Abstraction for Model Checking Modular Interpreted Systems over ATL

Michael Köster and Peter Lohmann

Computational Intelligence Group,
Clausthal University of Technology
Singapore, May 11, 2011

Outline

Motivation

Model Checking

Advantages:

- Faster than **Theorem Proving**

Disadvantages:

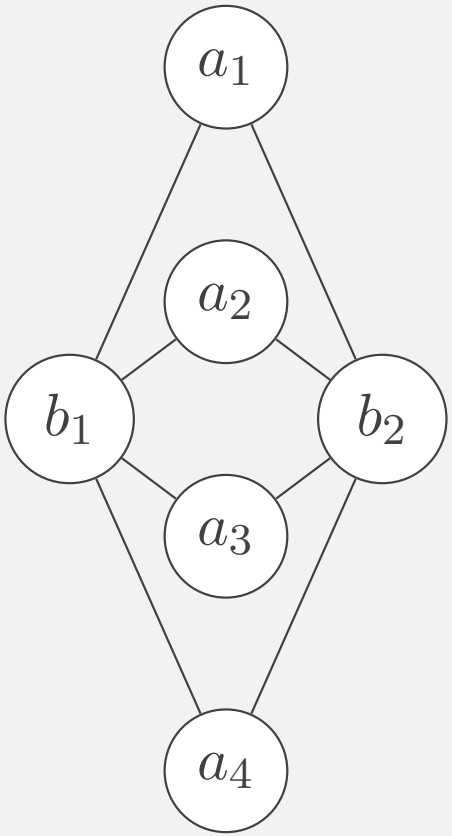
- Explicit representation: a lot of states, in practise not usable
- Compact representation: model checking complexity increases

Idea:

- Reduce the state space before model checking

Communication between Agents

Example 1 (Communicating Agents)

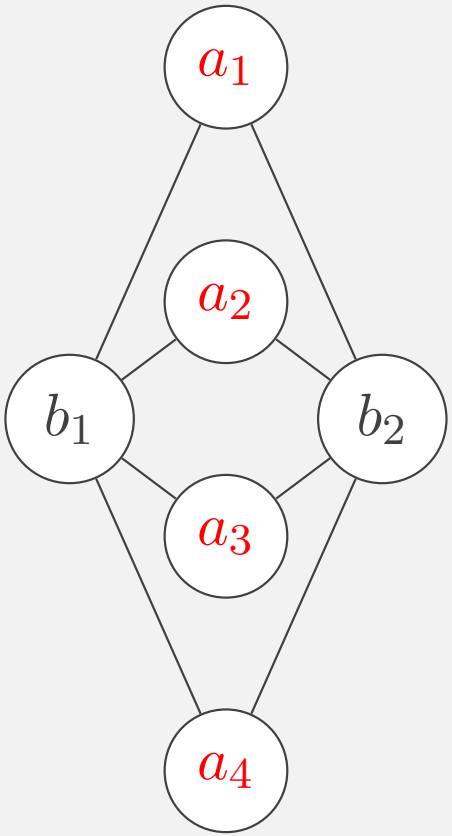


- Agents of team B are not allowed to send a message back.
- If an agent b_j has received a message from a_i then it has to send it to some agent a_k in the following round, $k > i$.

Is it possible for team A to ensure that a_4 will know the message eventually?

Communication between Agents

Example 1 (Communicating Agents)

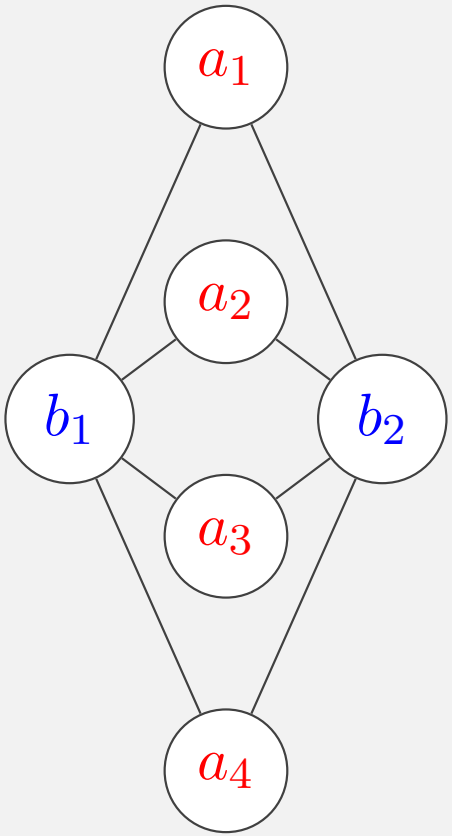


- Agents of team B are not allowed to send a message back.
- If an agent b_j has received a message from a_i then it has to send it to some agent a_k in the following round, $k > i$.

