Constraint Logic Programming II
Success and Failure

A *success state* is a state $\langle G \mid C \rangle$ where $G$ is the empty goal and $\text{solv}(C) \neq false$.

A *fail state* is a state $\langle G \mid C \rangle$ where $G$ is the empty goal and $\text{solv}(C) \equiv false$. 
A derivation

$$\langle G_0 \mid C_0 \rangle \Rightarrow \ldots \Rightarrow \langle G_n \mid C_n \rangle$$

is *successful* if $$\langle G_n \mid C_n \rangle$$ is a success state.

The constraint $$\text{simpl}(C_n, \text{vars}(\langle G_0 \mid C_0 \rangle))$$ is said to be an *answer* to the state $$\langle G_0 \mid C_0 \rangle$$

If $$G$$ is a goal, then an *answer* for $$G$$ is an answer to the state $$\langle G \mid \text{true} \rangle$$.

A derivation

$$\langle G_0 \mid C_0 \rangle \Rightarrow \ldots \Rightarrow \langle G_n \mid C_n \rangle$$

is *failed* if $$\langle G_n \mid C_n \rangle$$ is a fail state.
Derivation trees

A *derivation tree* for a goal \( G \) and a program \( P \) is a tree with states as nodes.

The root of the tree is \( \langle G \mid true \rangle \).

The children of each state \( \langle G_i \mid C_i \rangle \) are those states that can be reached in a single derivation step form that state.

A state that has two or more children is called a *choicepoint*. 
Finite Failure

If a state or goal $G$ has a finite derivation tree and all derivations in the tree are failed, $G$ is said to *finitely fail*.

**Example** \texttt{fact(0,2)}

Infinite derivation trees are possible! (You will do this!)

\begin{verbatim}
stupid(X) :- stupid(X).
stupid(1).
\end{verbatim}
Goal Evaluation

A goal is evaluated by performing an in-order (left to right depth first) traversal of the goal’s derivation tree.

Whenever success is encountered the result is returned to the user.

If the user elects to continue execution continues with the tree traversal.

Execution stops either when the user does not ask for more solutions or the tree has been completely traversed.
Simplified Derivation Trees

A state \( \langle G_0 \mid C_0 \rangle \) occurring in a derivation tree for a goal \( G \) can be simplified in the following way.

Let \( V = \text{vars}(G) \) and \( V_0 = \text{vars}(G_0) \).

Compute \( C_1 = \text{simpl}(C_0, V \cup V_0) \).

For each \( x \in V_0 - V \) which appears exactly once in \( C_1 \) in an equation of the form \( x = t \) replace \( x \) everywhere in \( G_0 \) by \( t \) giving \( C_2 \).

Simplify again to get \( C_2 = \text{simpl}(C_1, V \cup \text{vars}(G_2)) \).

The simplified state is \( \langle G_2 \mid C_2 \rangle \)
The CLP Scheme

We have in fact defined a family of languages, depending on

- The constraint domain,

- The constraint solver,

- the constraint simplifier