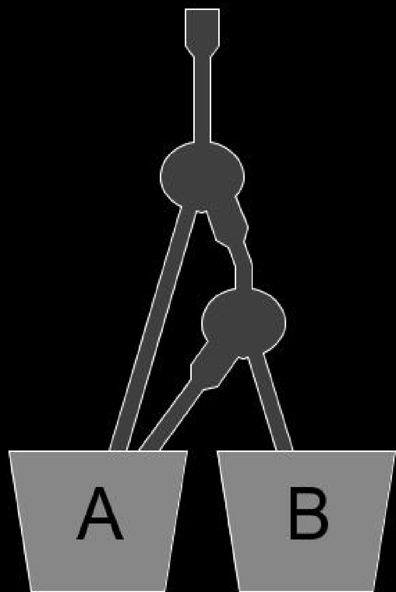
Modeling Uncertain Reasoning with Stochastic Simulation in ACT-R

Nick Cassimatis & Greg Trafton
U.S. Naval Research Laboratory

(Naïve) Uncertain Reasoning



Uncertainty in ACT-R

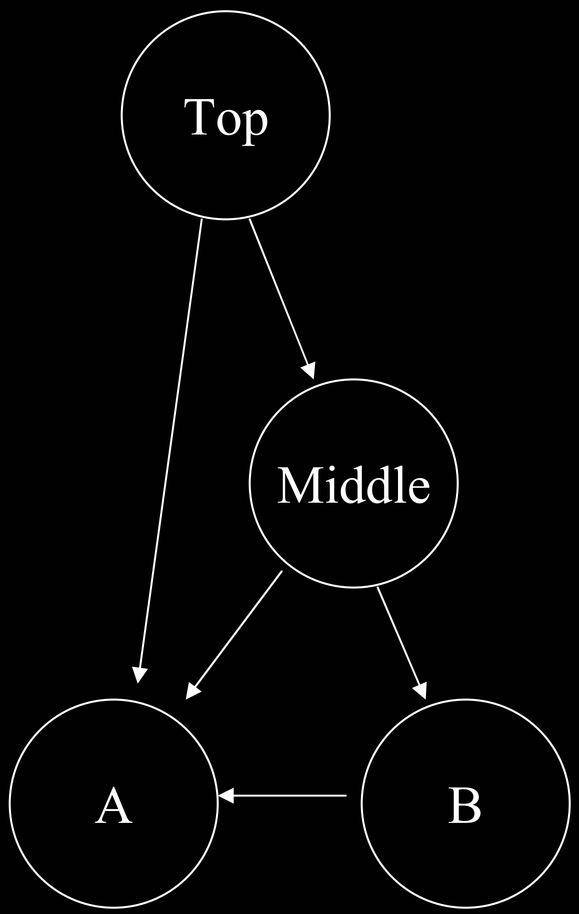
(chunk-type ball … location …) or

(chunk-type ball-location ball place)

- Hard to represent that the ball only has a probability of being at that location.
  - Chunks either exist or not.
  - No more than one slot value.
- You could add slot for probability:
  - But no way to propagate

Bayes Nets are great with probabilities …

- Bayes Nets represent conditional probabilities among state variables..
- Given initial evidence one can compute probabilities of the rest of the network.
- Even though cognitively implausible, many models based on them because of their unique abilities: object recognition, part of speech recognition, medical diagnosis …



… but not much else.

- Representationally inexpressive:
  - Propositional, no way of encoding that `Brother(John,Sam)` and `Brother(Sam,Fred)` are more similar than `Vapid(Britney)`.
  - Cannot model empirically established cognitive structures such as mental images, cognitive maps, semantic networks, etc.

Tough choice?

Radically change ACT-R or settle for BNs?

Stochastic simulation in ACT-R

- Run the ACT-R model multiple times. To get the probability of a particular outcome, see what proportion of simulations it comes up in.
  - P(Ball at A) = (# simulations where A is true) / (total number of model runs).
  - Similar to Monte Carlo simulation.
- Stochastic simulation algorithms used in computer science, but with limited propositional representations.
- Introspective cognitive plausibility: When uncertain of an event's outcome, we focus more on the most likely outcome.
  - E.g., snow vs. hail in buying a car.

