

Spotlight Abstraction of Agents and Areas

Tobe Toben, OFFIS e.V., Oldenburg, Germany
Joint work with Bernd Westphal and Jan Rakow

03.02.2010, Dagstuhl Seminar 10051 on
“Quantitative and Qualitative Analysis of Network Protocols”

Intro

Abstraction

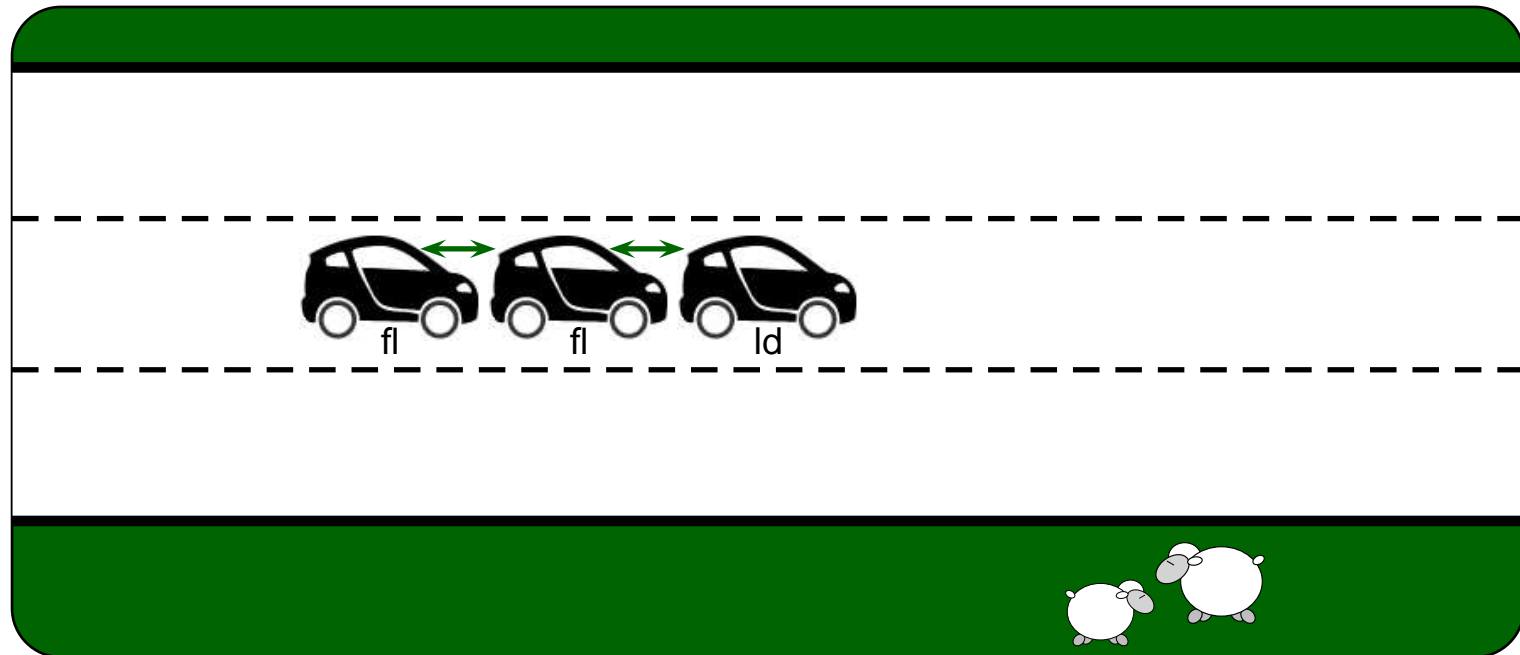
Agents

Areas

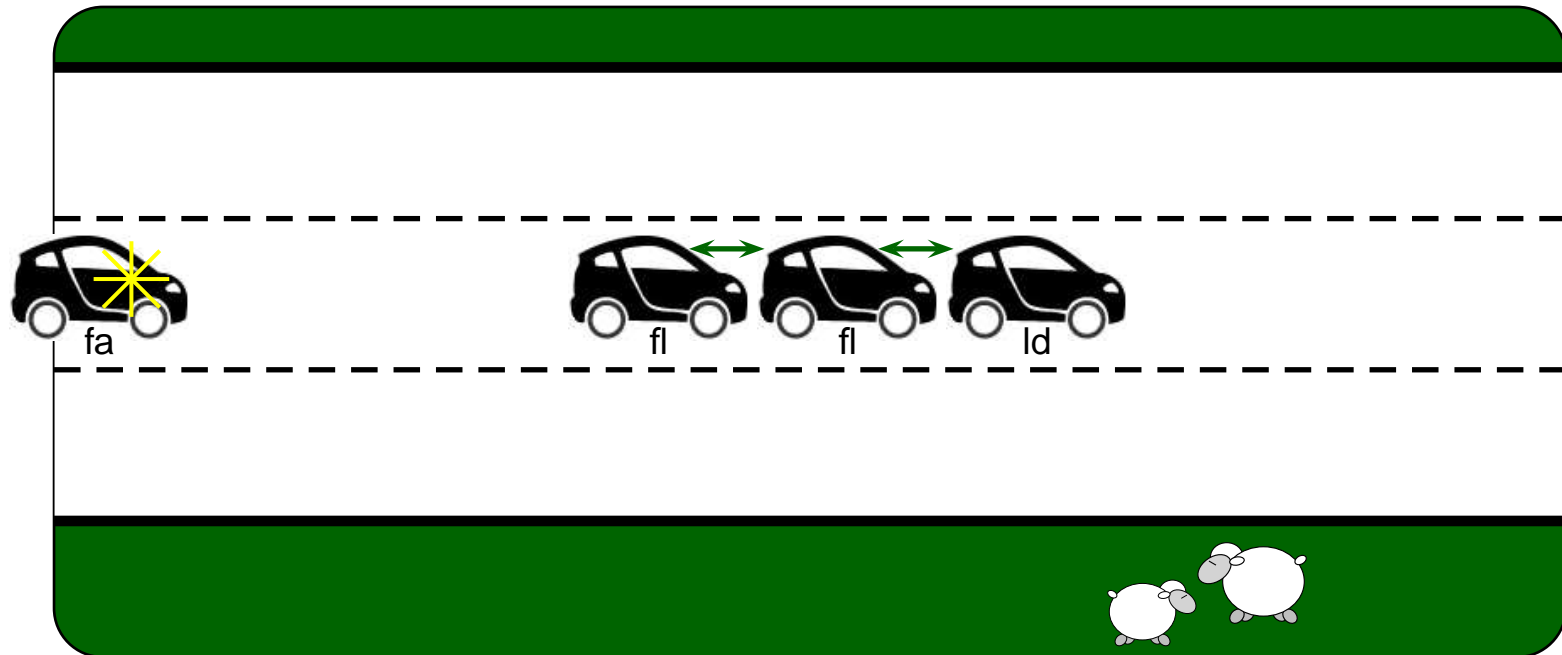
End

We study *abstraction techniques* for systems comprising an unbounded number of communicating agents (or objects, processes, ...).

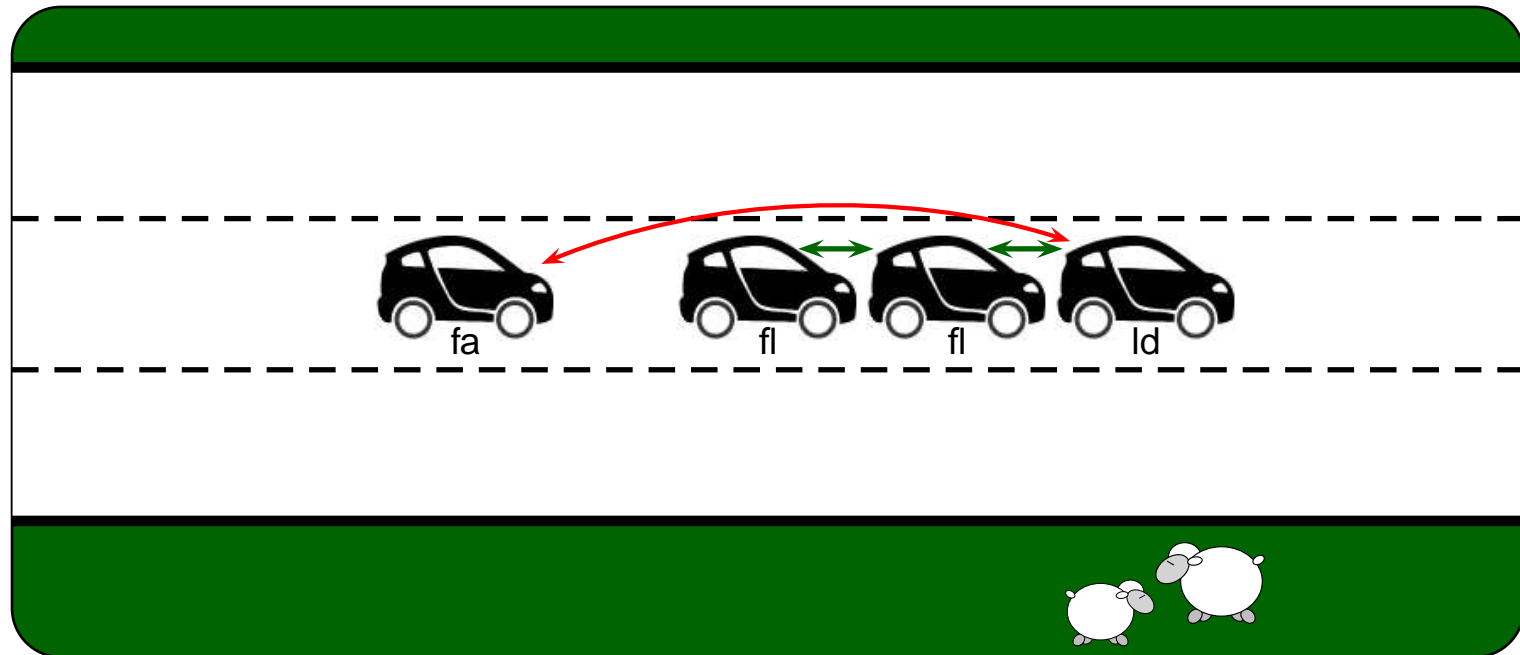
We study *abstraction techniques* for systems comprising an unbounded number of communicating agents (or objects, processes, ...).