Is it possible for team A to ensure that a_4 will know the message eventually?

Communication between Agents

Example 1 (Communicating Agents)

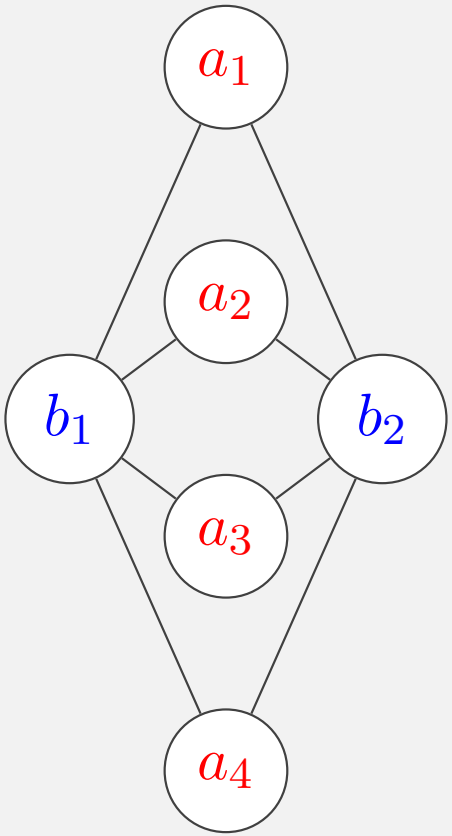


- Agents of team B are not allowed to send a message back.
- If an agent b_j has received a message from a_i then it has to send it to some agent a_k in the following round, $k > i$.

Is it possible for team A to ensure that a_4 will know the message eventually?

Communication between Agents

Example 1 (Communicating Agents)

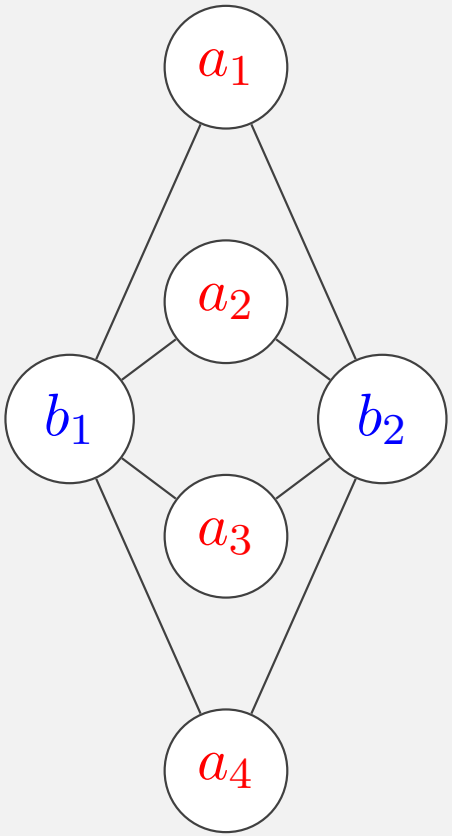


- Agents of team B are not allowed to send a message back.
- If an agent b_j has received a message from a_i then it has to send it to some agent a_k in the following round, $k > i$.

Is it possible for team A to ensure that a_4 will know the message eventually?

Communication between Agents

Example 1 (Communicating Agents)



- Agents of team B are not allowed to send a message back.
- If an agent b_j has received a message from a_i then it has to send it to some agent a_k in the following round, $k > i$.

Is it possible for team A to ensure that a_4 will know the message eventually?

Issues

How can we

- formalize such a system?
- deal with big systems?
- formulate minimal properties (fairness, liveness, safety)?
- ensure that the minimal properties hold?

Issues

How can we

- formalize such a system?
  Modular Interpreted System (MIS)
- deal with big systems?
- formulate minimal properties (fairness, liveness, safety)?
- ensure that the minimal properties hold?