Worlds keep track of simulations

- Need a memory for the outcome of each simulation.
- Worlds represent imagined states of world.
- Broad basis in Cognitive science:
  - Problem solving: States in search and planning.
  - Logical AI: Possible worlds, contexts.
  - Semantics: Situations, mental spaces.
  - Psychology of Reasoning: Mental models.

Worlds as part of ACT-R chunks

Each chunk type has a world slot.

E.g.,

```
(chunk-type
  ball-location isa ball place)
```

Becomes

```
(chunk-type
  ball-location isa ball place
  world a-world)
```

Worlds as part of ACT-R productions

Old:

```
(P production
    =goal>
     isa goal
     slot value

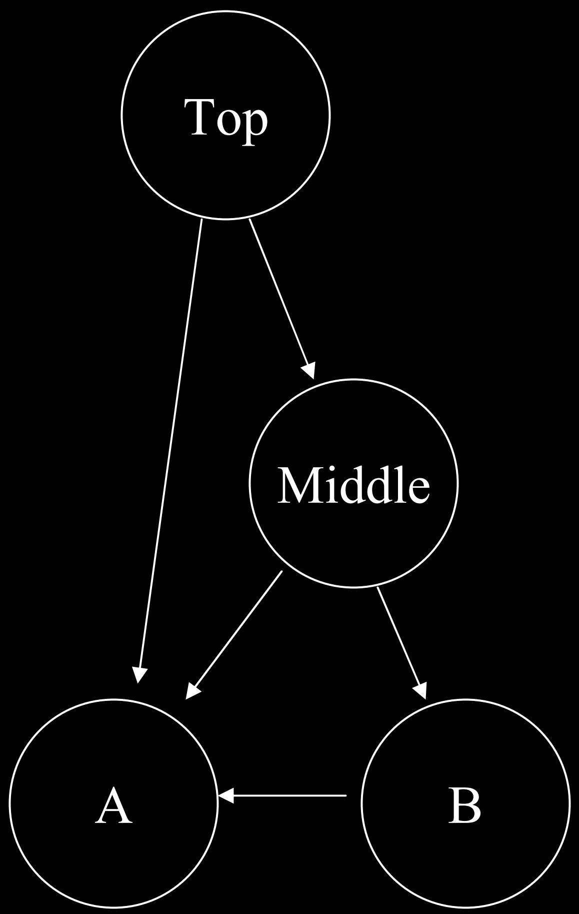
   ==>
    =goal>
     isa goal
     slot new-value)
```

New:

```
(P production
    =goal>
     isa goal
     slot value
     world =world
   ==>
    =goal>
     isa goal
     slot new-value
     world =world)
```

Converting BN nodes to ACT-R chunks

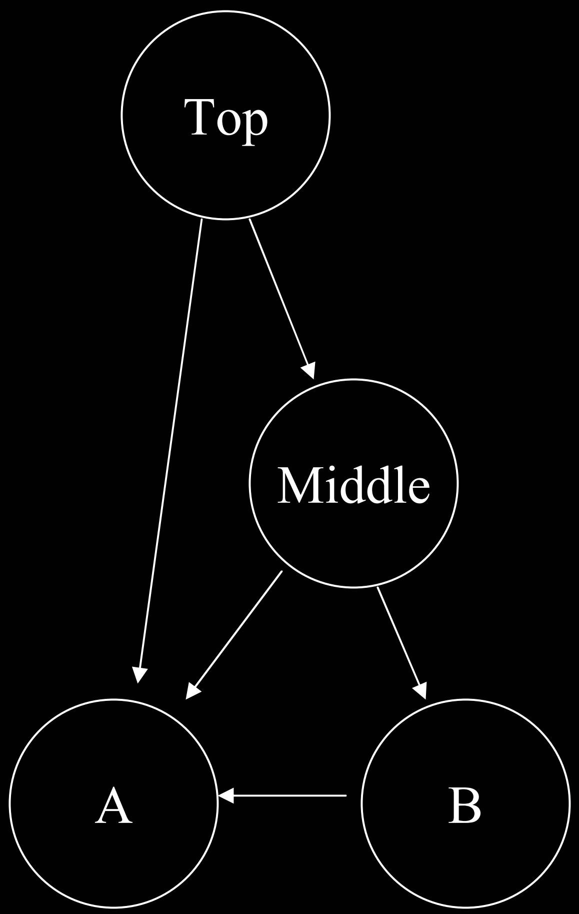
- For each node, make a chunk:
  - E.g.,
  - ```
    (MIDDLE-3
     isa node
     value nil
     name “MIDDLE”
     world w3)
    ```


    ```

Converting BN edges to ACT-R productions – I

- For each edge, make a production:
  - Preliminarily using old-style productions.

