Multi-Agent-Contexts Systems for Reasoning and Acting in Heterogeneous Environments

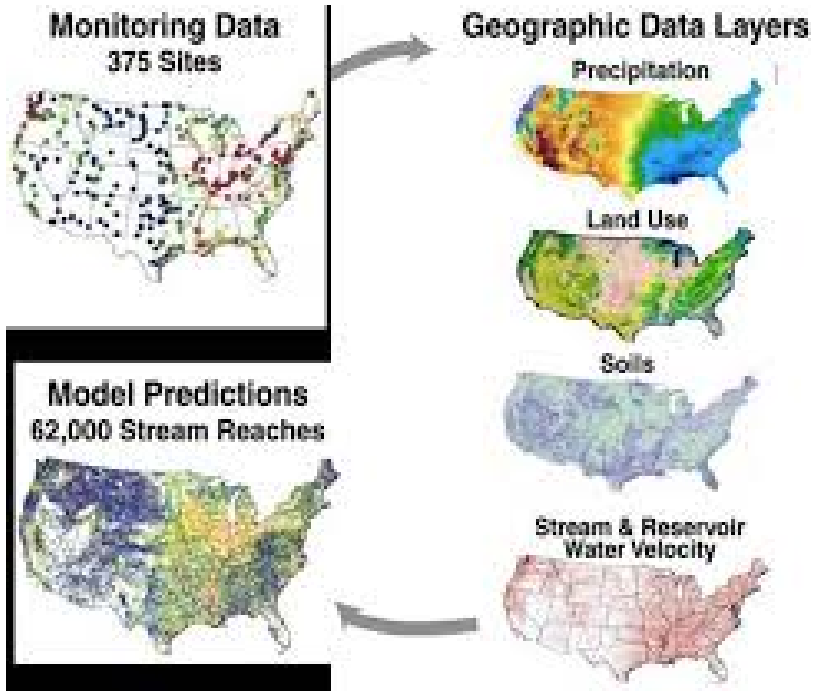
Stefania Costantini

University of L'Aquila, Italy

LPNMR 2015 Lexington, Kentuky

Data Representation and Knowledge Exchange

- The importance of data/knowledge exchange in Artificial Intelligence (AI) applications is constantly increasing.
- In many application domains, it is necessary to comprise and elaborate information provided by multiple sources.



Data Representation and Knowledge Exchange in Logical Agents

Multi-Agent Systems (MAS) are particularly suited for AI distributed applications. Our interest: logical agents.

- Logic-based data management and exchange are therefore important aspects of knowledge representation and reasoning in logical agents.
- Non-monotonic reasoning crucial for:
  - definition of patterns for knowledge exchange;
  - definition of modalities for knowledge integration and exploitation.

DACMAS: Data-Aware Commitment-based Multi-Agent System (MAS)

A recent interesting data-centric agent architecture, knowledge representation via ontologies.

- *Institutional* agent which owns a "global" (DLR-Lite) TBox, specifying the domain in which the MAS operates.
- Each participating agent is equipped with its local ABox (consistent with the TBox, mutual consistency not required).
- Commitment-based Communication.
- Patterns for simple reactive and proactive rules.

Our view of DACMAS

DACMAS: yet another agent architecture or useful Meta-Model for Multi-Agent Systems?

- Particularly interesting for a general specification of knowledge management and communicative features.
- Allows for affordable verification.
- Very general about an agent program's definition, so it can be specialized to many existing agent-oriented logic languages.
- Missing: access to external knowledge sources.

Accessing External Sources

- Agents & Artifacts approach: postulates that a (homogeneous) description is available for such sources, that can in general be manipulated by agents.
- (managed) Multi-Context Systems (mMCSs): drop the assumption of making external sources in some sense homogeneous: rather, the approach deals explicitly with their different representation languages and semantics.
- mMCSs: allow for datalog-like non-monotonic queries to external sources, called “bridge rules”.

Bridge Rules in mMCSs

Bridge Rule Syntax

$$o(s) \leftarrow (c_1 : p_1), \ldots, (c_j : p_j), \\ not\ (c_{j+1} : p_{j+1}), \ldots, not\ (c_m : p_m).$$

where the c_is are *contexts*, i.e., external knowledge sources, and o is a knowledge integration operator dealt with by a *management function*.

Management function: responsible for the integration of the new knowledge in the destination context.

mMCSs Definitions

- Multi-context System $M = (C_1, \ldots, C_n)$, each C_i consisting of an underlying logic L , a knowledge base, a set of bridge rules and a management function, where $L = (KB_L; Cn_L; ACC_L)$.
- Data state $S = (S_1, \ldots, S_n)$ for M, where S_i is a data state for C_i, i.e., an acceptable set of consequence of its knowledge base according to the logic.

mMCSs Semantics

Equilibrium

S is an equilibrium for *M* iff each S_i is "stable" w.r.t. the application of bridge rules, given the associated management function.

A management function for a context is called *local consistency (lc-) preserving* iff, after bridge rule application, the resulting data state is consistent.

DACMACS

Data-Aware Commitment-based managed Multi-Agent-Context Systems)

- Agents can query (sets of) contexts, but contexts cannot query agents.
- Agents are equipped with bridge rules, whose application is however activated via special *trigger rules*, which allow a bridge rule to be invoked upon certain conditions and/or according to a certain timing.
- The result of a bridge rule is interpreted as an agent-generated internal event, and captured by reactive rules which may determine modifications to the agent's ABox.

