MCMAS: A model checker for multi-agent systems

Alessio R. Lomuscio, Hongyang Qu, Franco Raimondi

Verification of Autonomous Systems Research Group
Department of Computing
Imperial College London

HIGHLIGHTS, 21 September 2013

Multiagent systems

Computer systems that

- sense the environment,
- are capable of autonomous action,
- display a degree of social interaction with peers or humans by negotiating, coordinating, cooperating, ...

in order to meet their design objectives.

Key properties: "intelligent behaviour", adaptability, fault-tolerance, self-diagnosability, etc.

MAS specifications

- Emphasis on *intentional* properties of the processes.
- Concepts beyond temporal logics are commonly used: *knowledge, beliefs, desires, intentions, goals, commitments*, each with their own modal formalisation.

This takes inspiration from earlier AI work by Dennett and McCarthy.

Some generic MAS specifications

- "If an agent *knows* that one of its current *goals* is no longer achievable, it will drop it at the next tick and begin replanning.".
- "Whenever a *fault* occurs, all agents in the system *can coordinate* to ensure that within x ticks they will *know* what fault it is and *can cooperate* to rectify it within y ticks".
- "All *intentions* selected by the agent's planning mechanism are acted upon within x ticks".

Wide variety of modal logics including epistemic, strategic, etc: *all strictly more expressive than plain temporal logic*.

Epistemic specifications: the logic CTLK

Specifications from a fault diagnosis mechanism in UAV.

- $AG(fault \rightarrow AFK_i fault)$
- $AG(fault \rightarrow AFD_\Gamma fault)$
- $AG(D_\Gamma fault \rightarrow AFC_\Gamma fault)$

Gives rise to *natural and intuitive* specifications pertaining to states of information of agents.

$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid K_i\phi \mid E_\Gamma\phi \mid D_\Gamma\phi \mid C_\Gamma\phi \mid AG\phi \mid AX\phi \mid A(\phi U \phi)$

Interpreted systems [PR85,FHMV95]

- A MAS is composed of a set of agents $A = \{1, ..., n\}$ and an environment e.
- Each agent is described by
  - A set of *local states* L_i,
  - A set of *local actions* Act_i,
  - A *local protocol* function $P : L_i \rightarrow 2^{Act_i}$.
  - An *evolution* function $\tau_i : L_i \times Act_1 \times \ldots \times Act_n \times Act_e \rightarrow L_i$.
- Evolution by synchronous composition of τ_i.

Models for CTLK

A model $M = (S, I, T, \sim_1, \ldots, \sim_n, h)$ is a tuple such that:

- $S \subseteq L_1 \times \ldots \times L_n \times L_e$ is the set of global states for the system,
- $I \subseteq S$ is a set of initial states for the system,
- T is the temporal relation for the system defined by $s \; T \; s'$ if there exist actions $a_1, \ldots, a_n, a_e$ such that $a_i \in P_i(l_i(s))$ and $\tau_i(l_i(s), a_1, \ldots, a_n, a_e) = l_i(s')$ for all $i \in A$ and e.
- $\sim_i, i \in A$, is an epistemic relation defined by $(l_1, \ldots, l_n, l_e) \sim_i (l'_1, \ldots, l'_n, l'_e)$ if $l_i = l'_i$.
- $h : \mathcal{P} \to 2^S$ is an interpretation for the set of propositional atoms $\mathcal{P}$.

Satisfaction

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \psi \mid EX\phi \mid E\phi U\phi \mid EG\phi \mid K_i\phi \mid E_\Gamma\phi \mid D_\Gamma\phi \mid C_\Gamma\phi$$

Satisfaction

- CTL as usual
- $(M, s) \models K_i\phi$ iff $\forall s' \in S$ if $l_i(s) = l_i(s')$ then $(M, s') \models \phi$
- $(M, s) \models E_\Gamma\phi$ iff $\forall s' \in S$ if $s\ R_\Gamma^E\ s'$ then $(M, s') \models \phi$
- $(M, s) \models D_\Gamma\phi$ iff $\forall s' \in S$ if $s\ R_\Gamma^D\ s'$ then $(M, s') \models \phi$
- $(M, s) \models C_\Gamma\phi$ iff $\forall s' \in S$ if $s\ R_\Gamma^C\ s'$ then $(M, s') \models \phi$

$$R_\Gamma^E = \bigcup_{i\in\Gamma} \sim_i,\ R_\Gamma^D = \bigcap_{i\in\Gamma} \sim_i,\ R_\Gamma^C = (R_\Gamma^E)^*$$

The bit transmission problem [FHMV95]

Sender and receiver communicating over a faulty line.
What specification to the system?

