Dynamic Rule Systems: fighting against Brittleness

Stefania Costantini

University of L'Aquila, Italy

RuleML, Skype Telecon

Rule-Based AI Systems:
often brittle and inflexible?

Perlis, Anderson, SOAR team, and others...

- ▶ Brittleness: propensity of an agent to perform poorly or fail outright in the face of unanticipated changes.
- ▶ The complexities of real-world environments are difficult to account for in advance.
- ▶ Organization and integration of multiple, varied cognitive components a challenging task.

Dynamic Rule Systems
ACE: Agent Computational Environment

$$\mathcal{A} = \langle A, R_1, \ldots, R_q, C_1, \ldots, C_s, \rangle$$

- ▶ module A is the "basic agent", i.e., a rule-based program written in any computational logic language;
- ▶ R_is are "Reasoning modules", specialized in specific reasoning tasks (among which, e.g., quantitative reasoning, complex event processing, etc.).
- ▶ C_js are heterogeneous knowledge sources, either internal or external.
- ▶ Agents in ACEs can interact not only with other agents via message-passing, but also via *bridge rules* with the R_is and C_js.
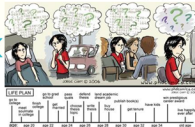
Avoiding Brittleness: Knowledge Integration

Forms of Knowledge Integration are often required for taking critical decisions or drawing relevant conclusions. Knowledge is in general, from sources that can be heterogeneous, and can be distributed on the web.

Case Study (EMAS@AAMAS2016)

I NEED a Personal Assistant!!!

Prospective College Student:

Case Study (Cont'd)
Student's Requirements and steps to be done

List of preferred universities, eligible if **ranking is higher than a threshold**.

Preference on universities with the **best ranking** and, then, with the **best basketball team**.

Procedure, all steps are subject to the payment of fees:

- ▶ pass the general SAT test, and the specific SAT test for the subject of interest;
- ▶ in case of foreign students, pass the TOEFL test;
- ▶ fill the general application on the application website;
- ▶ send the SAT results to the universities of interest, and complete the application for such universities.

Case Study

The procedure requires information to be gathers, steps to be suitably organized, reasoning to be performed.

Two aspect of *quantitative reasoning*:

- ▶ the cost of knowledge, as in practical terms a student applies in order *to know* whether she is admitted;
- ▶ reasoning under budget limits: an application can be sent only if one can pay the fees related to the application, and can then afford tuition fees.

Planning: choose the universities to which applications can be sent, according to preferences and available budget for application and tuition.

Knowledge Integration: Bridge Rules

ACEs extend MCS (Multi-Context Systems, Brewka, Eiter & a.). A **bridge rule** occurring in module M of an MCS or ACE has a form and a functioning reminiscent of datalog with negation (or ASP):

$$s \leftarrow (\mathcal{C}_1 : p_1), \ldots, (\mathcal{C}_j : p_j),$$
$$not\,(\mathcal{C}_{j+1} : p_{j+1}), \ldots, not\,(\mathcal{C}_m : p_m)$$

Bridge-Rules: Enhancements

- We introduced **bridge rule patterns** where each C_i can be a term of the form $m_i(k_i)$ that we call *context designator*, which indicates the *kind* of module to be queried;

- Bridge rule patterns are dynamically instantiated to actual bridge rules prior to activation;

- In the original formulation (ground) bridge rules are applied whenever applicable; in our approach, bridge rules are grounded dynamically w.r.t. the system's current state, and their activation is proactive, according to each context's policy.

Proactive (Bridge-Rule) Treatment

Instantiation and Activation

- Special action *instantiate* to substitute context designators with actual modules' names.
- a (bridge) rule with head $A(\hat{x})$ can be instantiated and "fired" via a *trigger rules* of the form

  $$Q \textbf{ enables } A(\hat{x})$$

  whenever query Q to the agent's knowledge-base evaluates to *true*;

- the result returned by a (bridge) rule with head $A(\hat{x})$ is exploited via a *bridge-update rule* of the form

  $$\textbf{upon } A(\hat{x}) \textbf{ then } \beta(\hat{x})$$

  where $\beta(\hat{x})$ specifies the elaboration to be applied for $\hat{x}$ to be incorporated into the knowledge base (management function).

Semantics (informal) and Enhancements

- in MCSs and ACEs
  Module = Logic + Semantics + Knowledge Base
- For MCS or ACE *M*, a **belief state** $S = (S_1, \ldots, S_h)$ includes for each component a set of semantically acceptable possible consequences of its knowledge base.
- Desirable data states (called **Equilibria**) are those which encompass bridge-rules application (each applicable bridge rule is actually applied).
- In our approach we introduced general **Update Operators** (one for each context), **Timed Belief States** and **Timed Equilibria**.

In the Case Study

- ▶ A student('s ACE) can obtain via a bridge rule the list of Universities which are suitable according to subjects, ranking, team ranking, etc.;
- ▶ The above information can be obtained proactively (upon need) and kept up-to-date;
- ▶ The student can then choose among the Universities (s)he can afford;
- ▶ A suitable bridge-rule pattern can be instantiated for issuing applications;
- ▶ A suitable bridge-rule pattern can be instantiated for finalizing enrollment.

But, the student must still undergo the tests in person!

Conclusions and Future Directions

Observation: RASP, practical affordable ASP-based tool for quantitative reasoning.

Evolution

- Enhanced ACEs, with partially specified bridge rules to be dynamically customized and activated.
- Future directions:
    - K-ACE, very general multi-level architecture with a "fractal" structure, i.e., components can be not only agents and contexts, but also other K-ACEs.
    - Potential Applications: eHealth resource management, Robot operation under limited resources, etc.
    - "Dynamic Proactive Expert Systems" (DyPES), with K-ACE as their "core".

Experimental Results

Now, I REALLY need an Alter Ego...
to do the hard work in my place!

Finally Enrolled! But...