Bridge Rules in DACMACS

Bridge Rule Syntax

$A(\hat{x})$ **determinedby** $E_1, \ldots, E_k, not\, G_{k+1}, \ldots, not\, G_r$

where each of the E_is and the G_is are datalog queries to external context, whose reference is either locally known or provided by the Institutional agent via a query Role@inst($role$)

Bridge Rules in DACMACS

Trigger Rule: proactively enables a bridge rule

$$Q(\hat{x}) \textbf{ enables } A(\hat{y}) \ [Time \mid Frequency]$$

Bridge-Update Rule: incorporates new knowledge

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

$\beta(\hat{x})$ specifies the operator, management function and actions to be applied to $\hat{x}$.

Example: Personalized Healthcare

Paula, 78, lively and beautiful lady

Example: Medical Assistant Agent

The patient: basic conditions

$person(paula).\quad age(paula, 78).$
$special\_condition(X, elderly) \text{ :- } person(X), age(X, A), A > 70.$
$has\_disease(paula, cardiac\_insufficiency).$
$takes\_medicament(P, anticoagulant) \text{ :-}$
$\qquad has\_disease(P, cardiac\_insufficiency).$

Example: Medical Assistant Agent (cont'd)

Symptom: extreme weakness

Possible cause: low hemoglobin level (value less than 10)

$$
\begin{aligned}
&\mathit{has\_symptom}(\mathit{paula}, \mathit{extreme\_weakness}).\\
&\mathit{symptom\_cause}(\mathit{extreme\_weakness}, \mathit{hemoglobin\_level}).\\
&\mathit{blood\_test}(\mathit{hemoglobin\_level}).\\
&\mathit{anomalous\_value}(P, \mathit{hemoglobin\_level}) :-\\
&\qquad \mathit{person}(P), \mathit{value}(P, \mathit{hemoglobin}, V), V \leq 10.
\end{aligned}
$$

Example: Medical Assistant Agent (cont'd)

Whenever it is the case, a query can be enabled to the knowledge base which records blood test results, so as to obtain most recent values.

Trigger Rule

$$person(P)\ AND\ has\_symptom(P,S)\ AND$$
$$symptom\_cause(S,T)\ AND\ blood\_test(T)\ AND$$
$$Spec@inst(Rec, blood\_tests\_records)\ \textbf{enables}$$
$$blood\_test\_result(Paula, T, V, Rec).$$

Example: Medical Assistant Agent (cont'd)

Bridge Rule and Bridge-Update Rule

$$blood\_test\_result(P, T, V, Rec) \textbf{ determinedby } value(P, T, V) : Rec.$$

$$\textbf{upon } value(P, T, V) \textbf{ then } add(value(P, T, V)).$$

Example: Medical Assistant Agent (cont'd)

Hemoglobin value too low: enable query an external diagnostic knowledge base about possible causes.

Trigger Rule

$$person(P)\ AND\ has\_disease(P,D)\ AND$$
$$special\_condition(P,S)\ AND\ anomalous\_value(P,V)\ AND$$
$$Spec@inst(Diag, medical\_diagnosis)\ \textbf{enables}$$
$$poss\_causes(P,D,S,V,C,Diag).$$

Example: Medical Assistant Agent (con'd)

Bridge Rule and Bridge-Update Rule

$$poss\_causes(P, D, S, V, C, Diag) \textbf{ determinedby } poss\_causes(P, D, S, V, C) : Diag.$$

$$\textbf{upon } poss\_causes(P, D, S, V, C, Diag) \textbf{ then } infer\_plausible(P, D, S, V, C, R).$$

The agent is thus able to reason about the returned diagnostic hypotheses, given the local data available about patient's history
Result: hemorrhages as side-effect of a medicaments

Situation now hopefully under control

Semantics

- DACMAS: operational, transition system.
- mMCSs: equilibria, where bridge rules are deemed to be applied whenever applicable.
- DACMACS: equilibria, extended w.r.t. mMCSs, because bridge rule applicability and incorporation of results must be proactively enabled.

Semantics (cont'd)

Explicit updates

$\Pi = \Pi_1, \Pi_2, \ldots$ is the sequence of finite updates performed to agents' and contexts' private knowledge bases at time instants $t_1, t_2, \ldots$

Update Operator

$\mathcal{U}$ is the tuple composed of the operators that each agent or context employs for incorporating the new knowledge.

Semantics (cont'd)

Timed Data State

is a tuple $S^T = (S_1^T, \ldots, S_n^T)$ such that each S_i^T is an element of Cn_i at time T. The timed data state S^0 is an equilibrium according to a definition analogous to mMCSs. Then, data states evolve according to updates and to bridge and bridge-update rule application.

Bridge rule applicability changes, depending upon trigger rules.

Semantics (cont'd)

DACMACS Equilibria

A timed data state $S^T = (S_1^T, \ldots, S_n^T)$ is an equilibrium iff it is stable w.r.t. bridge and bridge-update rule application, performed after considering the current update.

Properties

- *Safe evolution trajectory* of a DACMACS: sequence of equilibria w.r.t. agents' and contexts' knowledge base updates.
- *Local Consistency*: like mMCS, a DACMACS enjoys Local Consistency iff all management functions are *local consistency (lc-) preserving*.
- *Global Consistency*: not required.

Conclusions

- DACMACS = DACMAS + mMCS =
  Data-centric agents + knowledge exchange with external knowledge bases
- Issue: a-priori and run-time verification
- Extension: Agents and Contexts exchange Ontological Definitions
- Extension: use acquired ontological definitions for positive and negative explanations, remember the sources of new definitions, update level of trust and reputation.

The End!

Thank You for your Attention:-) Questions?