Issues

How can we

- formalize such a system?
  Modular Interpreted System (MIS)
- deal with big systems?
  Abstraction for MIS
- formulate minimal properties (fairness, liveness, safety)?
- ensure that the minimal properties hold?

Issues

How can we

- formalize such a system?
  Modular Interpreted System (MIS)
- deal with big systems?
  Abstraction for MIS
- formulate minimal properties (fairness, liveness, safety)?
  Alternating-Time Temporal Logic (ATL)
- ensure that the minimal properties hold?

Issues

How can we

- formalize such a system?
  Modular Interpreted System (MIS)
- deal with big systems?
  Abstraction for MIS
- formulate minimal properties (fairness, liveness, safety)?
  Alternating-Time Temporal Logic (ATL)
- ensure that the minimal properties hold?
  Model Checking

Modular Interpreted System

Modular Interpreted System

Idea:

- Several **loosely** connected components.
- Work is done in the components.
- Little **interaction** between the components.

Modular Interpreted System

Definition 2 (Modular Interpreted System (MIS))

A MIS is a tuple $S = (\mathbb{A}\text{gt}, Act, \mathcal{I}n)$ where:

- $\mathbb{A}\text{gt} = \{a_1, \ldots, a_k\}$ agents,
- Act actions,
- $\mathcal{I}n$ interaction alphabet.

Modular Interpreted System

Definition 2 (Modular Interpreted System (MIS))

A MIS is a tuple $S = (\mathbb{A}\mathrm{gt}, Act, \mathcal{I}n)$ where:

- $\mathbb{A}\mathrm{gt} = \{a_1, \ldots, a_k\}$ agents,
- Act actions,
- $\mathcal{I}n$ interaction alphabet.

Example 3

- $\mathbb{A}\mathrm{gt} = \{a_1, a_2, a_3, a_4, b_1, b_2\}$,
- $Act = \{\mathrm{send_x} \mid \mathrm{x} \in \mathbb{A}\mathrm{gt}\} \cup \{\mathrm{noop}\}$
- $\mathcal{I}n = \{\mathbf{nothing}, \mathbf{m_{a_1}}, \mathbf{m_{a_2}}, \mathbf{m_{a_3}}, \mathbf{m_{a_4}}\} \cup \mathrm{P}(\{\mathbf{m_{a_i b_j}} \mid i \in \{1, \ldots, 4\}, j \in \{1, 2\}\}))$

MIS Agent I

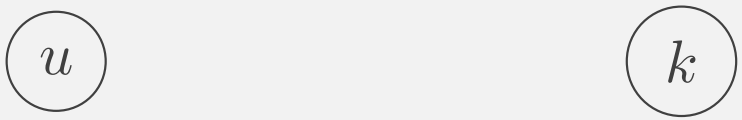
Definition 4 (MIS Agent)

Each agent has the following **internal structure**:

$$a_i = (St_i, \quad , \quad , \quad , \quad , \quad)$$

The global state space is defined as $St := St_1 \times \cdots \times St_k$.

Example 5 (Agent a_1)



MIS Agent I

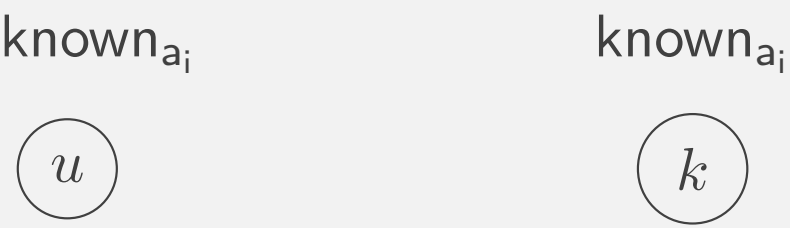
Definition 4 (MIS Agent)

Each agent has the following **internal structure**:

$$a_i = (St_i, \quad , \quad , \quad , \quad , \Pi_i, \pi_i)$$

The global state space is defined as $St := St_1 \times \cdots \times St_k$.

Example 5 (Agent a_1)



MIS Agent I

Definition 4 (MIS Agent)

Each agent has the following **internal structure**:

$$a_i = (St_i, d_i, \quad , \quad , \quad , \Pi_i, \pi_i)$$

The global state space is defined as $St := St_1 \times \cdots \times St_k$.