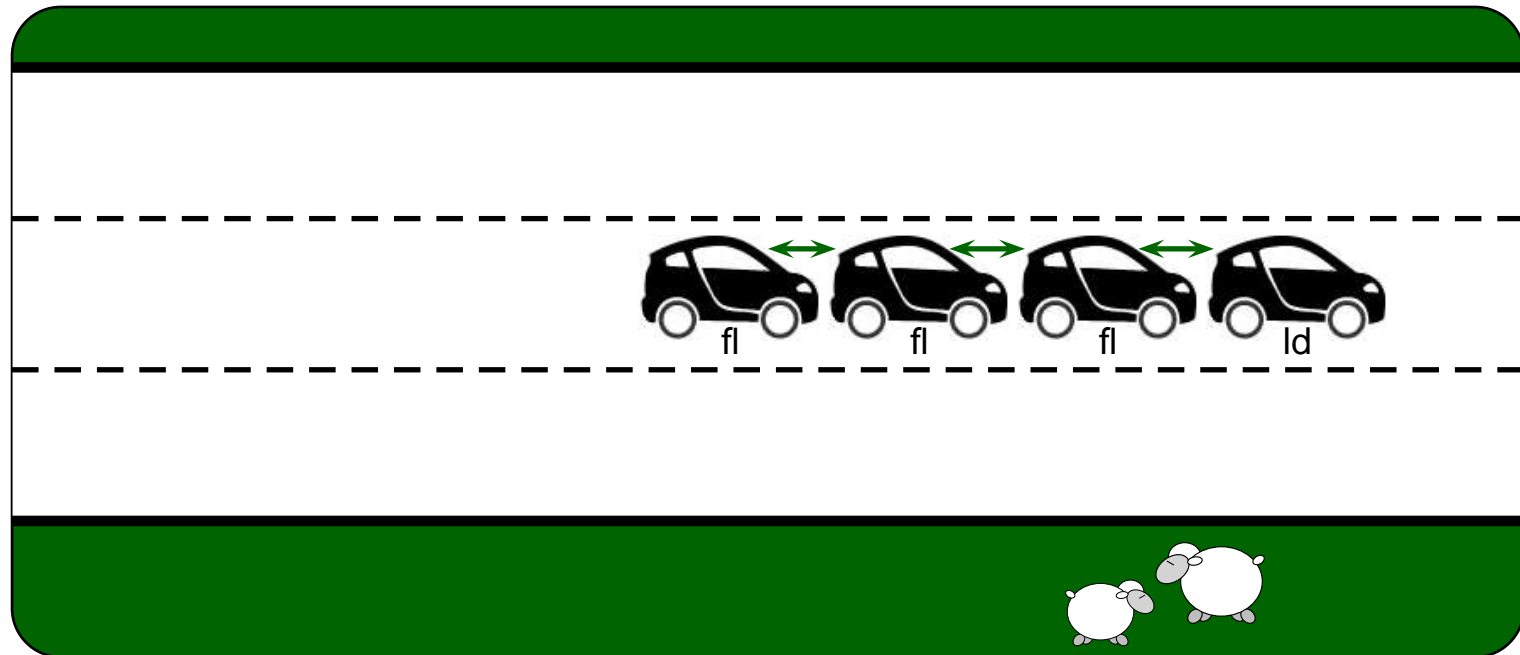
We study *abstraction techniques* for systems comprising an unbounded number of communicating agents (or objects, processes, ...).



We study *abstraction techniques* for systems comprising an unbounded number of communicating agents (or objects, processes, ...).



We study *abstraction techniques* for systems comprising an unbounded number of communicating agents (or objects, processes, ...).



Intro

Abstraction

Agents

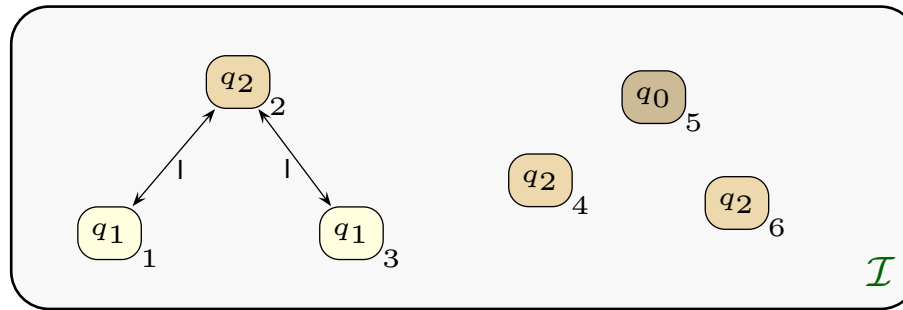
Areas

End

- \mathcal{A} is a set of agent identities
- \mathcal{P} is a set of predicates
- $\mathcal{I} : \mathcal{P} \times \mathcal{A}^K \rightarrow \mathbb{B}$ is a system state

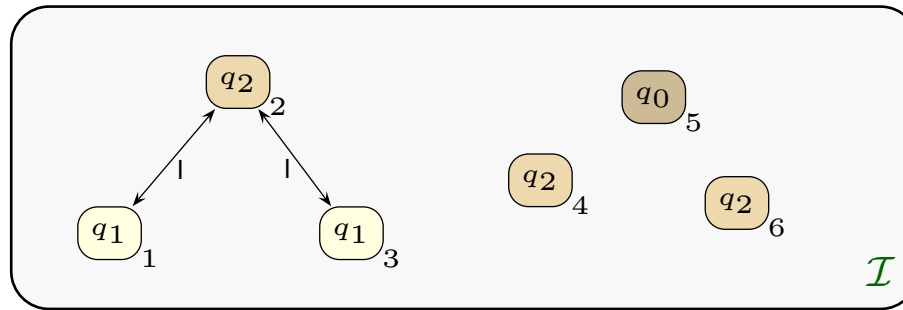
- \mathcal{A} is a set of agent identities
- \mathcal{P} is a set of predicates
- $\mathcal{I} : \mathcal{P} \times \mathcal{A}^K \rightarrow \mathbb{B}$ is a system state

With $\mathcal{A} = \mathbb{N}$ and $\mathcal{P} = \{q_0/1, q_1/1, q_2/1, l/2, \text{alive}/1\}$
a graphical representation of some system state \mathcal{I} is



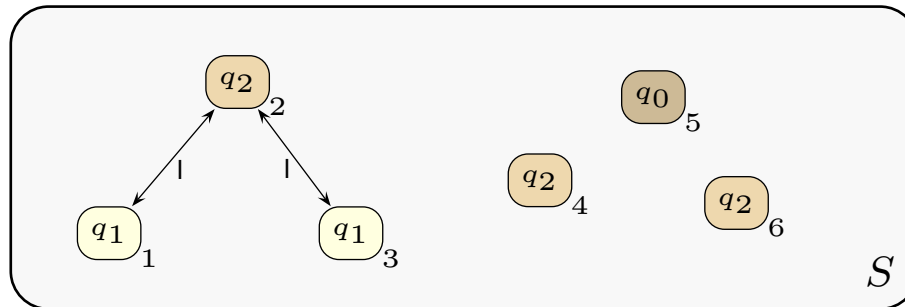
- \mathcal{A} is a set of agent identities
- \mathcal{P} is a set of predicates
- $\mathcal{I} : \mathcal{P} \times \mathcal{A}^K \rightarrow \mathbb{B}$ is a system state

