Report on Slate R&D
(for NIMD Quarterly Review Spring 05)

Selmer Bringsjord
Andy Shilliday, Josh Taylor,
Marc Destefano, Sunny Khemlani, Konstantine Arkoudas
Department of Cognitive Science
Department of Computer Science
Rensselaer AI & Reasoning (RAIR) Lab
Rensselaer Polytechnic Institute (RPI)
Troy NY 12180
selmer@rpi.edu
http://www.rpi.edu/~brings
for 4.5.05

Booz | Allen | Hamilton
“Sci Fi” Empowerment

Preclude bias/error in analytic reasoning
Overview

- Collaborators and Slate into the Future
  - IKRIS
    - MDL⁺ vs. description logics
  - Cycorp
  - HITIQA (TT’s ILS*; Acquaint project)
- Model-Based Hypothesis Generation: How?
- Enhancements to Slate since demos @ last PI meeting (eg)
- Report Generation ... Magic?
- IA & gaming: Prolegomenon to a Manifesto

*Tomek Strzalkowski’s Institute for Information, Logics & Security
Power

HITIQA (Q&A) info automatically installed

Cyc (ontologies) real-time seamless link to Cyc’s knowledge
"In any distribution of a radiological agent some radiological stuff is emitted."

\[(\text{implies} \ \ (\text{isa} \ \ ?\text{DIST} \ \ \text{DistributionOfARadiologicalAgent}) \ \ (\text{thereExists} \ \ ?\text{STUFF} \ \ (\text{and} \ \ (\text{isa} \ \ ?\text{STUFF} \ \ \text{RadioactiveStuff}) \ \ (\text{objectEmitted} \ \ ?\text{DIST} \ \ ?\text{STUFF}))))\]

\[(\text{implies} \ \ (\text{and} \ \ (\text{isa} \ \ ?\text{EVENT} \ \ \text{Explosion}) \ \ (\text{deviceUsed} \ \ ?\text{EVENT} \ \ ?\text{DEV}) \ \ (\text{isa} \ \ ?\text{DEV} \ \ \text{DirtyBomb}) \ \ (\text{dispersionOrigin} \ \ ?\text{DISP} \ \ ?\text{EVENT}) \ \ (\text{objectEmitted} \ \ ?\text{DISP} \ \ ?\text{EMIT}) \ \ (\text{isa} \ \ ?\text{EMIT} \ \ \text{RadioactiveStuff}) \ \ (\text{physicalDecompositions} \ \ ?\text{EMIT} \ \ ?\text{RCTV}) \ \ (\text{isa} \ \ ?\text{RCTV} \ \ ?\text{RTYPE}) \ \ (\text{isa} \ \ ?\text{RTYPE} \ \ \text{RadioActiveElementaryStuffType})) \ \ (\text{wasPhysicalDecompositionOf} \ \ ?\text{DEV} \ \ ?\text{RCTV}))))\]

"In any explosion of a dirty bomb, if there is a dispersion that originates from the explosion which emits some radioactive stuff, then the radioactive elementary stuff of which that radioactive stuff is composed was a physical decomposition of the dirty bomb."
Power

HITIQA (Q&A) info automatically installed

frame

Cyc (ontologies) real-time seamless link to Cyc’s knowledge
Power

**HITIQA (Q&A)**  
Info automatically installed

**Cyc (ontologies)**  
Real-time seamless link to Cyc's knowledge

Diagram showing a seamless link between HITIQA and Cyc.
Power

HITIQA
(Q&A)

info automatically installed

logic

real-time seamless link to Cyc's knowledge

Cyc
(ontologies)
Power

HITIQA (Q&A) info automatically installed

Cyc (ontologies) real-time seamless link to Cyc's knowledge
Initial Dialogue/Experiments
HITIQA Frames in SLATE

On 27 March 2000, the missile deployment was reported when Israeli fighter jets fired six air-to-ground missiles on the outskirts the village of Qabrikha in the Iglem Tufaah region.

event tags:
<EVTX TYPE="ATT_TYPE_1">fired</EVTX>
<EVTX TYPE="ATT_AGENT_1">Israeli fighter jets</EVTX>
<EVTX TYPE="ATT_TARGET_1">the outskirts the village of Qabrikha</EVTX>
<EVTX TYPE="ATT_INSTR_1">six air-to-ground missiles</EVTX>
<EVTX TYPE="ATT_TIME_1">27 March 2000</EVTX>
<EVTX TYPE="ATT_LOC_1">the outskirts the village of Qabrikha</EVTX>