```
(P middle-t2
  =goal>
   isa set-value node =middle world
   =world
  =middle>
   isa node value nil        world
   =world
   name "MIDDLE"
  =top>
   isa node  value t name "TOP"
   world =world
==>
  =middle>   isa node value t)
```

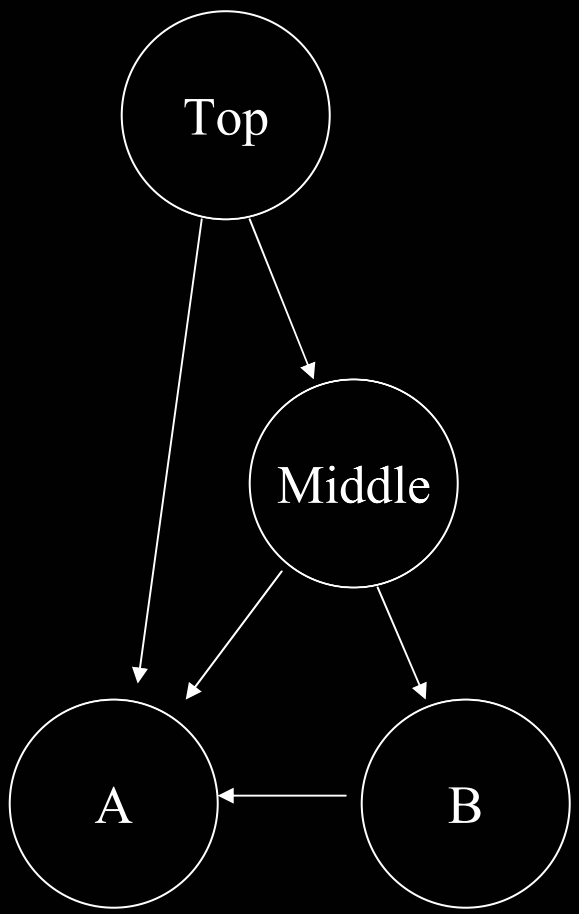


Converting BN edges to ACT-R productions – II

- The relative number of productions should match conditional probabilities