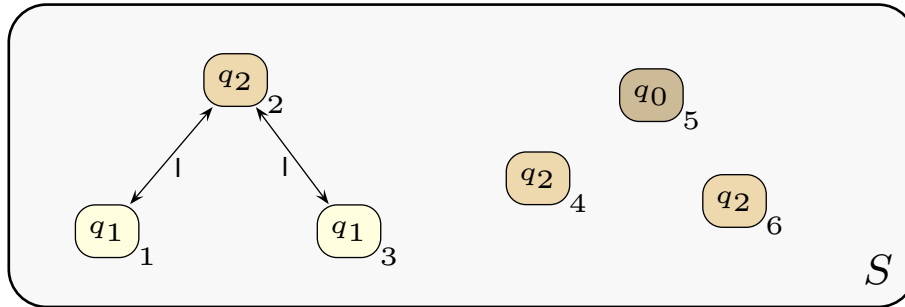
With $\mathcal{A} = \mathbb{N}$ and $\mathcal{P} = \{q_0/1, q_1/1, q_2/1, l/2, \text{alive}/1\}$
a graphical representation of some system state \mathcal{I} is



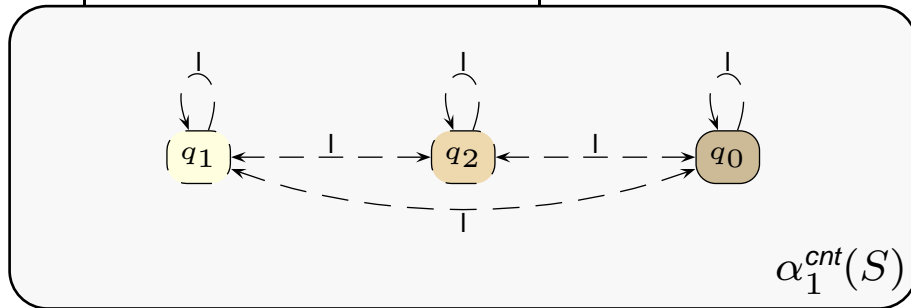
(Spotlight) Abstraction maps two-valued states to three-valued states.

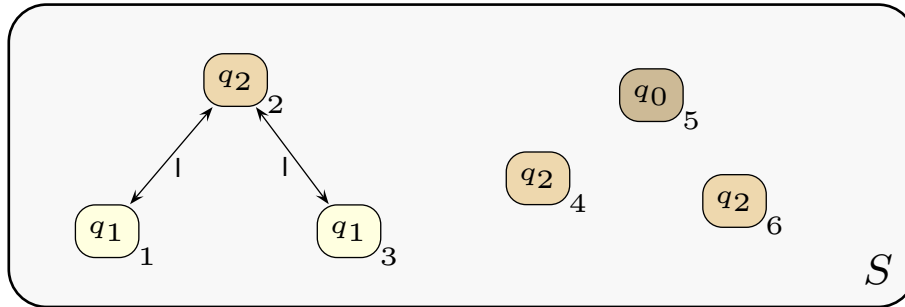
$$\alpha_P(\mathcal{I}) : \mathcal{P} \times \mathcal{A}^K \mapsto \mathbb{B} \cup \{1/2\}$$



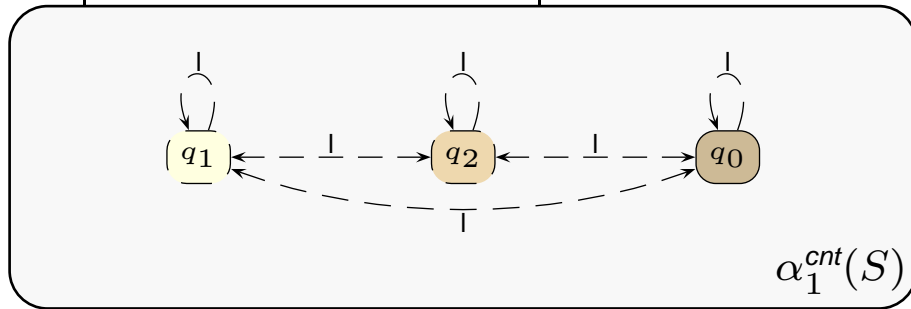


Counter Abstraction [GS92]

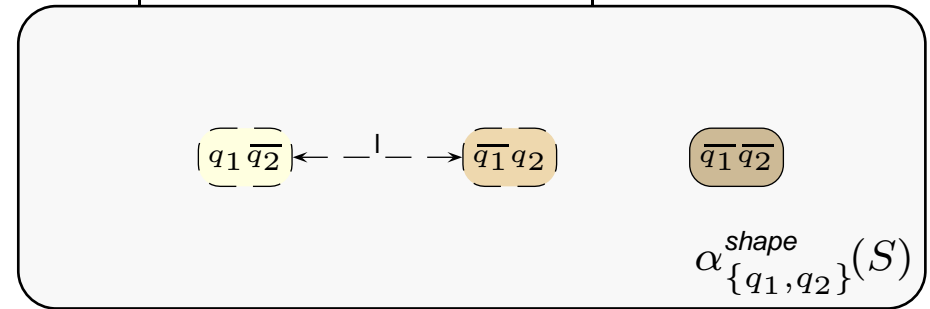


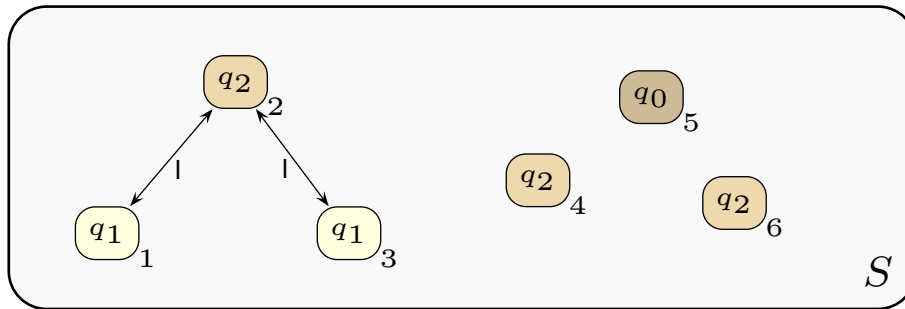


Counter Abstraction [GS92]

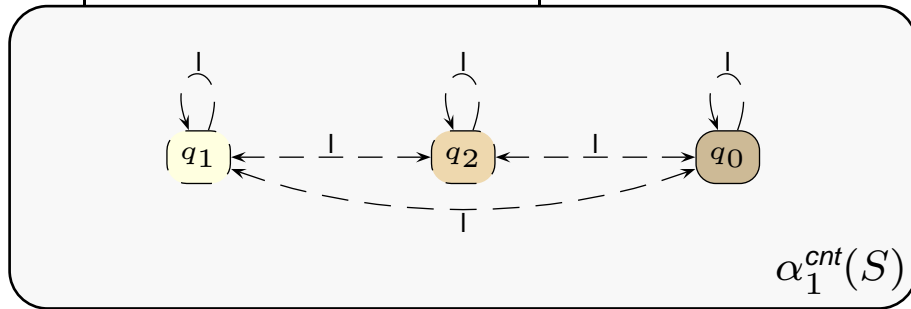


Shape Abstraction [SRW02]

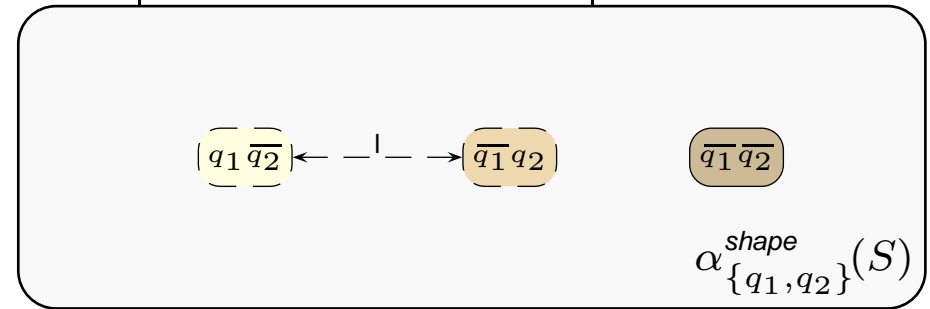




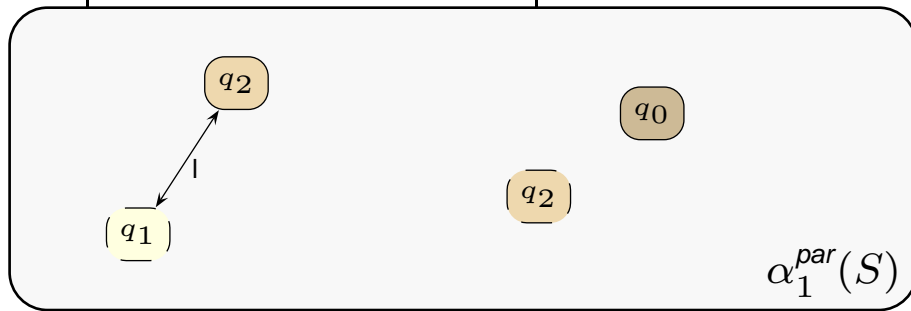
Counter Abstraction [GS92]

