KnoMic: A Knowledge Mimic

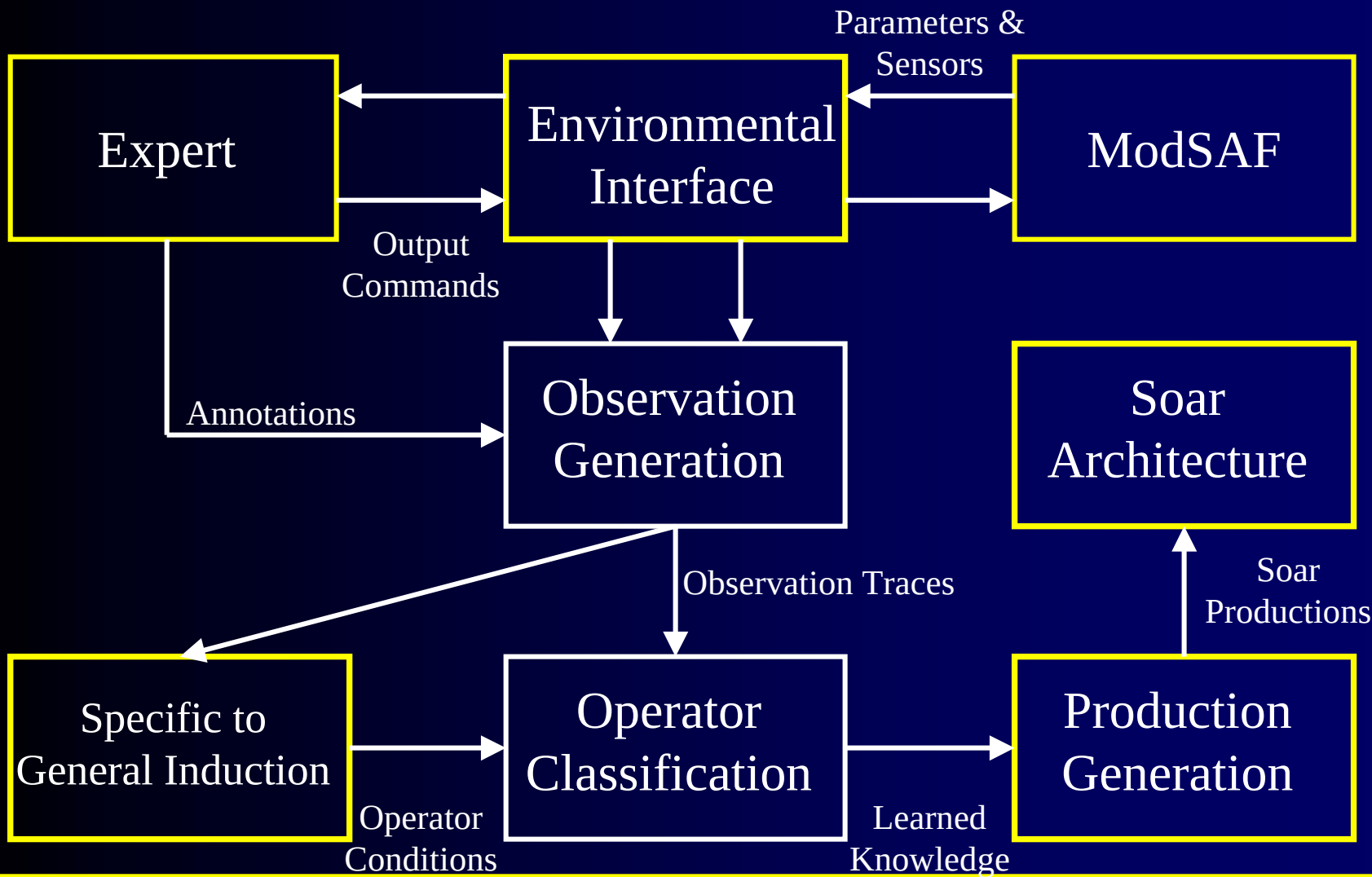
Michael van Lent

University of Michigan

KnoMic motivation

- Task performance agents are becoming common
  - Training exercises
  - Military simulations
  - Computer games
- Task performance agents require lots of knowledge
  - TacAir-Soar: 8000+ Soar productions
  - Quake II agent: 800+ Soar productions
- Knowledge acquisition for these agents is expensive
  - 15 person/years for TacAir-Soar
- Machine learning should be able to learn this knowledge automatically