Example 5 (Agent a_1)

MIS Agent I

Definition 4 (MIS Agent)

Each agent has the following **internal structure**:

$$a_i = (St_i, d_i, \quad , in_i, o_i, \Pi_i, \pi_i)$$

The global state space is defined as $St := St_1 \times \cdots \times St_k$.

Example 5 (Agent a_1)

MIS Agent I

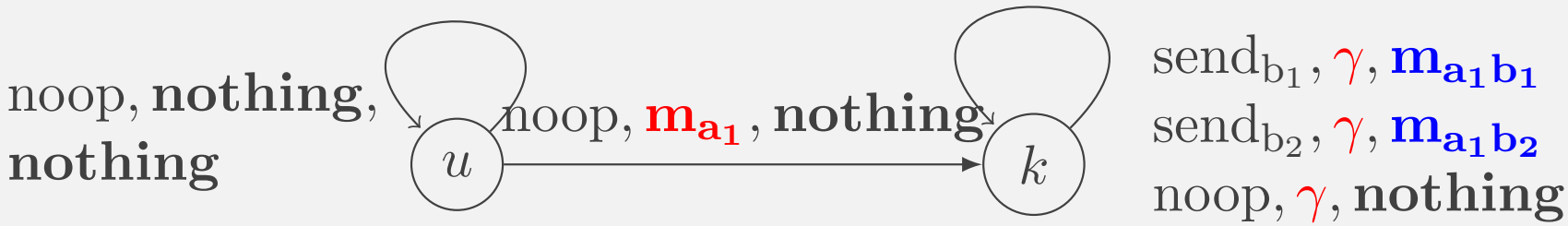
Definition 4 (MIS Agent)

Each agent has the following **internal structure**:

$$a_i = (St_i, d_i, out_i, in_i, o_i, \Pi_i, \pi_i)$$

The global state space is defined as $St := St_1 \times \cdots \times St_k$.

Example 5 (Agent a_1)



MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with known_{b_j} if the agent received some time ago the message from a_i.
- Others are labeled with unknown_{b_j}.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with known_{b_j} if the agent received some time ago the message from a_i.
- Others are labeled with unknown_{b_j}.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with $\text{known}_{\text{b}_\text{j}}$ if the agent received some time ago the message from a_i.
- Others are labeled with $\text{unknown}_{\text{b}_\text{j}}$.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with $\text{known}_{\text{b}_\text{j}}$ if the agent received some time ago the message from a_i.
- Others are labeled with $\text{unknown}_{\text{b}_\text{j}}$.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with known_{b_j} if the agent received some time ago the message from a_i.
- Others are labeled with unknown_{b_j}.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

Example 6 (Agents b_j)

- 256 states
- Every state is labeled with $\text{known}_{\text{b}_\text{j}}$ if the agent received some time ago the message from a_i.
- Others are labeled with $\text{unknown}_{\text{b}_\text{j}}$.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

MIS Agent II

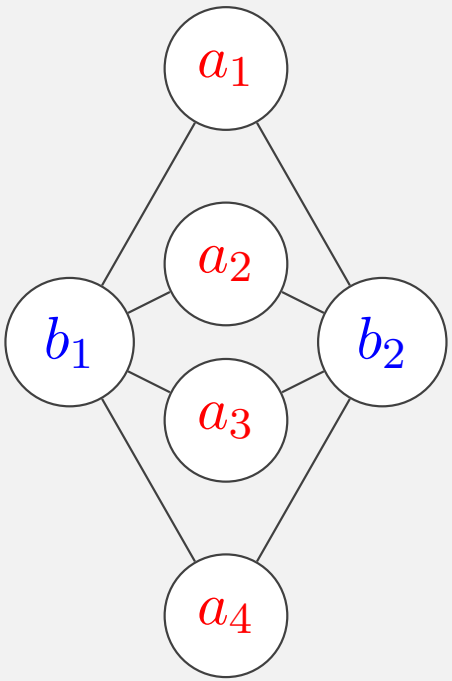
Example 6 (Agents b_j)

- 256 states
- Every state is labeled with $\text{known}_{\text{b}_\text{j}}$ if the agent received some time ago the message from a_i.
- Others are labeled with $\text{unknown}_{\text{b}_\text{j}}$.
- While the agent is waiting for a message it does nothing.
- When it receives a message it has to send the message to one of the opponents with a higher number then i.
- If the agent received the message from each agent a_i it does nothing.