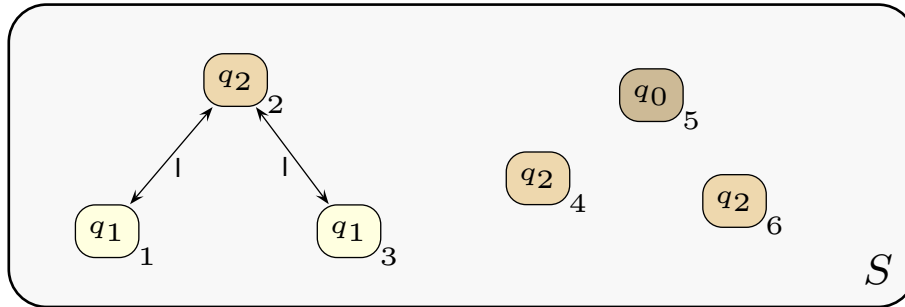


Shape Abstraction [SRW02]

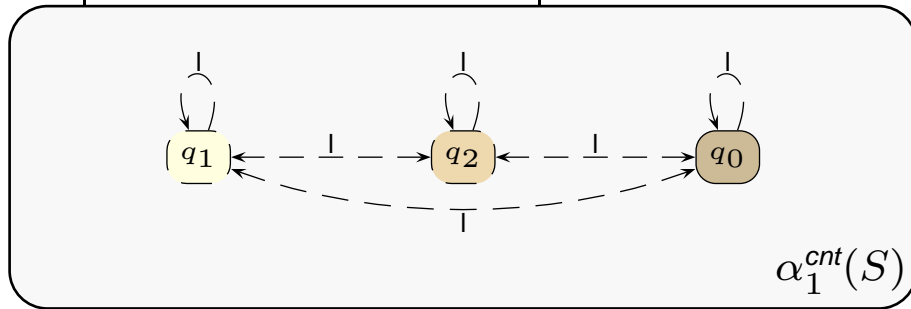


Partner Abstraction [BW07]

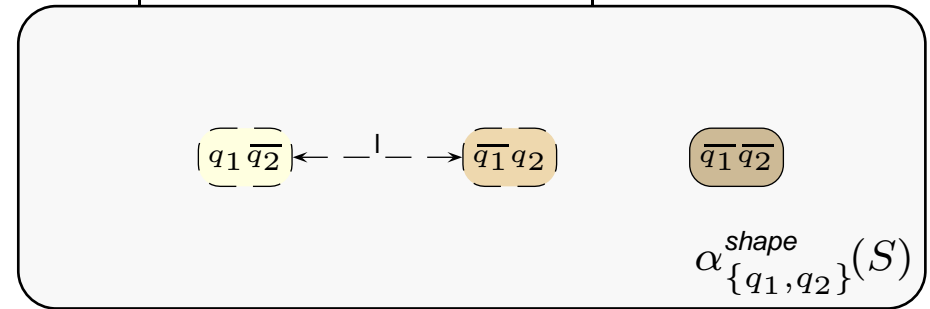




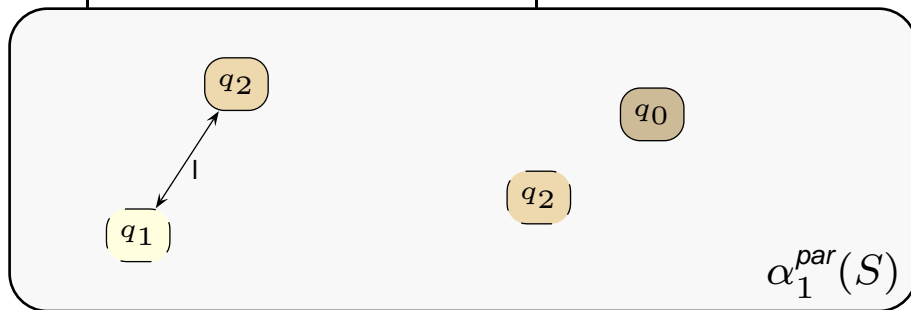
Counter Abstraction [GS92]



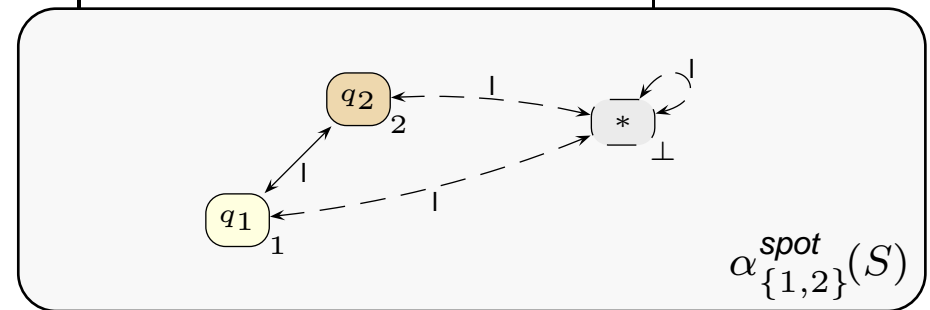
Shape Abstraction [SRW02]

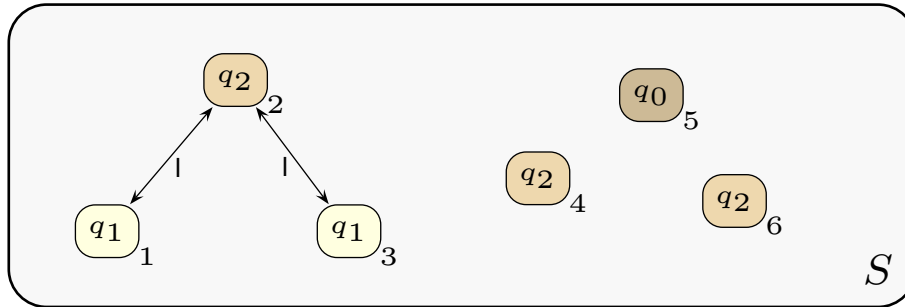


Partner Abstraction [BW07]

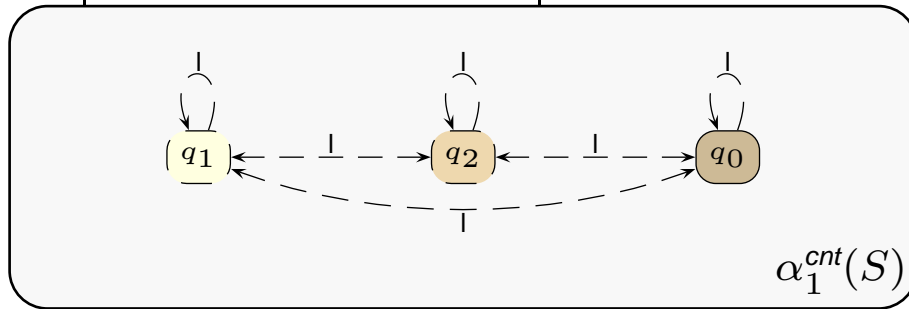


Spotlight Abstraction [McM99, WW07]

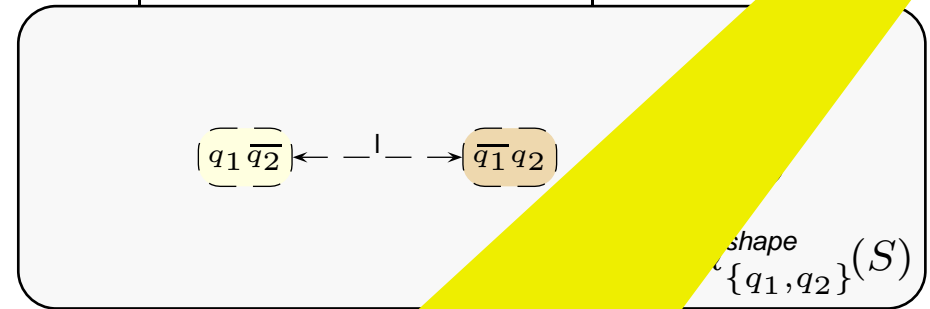




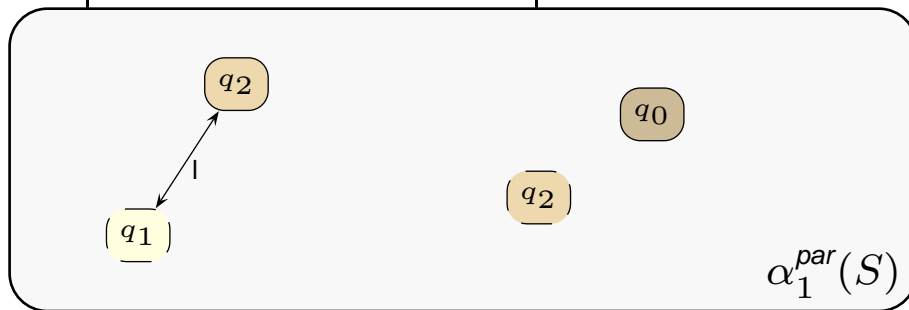
Counter Abstraction [GS92]



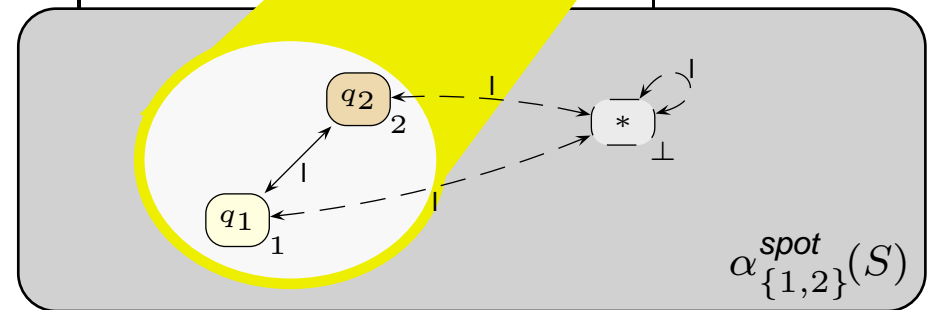
Shape Abstraction [SRW02]



Partner Abstraction [BW07]



Spotlight Abstraction [BW07]



Intro

Abstraction

Agents

Areas

End

A spotlight is a subset of agents $S \subset A$.