KnoMic review



What's new with KnoMic

- Re-implementing in Java
- Using a more natural observation trace format
- Adding structured sensors

Re-implementing in Java

- Why rewrite?
  - I'd feel important if other people used my system
  - No one's going to use it in its current form
    - 40 pages of ugly, undocumented Tcl code
- Why Java?
  - Might as well rewrite it in a "real" programming language
  - Speed isn't a major issue
  - Leverage some of the VisualSoar Java code
  - I didn't know Java
- How's it going
  - Slowly; but not because of Java

New observation trace format

- Old way of generating observation traces
  - Add a production for each sensor and output command
  - Tcl right-hand side function to write each to a file
  - Production to write operator changes to a file

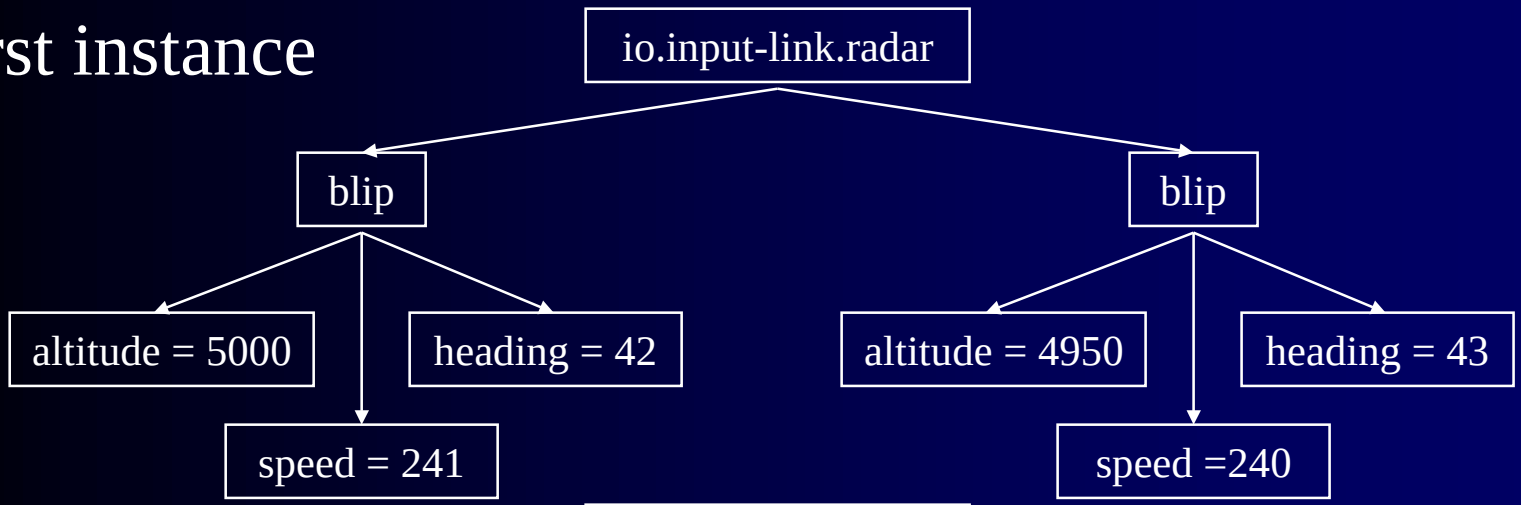
- New way of generating observation traces
  - Two Soar commands “watch –wmes on” and “log filename”
  - Callback Tcl command to indicate end of decision cycle
  - Detect operator changes in Soar trace
  - Captures internal wme changes (as well as external)
  - Tested on Towers of Hanoi, Eaters, TankSoar, and Quake II