Advantages of Modular Interpreted Systems I

- **Modular** (removing, replacing of agents)
- **Interaction** reduced to an abstract **interaction symbol**
- **Computational ground**

Example 7 (Communicating Agents)

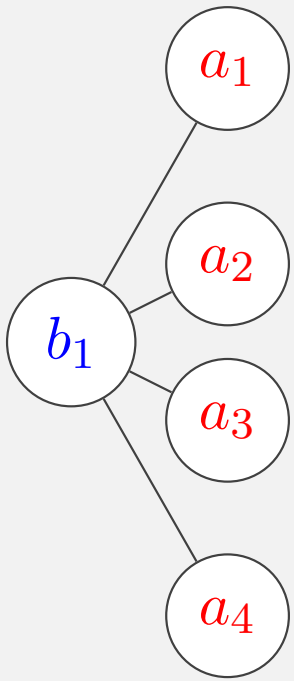


Is it possible for team A to ensure that a_4 will know the message eventually?

Advantages of Modular Interpreted Systems I

- **Modular** (removing, replacing of agents)
- **Interaction** reduced to an abstract **interaction symbol**
- **Computational ground**

Example 7 (Communicating Agents)



Is it possible for team A to ensure that a_4 will know the message eventually?

Advantages of Modular Interpreted Systems II

Example 8 (Communicating Agents)

- Simple example
- $256 + 256 + 2 + 2 + 2 + 2 = 520$ states
- Remove agent b_2: 264 states

Specification Logic: ATL

Alternating-time Temporal Logic

Definition 9 (Alternating-time temporal Logic (ATL))

The language of plain ATL is defined over the non-empty sets:

- Π of **Propositions** $\mathsf{p} \in \Pi$
- $\mathbb{A}\mathrm{gt}$ of **Agents** $A \subseteq \mathbb{A}\mathrm{gt}$

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\langle A \rangle\rangle \mathbf{X}\varphi \mid \langle\langle A \rangle\rangle \mathbf{G}\varphi \mid \langle\langle A \rangle\rangle \varphi \mathbf{U} \varphi.$$

Main Idea: cooperation modalities

- $\langle\langle A \rangle\rangle \mathbf{X}\varphi$ "coalition A has a collective strategy to enforce that φ holds after the next step"

Alternating-time Temporal Logic

Definition 9 (Alternating-time temporal Logic (ATL))

The language of plain ATL is defined over the non-empty sets:

- Π of **Propositions** $\mathsf{p} \in \Pi$
- $\mathbb{A}\mathrm{gt}$ of **Agents** $A \subseteq \mathbb{A}\mathrm{gt}$

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\langle A \rangle\rangle \mathbf{X}\varphi \mid \langle\langle A \rangle\rangle \mathbf{G}\varphi \mid \langle\langle A \rangle\rangle \varphi \mathbf{U} \varphi.$$

Main Idea: **cooperation modalities**

- $\langle\langle A \rangle\rangle \mathbf{X}\varphi$ "coalition A has a collective strategy to enforce that φ holds after the next step"

Example 10

Is it possible for team A to ensure that a_4 will know the message eventually?

$$S, \mathrm{q} \models \langle\langle A \rangle\rangle (\top \mathbf{U} \mathsf{known}_{\mathsf{a}_4})$$

Abstraction for MIS

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:
  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:
  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:
  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function:**
  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:

  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:

  Input: **abstract state**, **action**, **influence symbol**

  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea I

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Reducing all equivalent states to just one
- Agents get **fewer choices** and the **opponents more choices**
- **Influence symbols** of the abstract states:
  - all influence symbols that are an outcome of executing a action in each concrete state
- **Local transition function**:

  Input: **abstract state**, **action**, **influence symbol**
  Output: **abstract state**
  - unfold both equivalence classes
  - connection between concrete state of 1st equivalence class to concrete state of 2nd equivalence class?