A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

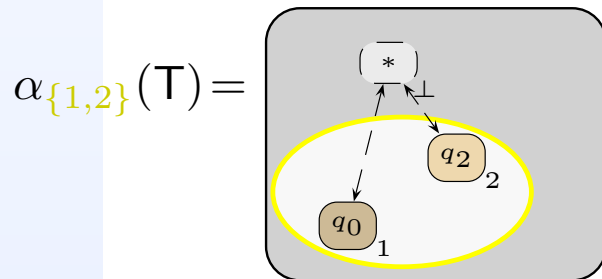
$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

In practice, one does not compute the *state abstraction* but rather the **abstract transition system**:

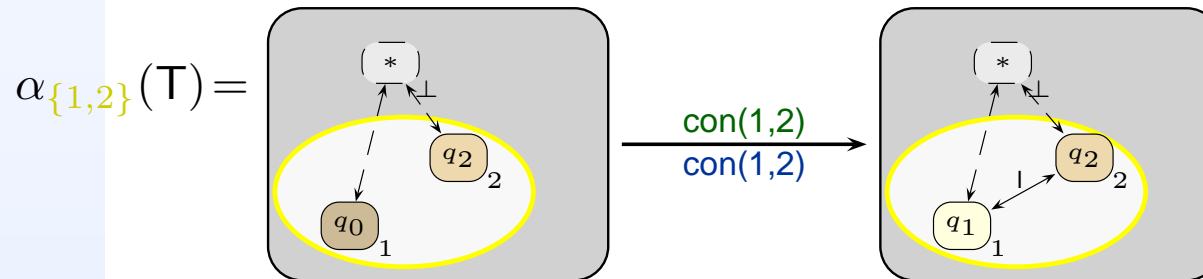


A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

In practice, one does not compute the *state abstraction* but rather the **abstract transition system**:

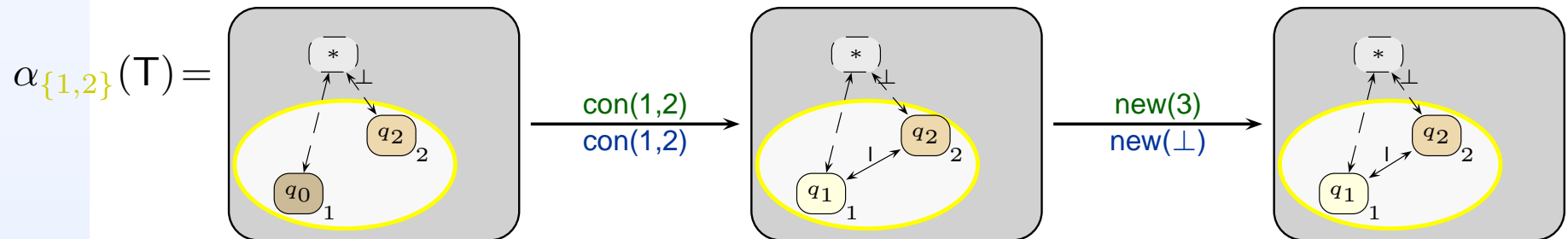


A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

In practice, one does not compute the *state abstraction* but rather the **abstract transition system**:

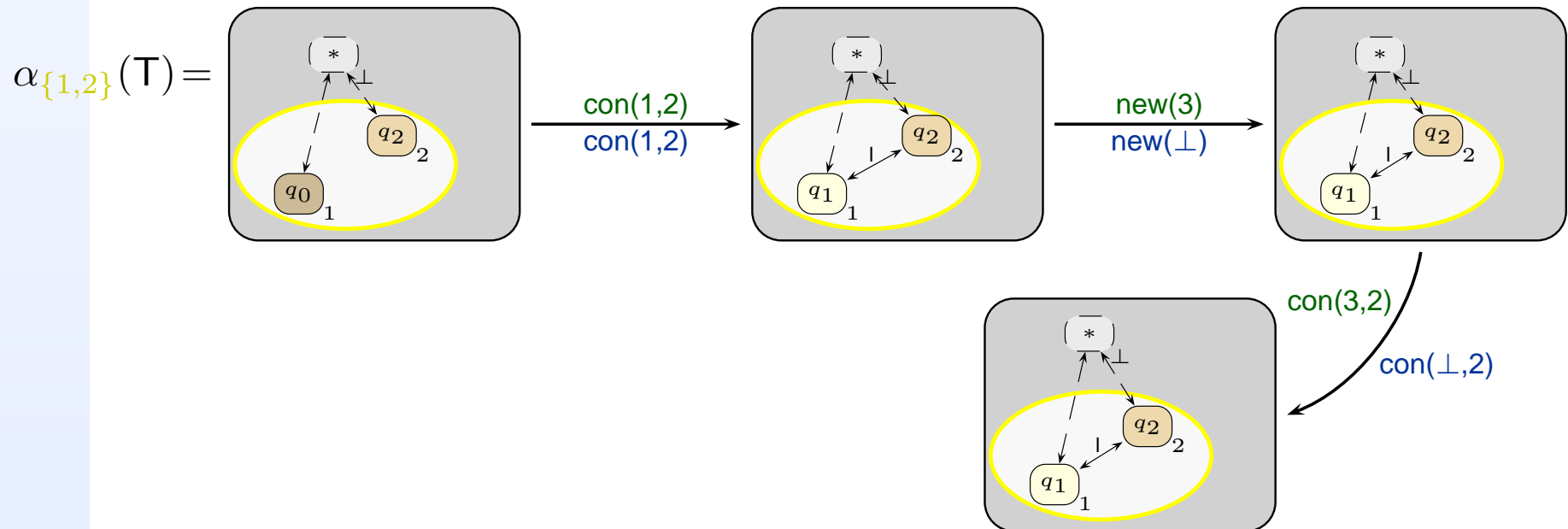


A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

In practice, one does not compute the *state abstraction* but rather the **abstract transition system**:

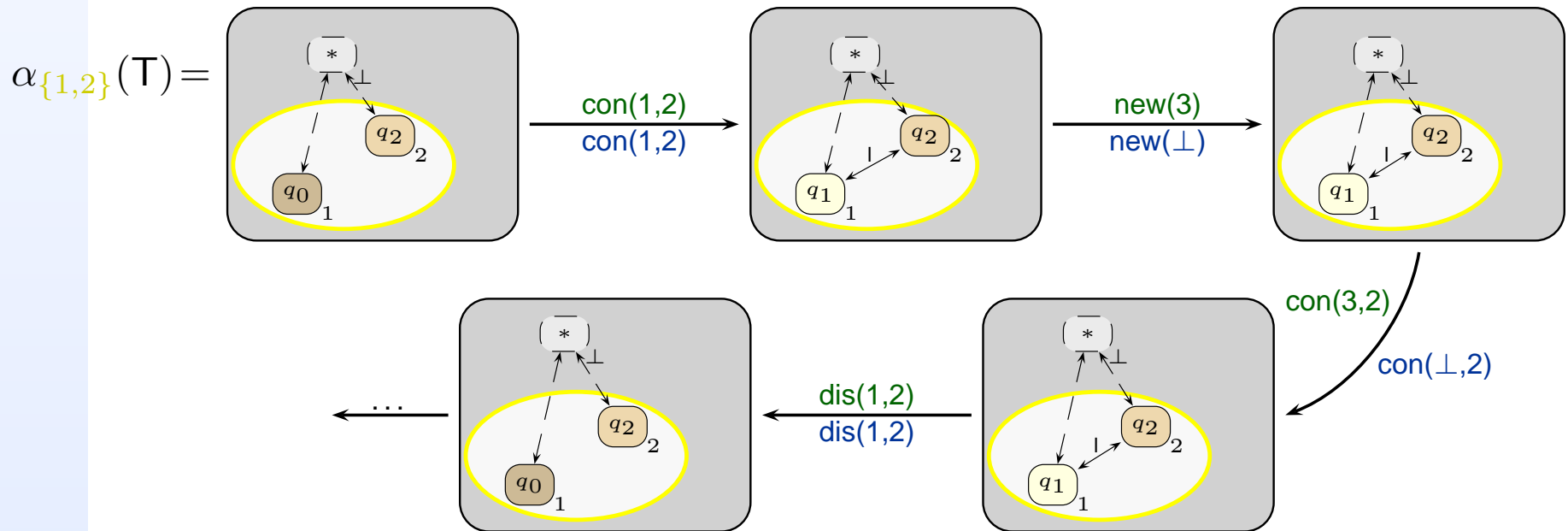


A spotlight is a subset of agents $\mathcal{S} \subset \mathcal{A}$.