HITIQA frame describing an event; Israeli fighter jets firing air-to-ground missiles at the outskirts of a village
The frame in Slate is the blue proposition. Green hypotheses represent queries an analyst might make which might be answered by the frame.
This inference shows that Israeli fighter jets did indeed fire at some point in time at someone. Snark confirms this for us (see ‘proof-found’, lower right).
Continuing our queries, we see that Israeli fighter jets fired on the outskirts of the village Qabrikha. Again, Snark will confirm this.
From the frame provided, we can tell that Israeli fighter jets fired on March 27, but we don’t know about March 28. Note Snark’s ‘Proof-Found’, and ‘Agenda-Empty,’ respectively.
Model-Based Hypothesis Generation and Reasoning Implemented in Slate
Multiple elements selected:
P1 - John was killed...
P2 - Conditions of killings...
P3 - Who Essid believes is not a traitor...
P4 - Who John believes is a traitor...
P5 - Who Majed believes is a traitor...
P6 - Who else Majed believes is a traitor...
P7 - No one believes everyone is a traitor...
Multiple elements selected:

P1 - John was killed...
P2 - Conditions of killings...
P3 - Who Essid believes is not a traitor...
P4 - Who John believes is a traitor...
P5 - Who Majed believes is a traitor...
P6 - Who else Majed believes is a traitor...
P7 - No one believes everyone is a traitor...

Select the propositions which you wish to model. Then, press the 'Generate Models' button to try to generate a model in which the propositions are true.
Multiple elements selected:
P1 - John was killed...
P2 - Conditions of killings...
P3 - Who Essid believes is not a traitor...
P4 - Who John believes is a traitor...
P5 - Who Majed believes is a traitor...
P6 - Who else Majed believes is a traitor...
P7 - No one believes everyone is a traitor...

Select the propositions which you wish to model. Then, press the 'Generate Models' button to try to generate a model in which the propositions are true.
Single element selected:
P1 - John was killed...

Description:
John was killed by a member of the Qaeda 'New Order' cell which consisted of John, Essid, and Majed.

Complete Information Details...

Label: H1
Name: John killed
Type: Hypothesis
Believed: 

Description:
John killed John

Data:
(Killed John John)

Finish

Workspace ...
Information ...
Model Build...
Check Cons...
Generate H...
Validate / I...
Generate R...

Generation method:
Simple Hypotheses

Propositions:
P1

Generated information
It's false that 'At least one of these is true'
John killed John

Essid killed John

It's not true that 'Essid killed John'
Majed killed John

It's not true that 'Majed killed John'
Single element selected:
H1 - John killed himself

Description:
John killed John

Generated information
It's false that 'At least one of these is true
John killed John
Essid killed John
It's not true that 'Essid killed John'
Majed killed John
It's not true that 'Majed killed John'

Select the icons from which you would like slate to generate new hypotheses. The newly generated hypotheses (if any) will appear in the box to the right and you may add them to the workspace by double clicking on one.
Single element selected:
H1 - John killed himself

Description:
John killed John
Single element selected:

H1 - John killed himself

Description:
John killed John

Premises:
P1  P2  P3  P4  P5  P6  P7

Prover: Oscar  Show Proof:  

Validate / Invalidate Sequent

Validate / Invalidate Output:
Oscar: Proof-Found: P1 P2 P3 P4 P5 P6 P7 -> H1

Select:
- Premises
- Conclusion
- Argument

Press the Argument button to select and validate an argument. Otherwise, press Premises and select premises, press Conclusion and select a conclusion, and then validate the argument.
This is an undefeated argument of strength 1.0 for:
\[ (\text{Killed John John}) \]
which is of ultimate interest.