Basic Idea II

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Labeling:
  - Assigning a proposition to an abstract state if all concrete states in the equivalence class are labeled with that proposition
  - If a proposition only holds in some states of the class we remove it from set of propositions

Basic Idea II

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Labeling:
  - Assigning a proposition to an abstract state if all concrete states in the equivalence class are labeled with that proposition
  - If a proposition only holds in some states of the class we remove it from set of propositions

Basic Idea II

Partitioning the **local state space** by using **handcrafted equivalence relation**:

- Labeling:
  - Assigning a proposition to an abstract state if all concrete states in the equivalence class are labeled with that proposition
  - If a proposition only holds in some states of the class we remove it from set of propositions

Example for Abstraction

Example 11 (Agents b_j)

- 256 states

$$
\begin{aligned}
S_i &:= \{(R,N) \mid \emptyset \neq R \subseteq \{r_1,\ldots,r_i\}, n_i \in N\} \setminus \textstyle\bigcup_{j=1}^{i-1} S_j \\
&\quad \text{for } i = 1,\ldots,3 \\
S_{\text{rest}} &:= \{(R,N) \mid (R,N) \notin S_1 \cup \cdots \cup S_3\}
\end{aligned}
$$

Result: **4 states**

Advantages of Abstraction of Modular Interpreted Systems

Example 12 (Communicating Agents)

- $256 + 256 + 2 + 2 + 2 + 2 = 520$ states
- Remove agent b_2: 264 states
- Abstraction of b_1: 12 states

The Model Checking Algorithm

Idea

- **Input**:
  - MIS S
  - $init$ of global states of S
  - ATL formula φ
  - for each quantifier subformulae an abstraction relation
- **Output**:
  - true : if $S, q \models \varphi$ for all $q \in init$
  - unknown : we do not know whether S satisfies φ or not

Idea

- **Input**:
  - MIS S
  - $init$ of global states of S
  - ATL formula φ
  - for each quantifier subformulae an abstraction relation
- **Output**:
  - true : if $S, q \models \varphi$ for all $q \in init$
  - unknown : we do not know whether S satisfies φ or not

Complexity and Soundness

Theorem 13

Algorithm modelcheck$(S, init, \varphi, (\equiv_\psi)_{\psi \in \mathrm{qsf}(\varphi)})$ *runs in time*

$$O\left(|init| + |S| \cdot |\varphi|\right) \cdot 2^{O\left(\sum_{\psi \in \mathrm{qsf}(\varphi)} \left| S^{[\![\psi]\!]}_{\equiv_\psi} \right|\right)}$$

where $|S|$ *denotes the size of the* MIS S *in a compact representation. The cardinality of the global state space of* S *may then be upto* $2^{\Theta(|S|)}$.

Theorem 14

Algorithm modelcheck is sound, i.e. if modelcheck$(S, init, \varphi, (\equiv_\psi)_{\psi \in \mathrm{qsf}(\varphi)})$ *outputs* true *then* $S, q \models \varphi$ *for all* $q \in init$.

Conclusion

Conclusion

- **Modular Interpreted Systems** facilitates **modularity**
- **ATL** allows to talk about strategic properties
- The **abstraction for MIS** :
  - is sound
  - allows to deal with bigger Systems
    (depending on the equivalent relations)

Thank you for your attention!

Questions?

Conclusion

- **Modular Interpreted Systems** facilitates **modularity**
- **ATL** allows to talk about strategic properties
- The **abstraction for MIS** :
  - is sound
  - allows to deal with bigger Systems
    (depending on the equivalent relations)

Thank you for your attention!

Questions?

Conclusion

- **Modular Interpreted Systems** facilitates **modularity**
- **ATL** allows to talk about strategic properties
- The **abstraction for MIS** :
  - is sound
  - allows to deal with bigger Systems
    (depending on the equivalent relations)

Thank you for your attention!

Questions?

References


W. Jamroga, T. Agotnes
Modular Interpreted Systems
Proceedings of AAMAS'07, pp. 892-899.


Rajeev Alur and Thomas A. Henzinger and Orna Kupferman
Alternating-time temporal logic
J. ACM, Volume 49, Number 5, 2002, pp. 672-713.


Michael Köster, Peter Lohmann
Abstraction for Model Checking Modular Interpreted Systems over ATL (Extended Abstract)
Proceedings of AAMAS'11


Michael Köster, Peter Lohmann
Abstraction for Model Checking Modular Interpreted Systems over ATL
Proceedings of Programming Multi-Agent Systems, AAMAS'11