Adding structured sensors

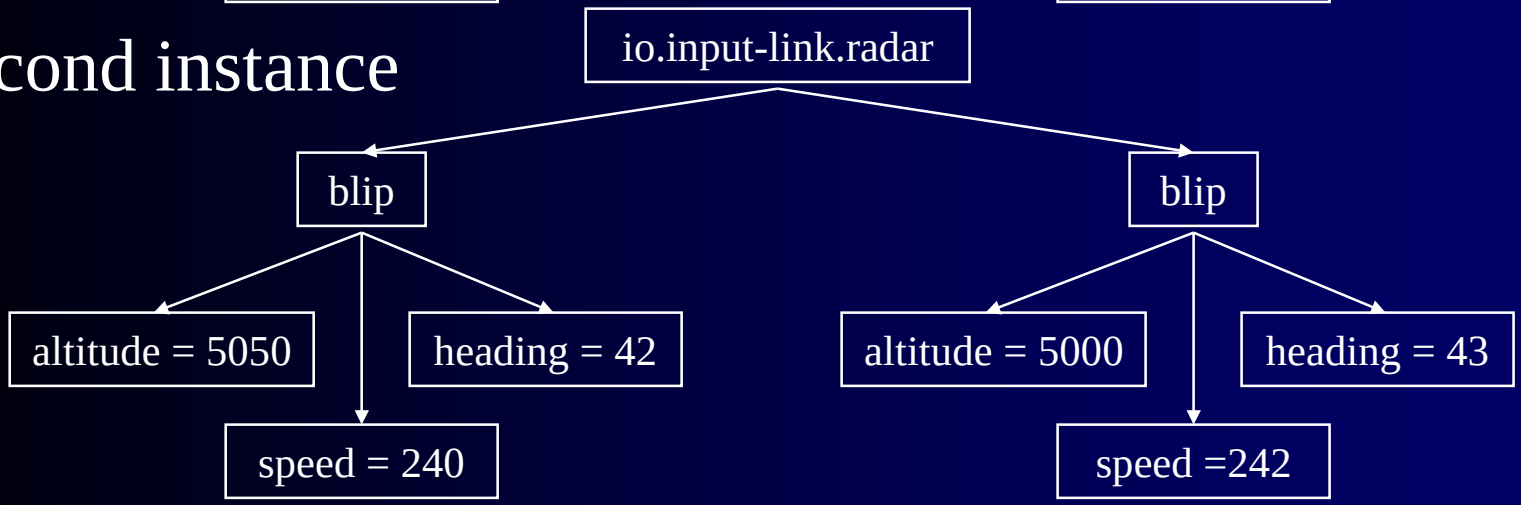
- Previously KnoMic sensors were independent
  - io.input-link.radar.blip.altitude 5000
  - io.input-link.radar.blip.speed 240
  - Doesn’t allow parallel attributes (multi-attributes)
  - Doesn’t capture all the information on the input-link
- Now KnoMic sensors can be related
  - Tree structure of sensors
    - io.input-link.radar.blip
      - altitude 5000
      - speed 240
  - Easy to do with new observation trace and VisualSoar code
  - Multi-attributes cause some problems with generalization