- $M_{BTP} \models AG((\mathbf{recack} \wedge \mathbf{bit = 0}) \implies K_r \mathbf{bit = 0})$
- $M_{BTP} \models AG((\mathbf{recack} \wedge \mathbf{bit = 0}) \implies K_s(K_r(\mathbf{bit = 0})))$.
- $M_{BTP} \not\models AG((\mathbf{recack} \wedge \mathbf{bit = 0}) \implies C_{s,r}(\mathbf{bit = 0})$.

Dining cryptographers [Ch88,MS02]

1. Each cryptographer flips a coin and observes that coin and the one to his right.
2. If a cryptographer did not pay for dinner he states whether the two coins he can see fell on the same side or not (saying "equal", or "different").
3. If a cryptographer paid for dinner he states the opposite of what prescribed in 2.

Dining cryptos: Specifications

If an even number of "different" is uttered at the table, then the company paid for dinner; if an odd number of "different" is uttered then one agent paid for dinner.

$$AG(\neg paid_1 \rightarrow AX(K_1 \neg \bigwedge_{i=1,2,3} paid_i \vee$$
$$(K_1(paid_2 \vee paid_3) \wedge (\neg K_1 paid_2 \wedge \neg K_1 paid_3)))))$$

$$AG(even \quad \rightarrow \quad AXC_{1,2,3} \neg (paid_1 \quad \vee \quad paid_2 \quad \vee \quad paid_3))$$

MCMAS

- Supports ATLK specifications.
- System description by means of ISPL, a compact description language for interpreted systems.
- State-space symbolically represented via OBDDs.
- Open-source.

State-space explosions

State space grows exponentially with the number of agents and vars used to describe them.

- Abstraction [CDLR08].
- Symmetry Reduction [CCL09-CCQL09].
- BDD-based BMC [JL10].
- BDD-based Parallel Verification [KLQ10].
- Parameterised verification [KL13a,KL13b] via cutoffs.

Various experimental builds.

Experimental results (AMD quad-core 9600B)

Table: Dining cryptos: $AG(even \rightarrow C_\Gamma(\bigwedge \neg paid_i))$

N	States	Seq	Semi	Simple	Merge	Full
10	45056	21s	11s	12s	6s	5s
		20MB	67MB	69MB	60MB	58MB
14	9.8×10^5	128s	26s	56s	28s	15s
		51MB	91MB	99MB	83MB	84MB
18	2.0×10^7	160s	149s	186s	21s	48s
		55MB	174MB	159MB	82MB	96MB
22	3.9×10^8	2098s	6783s	6622s	85s	85s
		127MB	357MB	353MB	126MB	149MB
26	7.2×10^9	365s	161s	184s	58s	55s
		58MB	176MB	170MB	117MB	164MB
30	1.3×10^{11}	2823s	12771s	12009s	160s	496s
		105MB	427MB	412MB	176MB	205MB

Applications

- Verification of diagnosability mechnanisms for an autonomous underwater vehicle.
- Verification of authentication protocols via automatic compilation from CAPL into ISPL. Verification of anonymity properties (untraceability, etc) in communication protocols.
- Verification of web-services composition in the context of contract-regulated evolutions.
- Verification of data-aware services, including artifact-centric systems.

Conclusions

- Symbolic verification for expressive specifications, including epistemic and ATL-based.
- Performance in line with other symbolic checkers.
- Proven useful in a number of scenarios where high-level expressive, intuitive specifications.
- Several techniques to deal with state-explosion problem including some initial approaches to unbounded number of components.
- Open-source implementation available from `vas.doc.ic.ac.uk`.