```
(P middle t1 … ==> … value t)
(P middle t2 … ==> … value t)
(P middle t3 … ==> … value false)
```

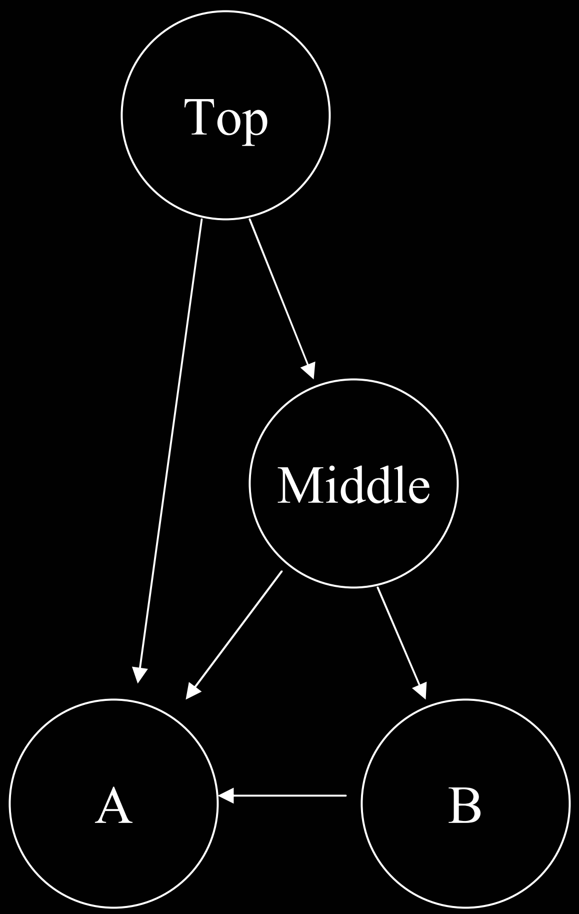
- Could also do it by setting conditional utility of production.



Running the Simulations

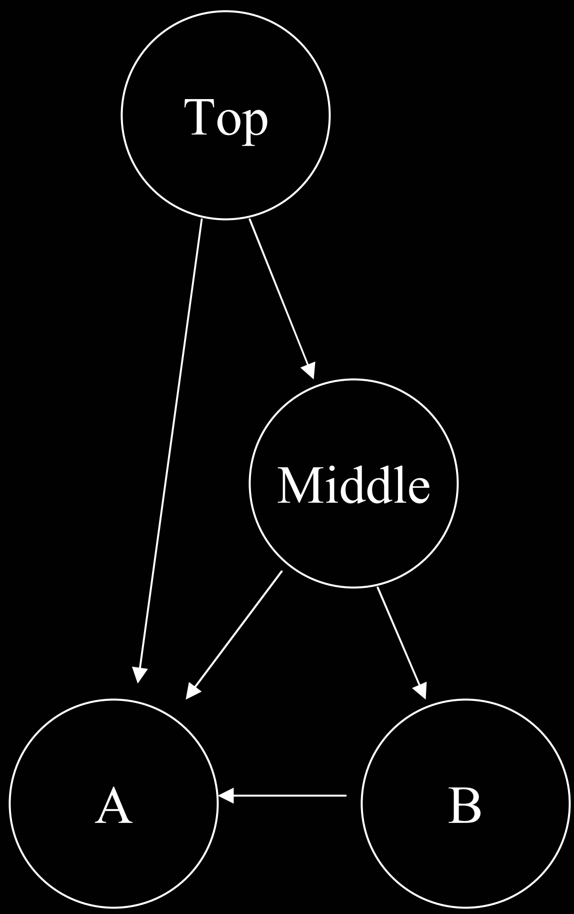
- Decide on number of worlds.
- Copy nodes of network for each world:
  - (TOP1 … world w1), (TOP2 … world w2) …
- First goal, start at edge of network.

```
(set-value-TOP
  isa  set-value
  node TOP1
  world w1)
```



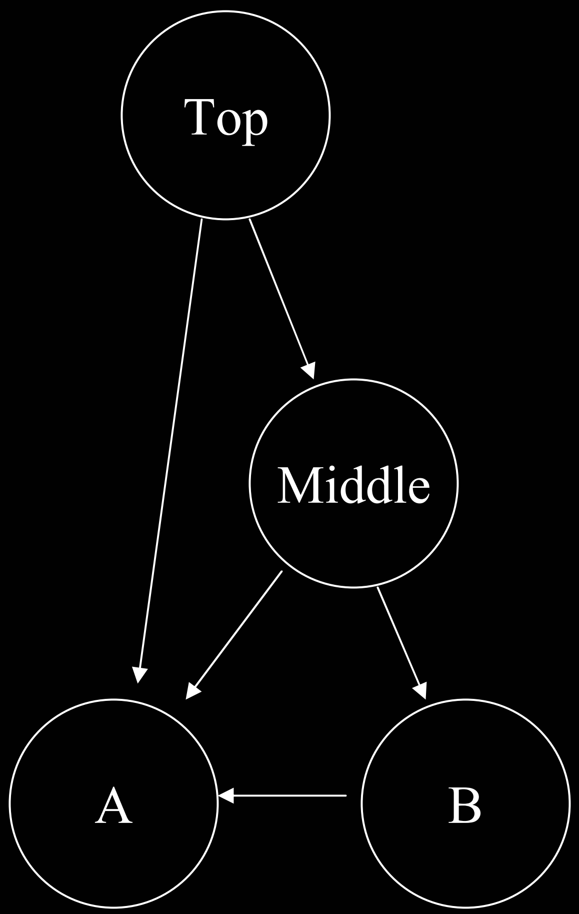
If at a set node, go to unset child.

