From generate-and-test to domain reduction

generate-and-test :- node(Node), goal(Node).
node([V1,...,Vn]) :- member(V1,L1),..., member(Vn,Ln).
goal(Node) :- ϕ₁, ϕ₂,..., ϕₘ.

Interleave generation with test
- instantiate variables \( V_i \) incrementally
  node₁ = partial variable instantiation
- test constraints \( ϕ_j \) incrementally
  node₂ = node₁ + constraints to satisfy
- reduce domain sets (lists) \( L_i \)
  node₃ = node₂ + domain sets of uninstantiated variables

Update constraints (e.g. fail) even before instantiating

The Zebra Puzzle

See Wikipedia

Node = partial variable assignment

Pre-process the constraints (before instantiating variables)
Analyze constraints as sets of nodes

\( \text{The Englishman lives in the red house} \)  
(chasing 5 ways)

\[
\begin{array}{l}
\text{red} \\
\text{english} \\
- \\
- \\
-
\end{array}
\]

Build a model (variable assignment) by applying constraints

arc(Node,Next) “iff” applying a constraint to Node gives Next
- need enough Next’s to cover all cases

Choose constraints minimizing branching (failing asap)
— e.g. from empty var assign, choose constraint 9 or 10, then 15