1. \((\text{Killed John John}) v (\text{Killed Essid John}) v (\text{Killed Majed John}))\) Given
26. \((\neg(\text{Killed John John}) -> (\text{Killed Essid John}) v (\text{Killed Majed John}))\) disj-simp from \{1\}
6. \((\forall x)(\text{Believes-Traitor John} X) -> (\text{Believes-Traitor Majed} X))\) Given
4. \((\text{Believes-Traitor John Essid}) & (\text{Believes-Traitor John John}) & \neg(\text{Believes-Traitor John Majed}))\) Given
31. \((\text{Believes-Traitor John John}) & \neg(\text{Believes-Traitor John Majed})\) simp from \{4\}
16. \((\text{Believes-Traitor John x4}) -> (\text{Believes-Traitor Majed x4})\) UI from \{6\}
34. \((\text{Believes-Traitor John John})\) simp from \{31\}
37. \((\text{Believes-Traitor Majed John})\) modus-ponens2 from \{16, 34\}
7. \((\forall x)(\neg(\text{Believes-Traitor X John}) & (\text{Believes-Traitor X Essid}) & (\text{Believes-Traitor X Majed}))\) Given
28. \((\neg(\text{Believes-Traitor x22 John}) & (\text{Believes-Traitor x22 Essid}) & (\text{Believes-Traitor x22 Majed}))\) UI from \{7\}
29. \((\neg(\text{Believes-Traitor x22 John}) v (\neg(\text{Believes-Traitor x22 Essid}) & (\text{Believes-Traitor x22 Majed})))\) DM from \{28\}
38. \((\text{Believes-Traitor x22 John} -> \neg(\text{Believes-Traitor x22 Essid}) & (\text{Believes-Traitor x22 Majed}))\) disj-simp from \{29\}
39. \((\neg(\text{Believes-Traitor Majed Essid}) & (\text{Believes-Traitor Majed Majed}))\) modus-ponens1 from \{38, 37\}
41. \((\neg(\text{Believes-Traitor Majed Essid}) v (\neg(\text{Believes-Traitor Majed Majed}))\) DM from \{39\}
30. \((\text{Believes-Traitor John Essid})\) simp from \{4\}
44. \((\text{Believes-Traitor Majed Essid}) -> (\neg(\text{Believes-Traitor Majed Majed}))\) disj-simp from \{41\}
33. \((\text{Believes-Traitor Majed Essid})\) modus-ponens2 from \{16, 30\}
45. \((\neg(\text{Believes-Traitor Majed Majed})\) modus-ponens1 from \{44, 33\}
5. \((\forall x)(\neg(\text{Lower-Rank X John}) -> (\text{Believes-Traitor Majed X})\) Given
18. \((\text{Lower-Rank x8 John} -> (\text{Believes-Traitor Majed x8}))\) UI from \{5\}
46. \((\text{Lower-Rank Majed John})\) modus-tollens2 from \{18, 45\}
86. \((\text{Believes-Traitor Majed John}) -> (\text{Lower-Rank Majed John})\) Conditionalization from \{46\}
87. \((\neg(\text{Believes-Traitor Majed John}) v (\text{Lower-Rank Majed John}))\) disj-cond from \{86\}
88. \((\text{Believes-Traitor Majed John}) & \neg(\text{Lower-Rank Majed John})\) i-DM from \{87\}

How does this work?

- Slate calls Paradox, which returns a model in its own format...
- Slate then translates Paradox’s output into MDF (our Model Description Format)...
- Then, the model represented in MDF is translated into GraphViz’s ‘dot’ format...
- From which GraphViz draws an image presenting the model, which Slate receives and displays!
What is MDF?

MDF (Model Description Format) is a general format, invented by the RAIR Lab’s J&A, for describing models. Model finders currently share no common formats, which means that there is no easy way to examine models generated by different model finders.

With simple translation utilities, MDF will allow models to be visually represented regardless of the methods used for finding the models.
Interface Improvements

Deductive Hypothesis Generation Refined;
Ditto for Consistency Checking
Multiple elements selected:
P1 - if Majed is sleeper, then John is Attan
P2 - Majed sleeper

Generation method:
- Simple Hypotheses
- MMOI Abduction
- MMOI Deduction

Generate Hypotheses
Select the icons from which you would like slate to generate new hypotheses. The newly generated hypotheses (if any) will appear in the box to the right and you may add them to the workspace by double clicking on one.
Multiple elements selected:

P1 - if Majed is sleeper, then John is Attan
P2 - Majed sleeper

[Image of a computer interface with options for generating hypotheses]
Multiple elements

P1 - if Majed is Attan
P2 - Majed is Attan

Complete Information Details...