```
(P if-set-focus-on-unset-child
 =goal>
   isa set-value     node =node
   world =world
 =node>
   isa node       - value nil
   - name "A"     world =world
 =edge>
   isa edge       parent =name
   child =child-name
 =child>
   isa node       name =child-name
   world =world   value nil
==>
 =goal>
  isa set-value     node =child)
```



Set any unset parents

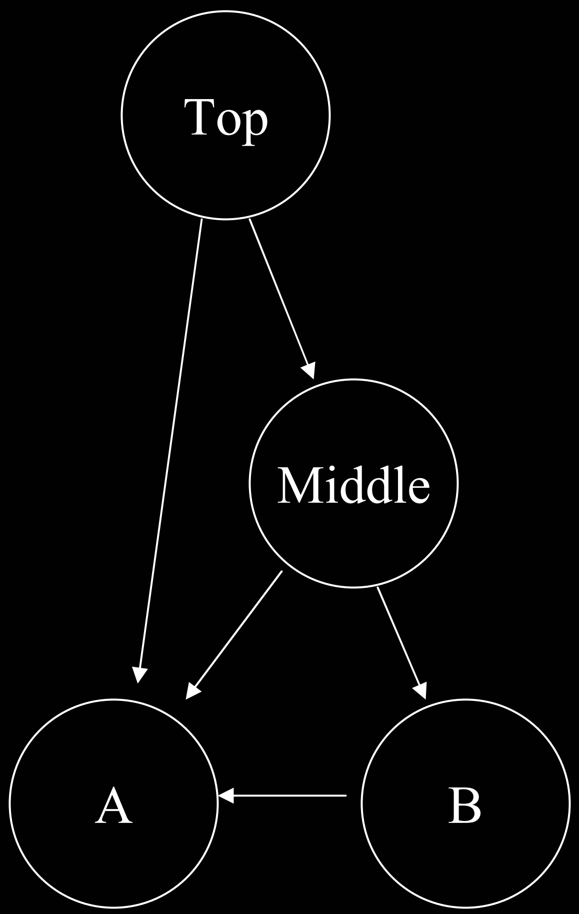
```
(P focus-on-unset-parent
 =goal>
   isa set-value      node =node
   world =world
 =node>
   isa node      name =name      value
   nil
   world =world
 =edge>
   isa edge      parent =parent-name
   child =name
 =parent>
   isa node      name =parent-name
   world =world value nil
==>
 =goal>
  isa set-value     node =parent)
```



Once parents are set, productions will fire to set it.

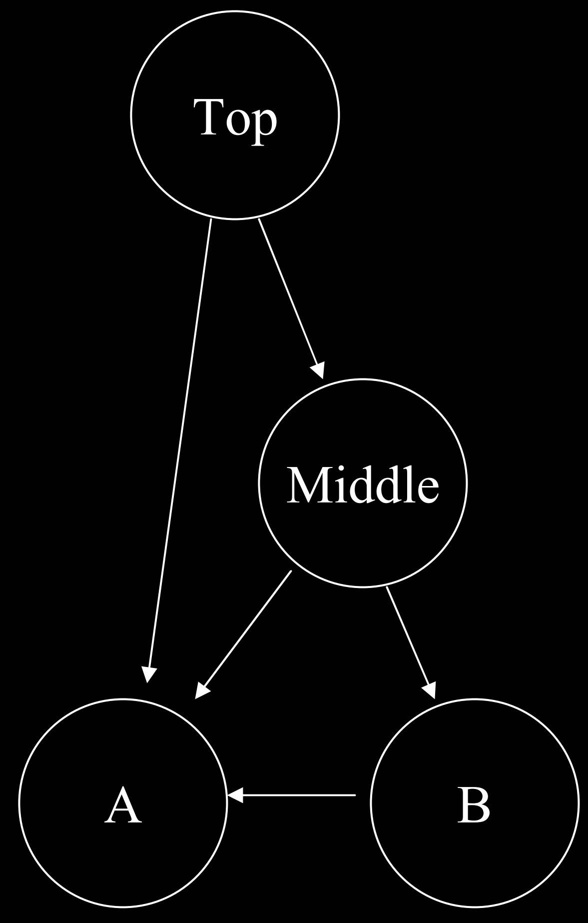
```
(P middle-t2
  =goal>
   isa set-value node =middle world
  =world
  =middle>
   isa node value nil        world
  =world
   name "MIDDLE"
  =top>
   isa node  value t name "TOP"
  world =world
==>
  =middle>   isa node value t)
```

- Conflict resolution between productions will decide which production value MIDDLE is set to.



Go to next unset child.