The spotlight abstraction of state \mathcal{I} under \mathcal{S} , denoted $\alpha_{\mathcal{S}}(\mathcal{I})$, is

$$\alpha_{\mathcal{S}}(\mathcal{I})(p, a_1, \dots, a_k) := \begin{cases} \mathcal{I}(p, a_1, \dots, a_k) & \text{if } \{a_1, \dots, a_k\} \subseteq \mathcal{S} \\ 1/2 & \text{else} \end{cases}$$

In practice, one does not compute the *state abstraction* but rather the **abstract transition system**:



Intro

Abstraction

Agents

Areas

End

- For a fixed spotlight \mathcal{S} , the best abstract transformer can be effectively computed.

Intro

Abstraction

Agents

Areas

End

- For a fixed spotlight \mathcal{S} , the best abstract transformer can be effectively computed.
- The abstract transition system $\alpha_{\mathcal{S}}(T)$ comprises both **over-** and **under-**approximation:

$$T \preceq \alpha_{\mathcal{S}}(T) \quad \text{and} \quad \text{Runs}(T|_{\mathcal{S}}) \subseteq \text{Runs}(\alpha_{\mathcal{S}}(T))$$

- For a fixed spotlight \mathcal{S} , the best abstract transformer can be effectively computed.
- The abstract transition system $\alpha_{\mathcal{S}}(\mathbb{T})$ comprises both **over-** and **under-**approximation:

$$\mathbb{T} \preceq \alpha_{\mathcal{S}}(\mathbb{T}) \quad \text{and} \quad \text{Runs}(\mathbb{T}|_{\mathcal{S}}) \subseteq \text{Runs}(\alpha_{\mathcal{S}}(\mathbb{T}))$$

- A new **three-valued** satisfaction relation for temporal specifications yields the following embedding property:

$$\mathbb{T} \llbracket \phi(a_1, \dots, a_n) \rrbracket \subseteq \alpha_{\mathcal{S}}(\mathbb{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket$$

- For a fixed spotlight \mathcal{S} , the best abstract transformer can be effectively computed.
- The abstract transition system $\alpha_{\mathcal{S}}(\mathbb{T})$ comprises both **over-** and **under-**approximation:

$$\mathbb{T} \preceq \alpha_{\mathcal{S}}(\mathbb{T}) \quad \text{and} \quad \text{Runs}(\mathbb{T}|_{\mathcal{S}}) \subseteq \text{Runs}(\alpha_{\mathcal{S}}(\mathbb{T}))$$

- A new **three-valued** satisfaction relation for temporal specifications yields the following embedding property:

$$\mathbb{T} \llbracket \phi(a_1, \dots, a_n) \rrbracket \sqsubseteq \alpha_{\mathcal{S}}(\mathbb{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket$$

- If the behaviour of the agents adhere to (syntactically checkable) **symmetry** criteria, the universal closure can be derived from some representative cases.

$$\mathbb{T} \llbracket \phi'(a_1, \dots, a_n) \rrbracket \iff \forall a_1, \dots, a_n \in \mathcal{A} : \mathbb{T} \llbracket \phi(a_1, \dots, a_n) \rrbracket$$

Intro

Abstraction

Agents

Areas

End

- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathbb{T})[[\phi(a_1, \dots, a_n)]] = 1/2$.

Intro

Abstraction

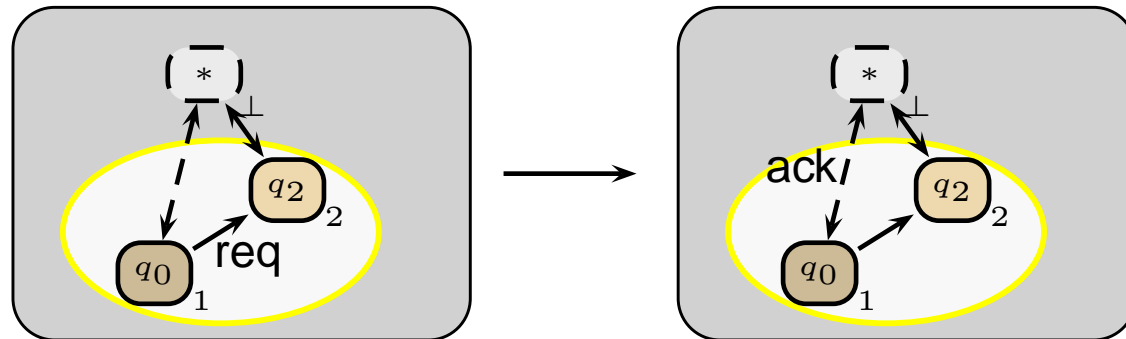
Agents

Areas

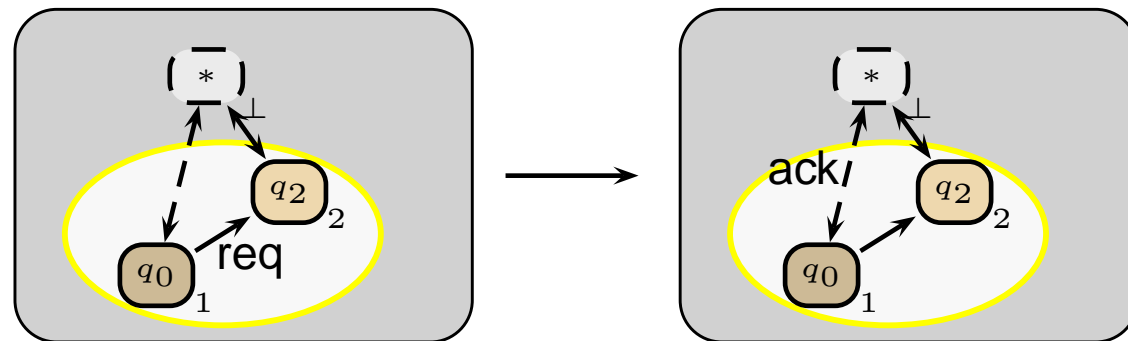
End

- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathbb{T})[[\phi(a_1, \dots, a_n)]] = 1/2$.
- Either the spotlight was too small, or the over-approximation of the "non-spotlight identities" produced **spurious behaviour**.

- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathcal{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket = 1/2$.
- Either the spotlight was too small, or the over-approximation of the "non-spotlight identities" produced **spurious behaviour**.

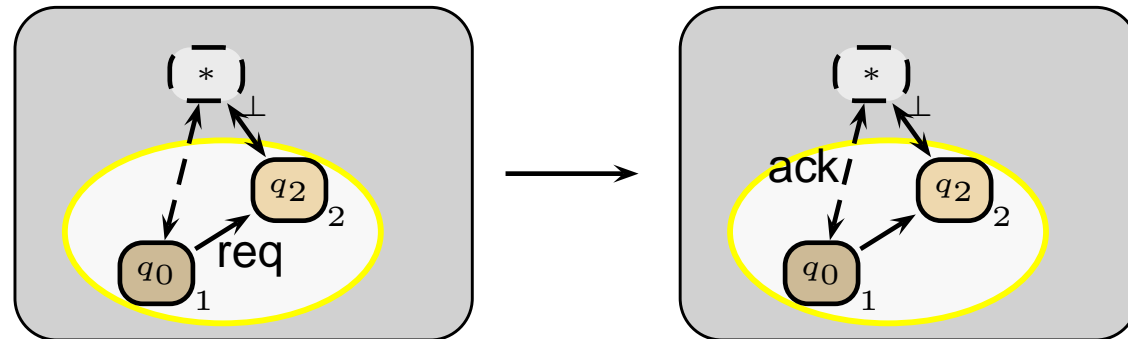


- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathcal{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket = 1/2$.
- Either the spotlight was too small, or the over-approximation of the "non-spotlight identities" produced **spurious behaviour**.