Label: H1  Name: John is Attan  Type: Hypothesis

Description:
John is Attan

Data:
(= John Attan)

Finish

Worksp... Informa... Model B... Check C... Generat... Validate... Generate Hypotheses

Generation method:
Formal Deduction

Propositions:
P2
P1

Select the icons from which you would like SLATE to generate new hypotheses. The newly generated hypotheses (if any) will appear in the box to the right and you may add them to the workspace by double clicking on one.

Generated information
John is Attan
Single element selected:
H1 - John is Attan

Description:
John is Attan

Validated / Invalidate Sequent

Premises:
P2
P1

Conclusion:
H1

Prover: Oscar
Show Proof: ✓
This is an undefeated argument of strength 1.0 for:
  
  \[ (= \text{John Attan}) \]
  
  which is of ultimate interest.

1. \[ (\text{Sleeper Majed}) \rightarrow (= \text{John Attan}) \]  Given
2. \[ (\text{Sleeper Majed}) \]  Given
8. \[ (= \text{John Attan}) \]  modus-ponens1 from \( \{2,1\} \)
Report Generation: Magic?
Prior R&D

PROVERB

But...

Dormant?

Taps into “unprincipled” NLG

Reasoning that is input lacks power of Athena

No natural language corresponding to diagrammatic knowledge

Can’t handle resolution-based reasoning

Can’t handle methods, only proofs (not dynamic proofs)
I have now concluded my analysis. The answer is that John committed suicide. I trust you will find this new fact useful in our agency’s ongoing attempt to defeat Al-Qaeda. My argument for this conclusion runs as follows.

Overall, if we know that either John killed himself, Majed killed John, or Essed killed John, and we know that neither Majed nor Essed killed John, we can infer that John committed suicide. This means that we have three sub-goals to aim for. The first one is that the culprit was either John, Majed, or Essed. But this goal is easy to substantiate: It can be derived from the given fact that someone in New Order performed the killing, combined with the fact that there were only three members of New Order at the time of the execution. We turn our attention now to the other two sub-goals. If we can explain how to reach them, the case is closed.

...
AAAI: Getting More Serious About Gaming

GGP Competition:

The AAAI General Game Playing Competition is designed to test the abilities of general game playing systems by comparing their performance on a variety of games. The competition will consist of two phases: a qualification round and a run-off competition.

Qualification Round

In the qualification round, entrants will play several different types of games, including single player games (Puzzle games), competitive games (such as Tic-tac-toe or some variant of Chess), games with both competitors and cooperators. In some cases, the game will be exhaustively searchable (as in Tic-tac-toe); in other cases, this will not be possible (as in Chess). Players will have to handle these possibilities. For this year’s competition, in all cases, complete information of the game will be available (as in Chess or Tic-tac-toe); in future competitions, only partial info will be available (as in Battleship). Entrants will be evaluated on the basis of consistent legal play, ability to win in winning positions, and overall time; and the best will advance to the second round.

Runoff Round

In the runoff round, the best of the qualifiers will be pitted against each other in a series of games of increasingly complexity. The entrant to win the most games in this round will be the winner of the overall competition.

Rules of the Games

Note that, prior to the competition, players will be told nothing about the games to be played. The rules of all games will be transmitted to the players electronically at the beginning of each game, in the GGP game description language. Game Playing Systems must be able to read the rules for each game, receive runtime information from the game manager, and inform the manager of its moves.

Prize and Eligibility

A $10,000 Prize will be awarded to the winning entrant. The competition is open to all computer systems, except those generated by affiliates of Stanford University. Sorry, no human players allowed.

Read the official announcement flyer.

email: info@games.stanford.edu
Aim: Formally Specified IA Game;
or: IA as a Formal Game
Partial Formalization of IA Game

role(agent)

init(prop(P50,P31,t))
init(prop(P50,P58,t))
init(prop(P50,P60,t))
init(prop(P50,P61,t))
init(prop(P50,P66,t))
init(prop(P50,P69,t))
init(prop(P31,P48,t))
init(prop(P31,P57,t))
init(prop(P31,P58,t))
init(prop(P31,P59,t))
init(prop(P31,P67,t))
init(prop(P11,P78,f))

...init(control(agent))

next(prop(M,N,x)) ⇐
  does(agent,mark(J,K))
  true(prop(M,N,x)) &
  (distinct(M,J) | distinct(N,K))

legal(agent,mark(X,Y)) ⇐
  true(prop(X,Y,t)) &
  true(control(agent))

conclusion ← true(prop(P11,P78,t))
terminal ← conclusion()
THE END