text{clear}(U), \text{do}(s(X,Y),S)) & : - \\
& \quad \text{tr}(\text{clear}(X),S) \& \text{tr}(\text{table}(X),S) \& \text{tr}(\text{clear}(Y),S) \& \\
& \quad \text{tr}(\text{clear}(U),S) \& \text{distinct}(U,Y)
\end{align*}

\begin{align*}
\text{tr}(\text{table}(U), \text{do}(s(X,Y),S)) & : - \\
& \quad \text{tr}(\text{clear}(X),S) \& \text{tr}(\text{table}(X),S) \& \text{tr}(\text{clear}(Y),S) \& \\
& \quad \text{tr}(\text{table}(U),S) \& \text{distinct}(U,X)
\end{align*}

\begin{align*}
\text{tr}(\text{on}(U,V), \text{do}(s(X,Y),S)) & : - \\
& \quad \text{tr}(\text{clear}(X),S) \& \text{tr}(\text{table}(X),S) \& \text{tr}(\text{clear}(Y),S) \& \\
& \quad \text{tr}(\text{on}(U,V),S)
\end{align*}
Describing States

\[
\begin{align*}
\text{u}(a,b) & \quad \Rightarrow \quad \text{do}(\text{u}(a,b),s) \\
& \quad \Rightarrow \quad \text{tr}(	ext{clear}(a),s) \\
& \quad \quad \text{tr}(	ext{on}(a,b),s) \\
& \quad \quad \text{tr}(	ext{on}(b,c),s) \\
& \quad \quad \text{tr}(	ext{on}(d,e),s) \\
& \quad \quad \ldots
\end{align*}
\]
Describing States

clear(a)
on(a,b)
on(b,c)
on(d,e)
...

clear(a)
table(a)
clear(b)
on(b,c)
on(d,e)
...
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 \( y \).

Operation Definitions:

- \( u(X,Y) :: \)
  - clear(X) & on(X,Y)
  - \( \implies \neg on(X,Y) \land table(X) \land clear(Y) \)

- \( s(X,Y) :: \)
  - table(X) & clear(X) & clear(Y)
  - \( \implies \neg table(X) \land \neg clear(Y) \land on(X,Y) \)
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 \( y \).

Operation Definitions:
\[ u(X,Y) :: 
\begin{align*}
\text{clear}(X) & \land \text{on}(X,Y) \\
\implies \neg\text{on}(X,Y) & \land \text{table}(X) & \land \text{clear}(Y)
\end{align*} \]

\[ s(X,Y) :: 
\begin{align*}
\text{table}(X) & \land \text{clear}(X) & \land \text{clear}(Y) \\
\implies \neg\text{table}(X) & \land \neg\text{clear}(Y) & \land \text{on}(X,Y)
\end{align*} \]
Pelican Hunters