```
(P if-set-focus-on-unset-child
 =goal>
   isa set-value     node =node
   world =world
 =node>
   isa node       - value nil
   - name "A"     world =world
 =edge>
   isa edge       parent =name
   child =child-name
 =child>
   isa node       name =child-name
   world =world  value nil
==>
 =goal>
  isa set-value     node =child)
```



When there are no more unset nodes, simulate a new world.

```
(P simulate-next-world
  =goal>
  isa set-value     node =a     world =world

  =a>
  isa node    name "A"     - value nil     world =world

  =next-world-relation>
  isa next-world    this =world     next =next-world

  =next-world-node>
  isa node     world =next-world     - value nil

==>

  =goal>
  isa set-value   node =next-world-node   world  =next-world)
```

Computing probabilities

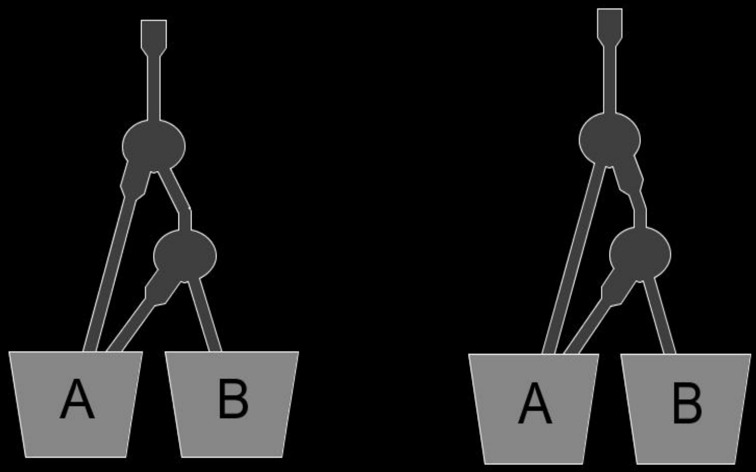
- After simulation working memory includes chunks with values set:
  - `(B1 value t world w1 … )`
  - `(B2 value false world w2 … )`
  - `(B3 value t world w3 …)`
- Manually or programmatically inspect working memory to see how many worlds a state is set as true…
  - 2/3 worlds have the ball having landed in B, so
  - *P*(*ball in B*) ~ .67

Review

- BNs map onto straightforward ACT-R chunks and productions.
- Only four productions needed to guide the simulation.
  - To set variable value, focus on parents with unset values.
  - Once variable value set, focus on unset children variables.
  - If all child values are set, focus on an unset variable.
  - Once all variables have values set, simulate another world.
- Worlds used to keep track of results of each simulation.
- ACT-R conflict resolution put the “stochastic” in “stochastic simulation”.

Evidence from ball and tube task

- If people compute probabilities, then you should have constant time over a network.
- If you are simulating, then simulations where most complicated path is more likely should take longer.
- We found evidence for stochastic simulation.



Possible future work

- ACT-R/PM for even more cognitive structures.
- Exemplar vs. prototype.
- Use BN induction algorithms to learn productions.
- Stochastic simulation is “any time” and thus lets you model the entire speed accuracy spectrum.
- Incorporate utilities.

Conclusion

- Stochastic simulation explains naïve uncertain reasoning in a way consistent with existing ACT-R models.
  - Computes probabilities without explicit probabilities.
- Simulations and (hence) worlds broadly useful: search, counterfactual reasoning, planning, prediction.