Generalizing with structured sensors

First instance



Second instance

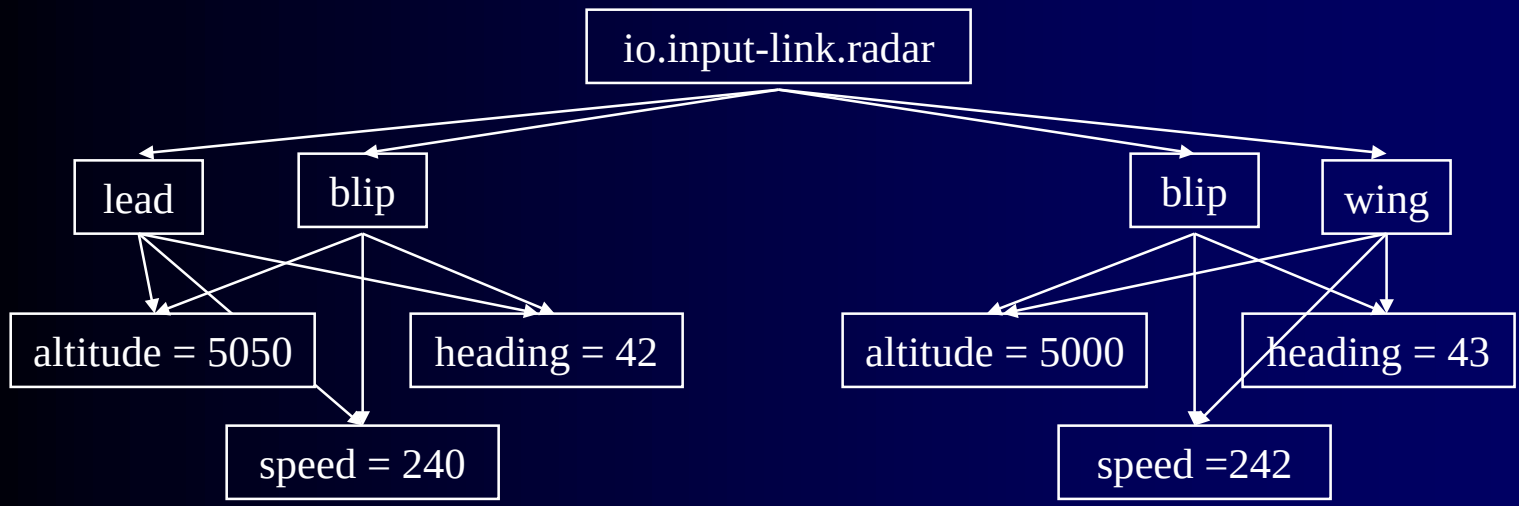


Potential Solutions

- Most specific matching
  - If identifying wmes exist use them
  - If not then match the sub-trees with the greatest number of equivalent wmes
    - If necessary apply recursively to sub-sub-trees
- Problems
  - Doesn’t guarantee a correct pairing
  - How to resolve ties
  - Requires lots of time to compute
    - Need to count matches for each pair of sub-trees
- Need to see how it works in a few domains

Potential Solutions

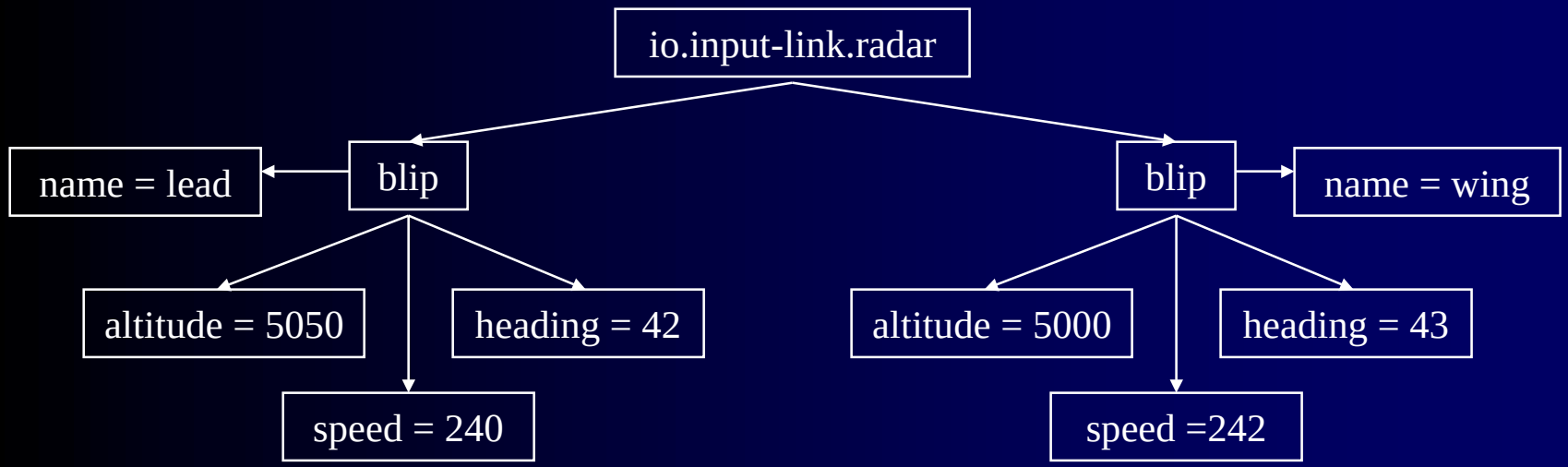
- Require unique attributes
  - Allow multi-attributes but require unique attributes also



- Doesn’t really address the problem
  - In hard cases learning will just ignore the multi-attributes

Potential Solutions

- Require unique identifiers
  - Each multi-attribute has a “name” with a unique value



- Equivalent to the previous solution
  - Easy to write a production which creates unique attribute from the name

Next steps

- Finish Java implementation
- Run some experiments with Quakebot
  - Explore different solutions to the matching problem
- Replace current learning algorithm with C4.5
  - How does C4.5 handle the matching problem?
- Rerun Quakebot experiments
- Modify Quakebot to allow observations of humans
- Work on learning from noisy observation traces

Nuggets and Coal

- Nuggets
  - Successfully defended Ph.D
  - Research is continuing
    - Interesting problems
    - Clear what the next steps should be
  - Related research efforts (Tolga, Scott) are kicking off
- Coal
  - Java rewrite is proceeding slowly
  - Reviewers want to see experiments with noisy traces
  - Future work slide from last year still mostly applicable