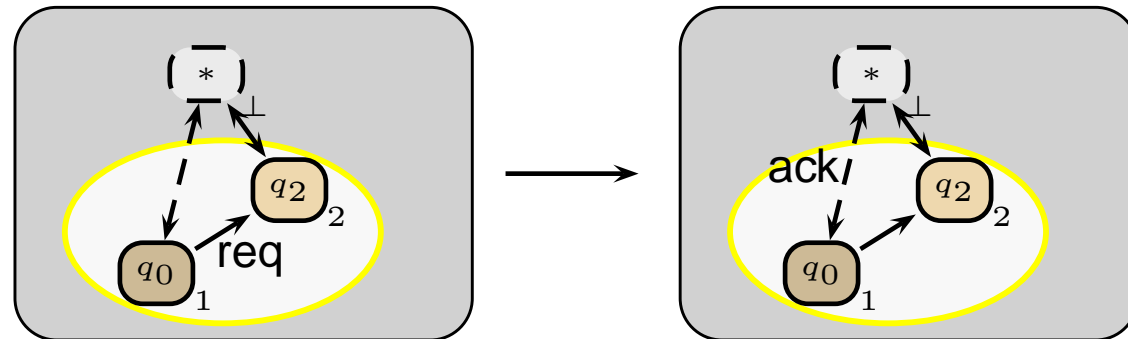
- Validation of abstract counterexample is done by increasing the size of the spotlight.

- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathcal{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket = 1/2$.
- Either the spotlight was too small, or the over-approximation of the "non-spotlight identities" produced **spurious behaviour**.



- Validation of abstract counterexample is done by increasing the size of the spotlight.
- Abstract counterexample which have been identified as being spurious yield assumptions on the non-spotlight part.

- Abstraction Refinement is necessary if $\alpha_{\mathcal{S}}(\mathcal{T}) \llbracket \phi(a_1, \dots, a_n) \rrbracket = 1/2$.
- Either the spotlight was too small, or the over-approximation of the "non-spotlight identities" produced **spurious behaviour**.



- Validation of abstract counterexample is done by increasing the size of the spotlight.
- Abstract counterexample which have been identified as being spurious yield assumptions on the non-spotlight part.
- This iteration of validation and refinement is an instance of the CEGAR approach.

Intro

Abstraction

Agents

Areas

End

- So far, the models and specifications purely address “topological properties”, e.g.

Intro

Abstraction

Agents

Areas

End

- So far, the models and specifications purely address “topological properties”, e.g.
 - the fact that cars recognise each other is modelled by non-deterministic inputs

Intro

Abstraction

Agents

Areas

End

- So far, the models and specifications purely address “topological properties”, e.g.
 - the fact that cars recognise each other is modelled by non-deterministic inputs
 - specifications only require that e.g. the interlinking within a platoon is acyclic.

Intro

Abstraction

Agents

Areas

End

- So far, the models and specifications purely address “topological properties”, e.g.
 - the fact that cars recognise each other is modelled by non-deterministic inputs
 - specifications only require that e.g. the interlinking within a platoon is acyclic.
- To fully represent the behaviour of e.g. automated highway systems, the physical position of agents becomes relevant.

Intro

Abstraction

Agents

Areas

End

- So far, the models and specifications purely address “topological properties”, e.g.
 - the fact that cars recognise each other is modelled by non-deterministic inputs
 - specifications only require that e.g. the interlinking within a platoon is acyclic.
- To fully represent the behaviour of e.g. automated highway systems, the physical position of agents becomes relevant.
- Let \mathcal{C} be a characterisation of some metric space (e.g. \mathbb{R}^2), we now assume that an interpretation \mathcal{I} in particular determines the actual position of an agent, denoted by $\text{pos}_{\mathcal{I}}(a) \in \mathcal{C}$.

- So far, the models and specifications purely address “topological properties”, e.g.
 - the fact that cars recognise each other is modelled by non-deterministic inputs
 - specifications only require that e.g. the interlinking within a platoon is acyclic.
- To fully represent the behaviour of e.g. automated highway systems, the physical position of agents becomes relevant.
- Let \mathcal{C} be a characterisation of some metric space (e.g. \mathbb{R}^2), we now assume that an interpretation \mathcal{I} in particular determines the actual position of an agent, denoted by $\text{pos}_{\mathcal{I}}(a) \in \mathcal{C}$.
- Approach: To analyse the behaviour of physically adjacent agents, let the content of the spotlight comprise agents within a certain area $\mathcal{R} \subseteq \mathcal{C}$ (rather than by using a *fixed* set of spotlight identities).

Intro

Abstraction

Agents

Areas

End

- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

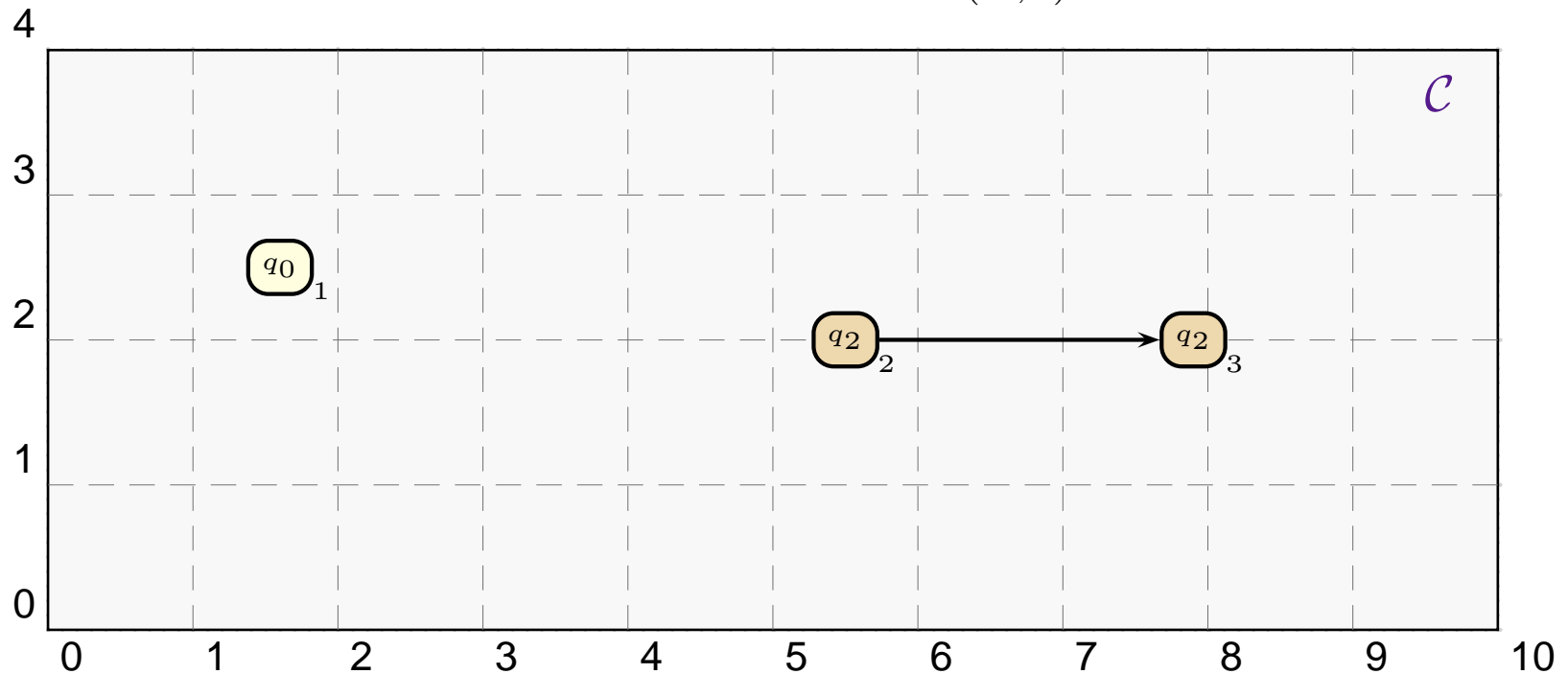
$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.

- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

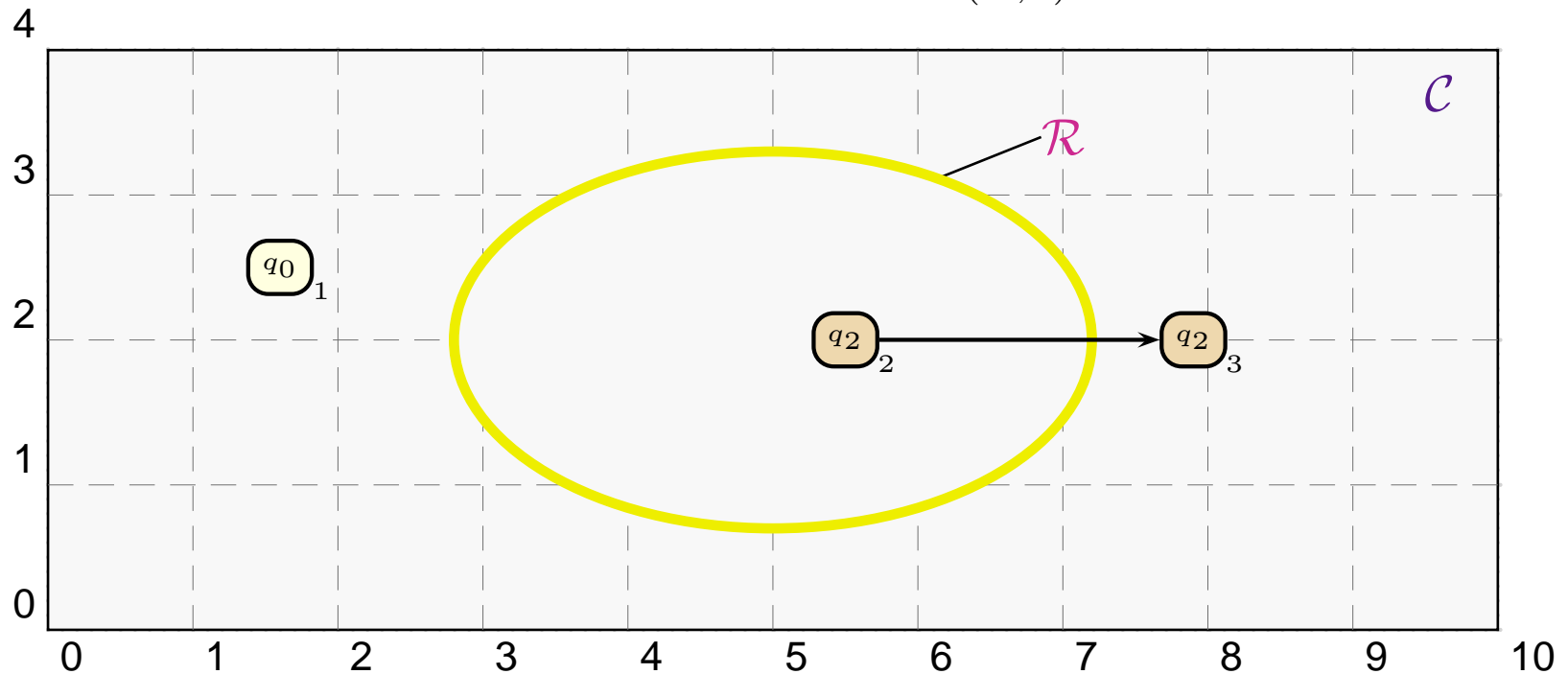
- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.



- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

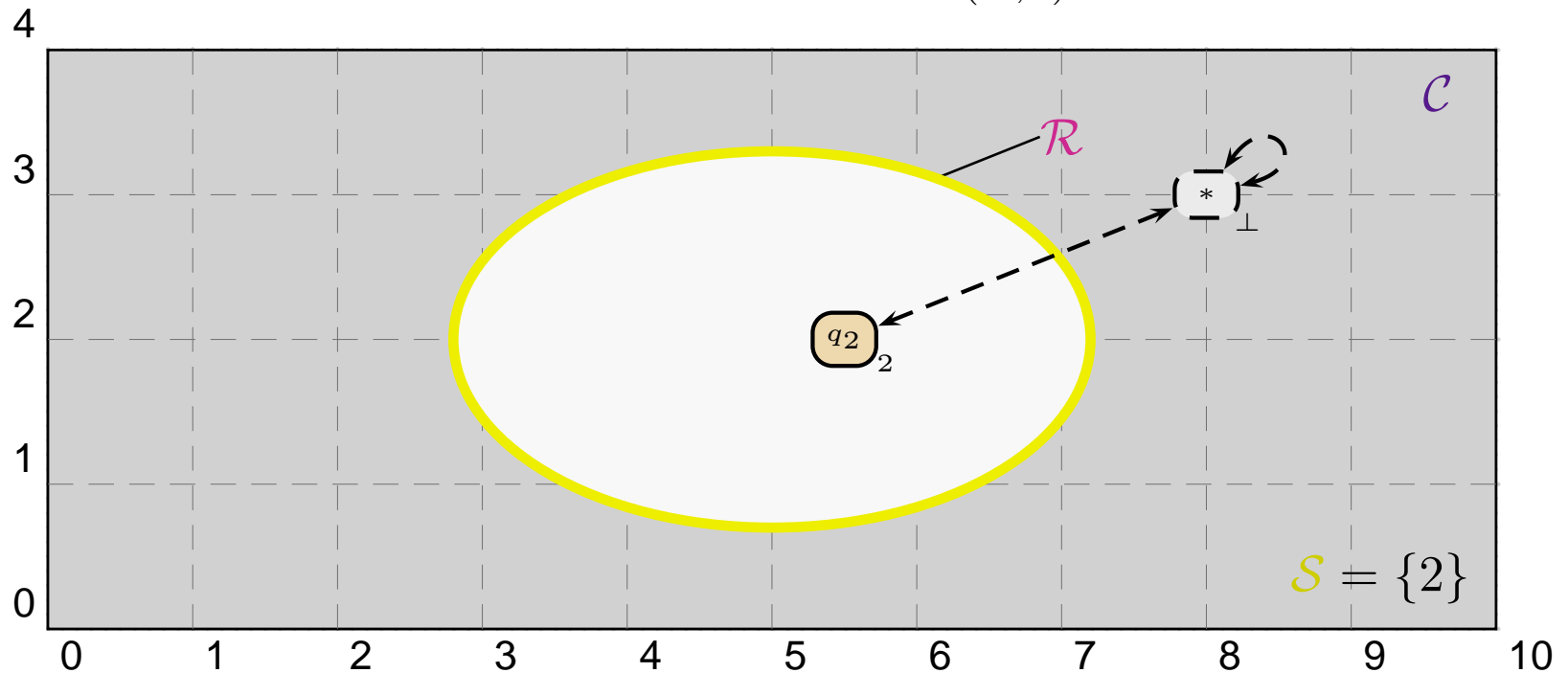
- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.



- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

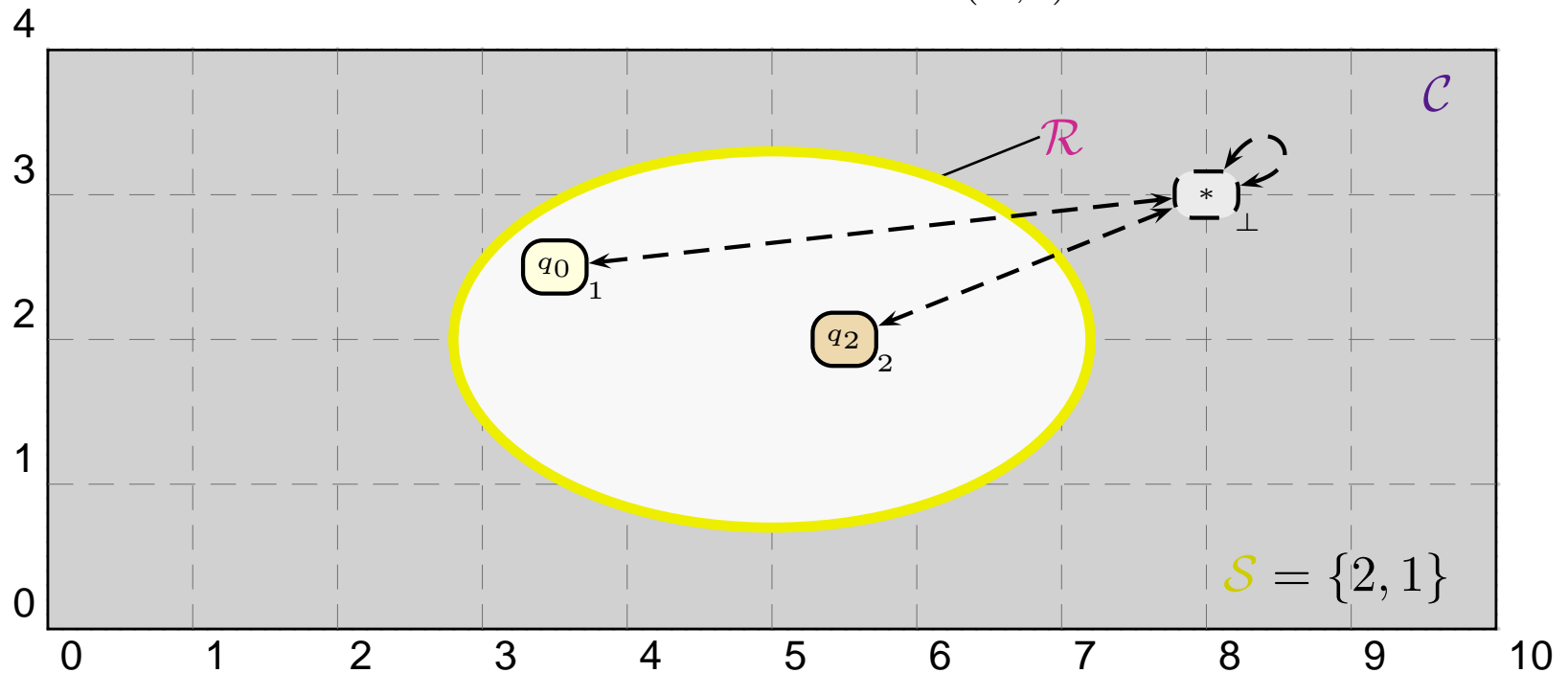
- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.



- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

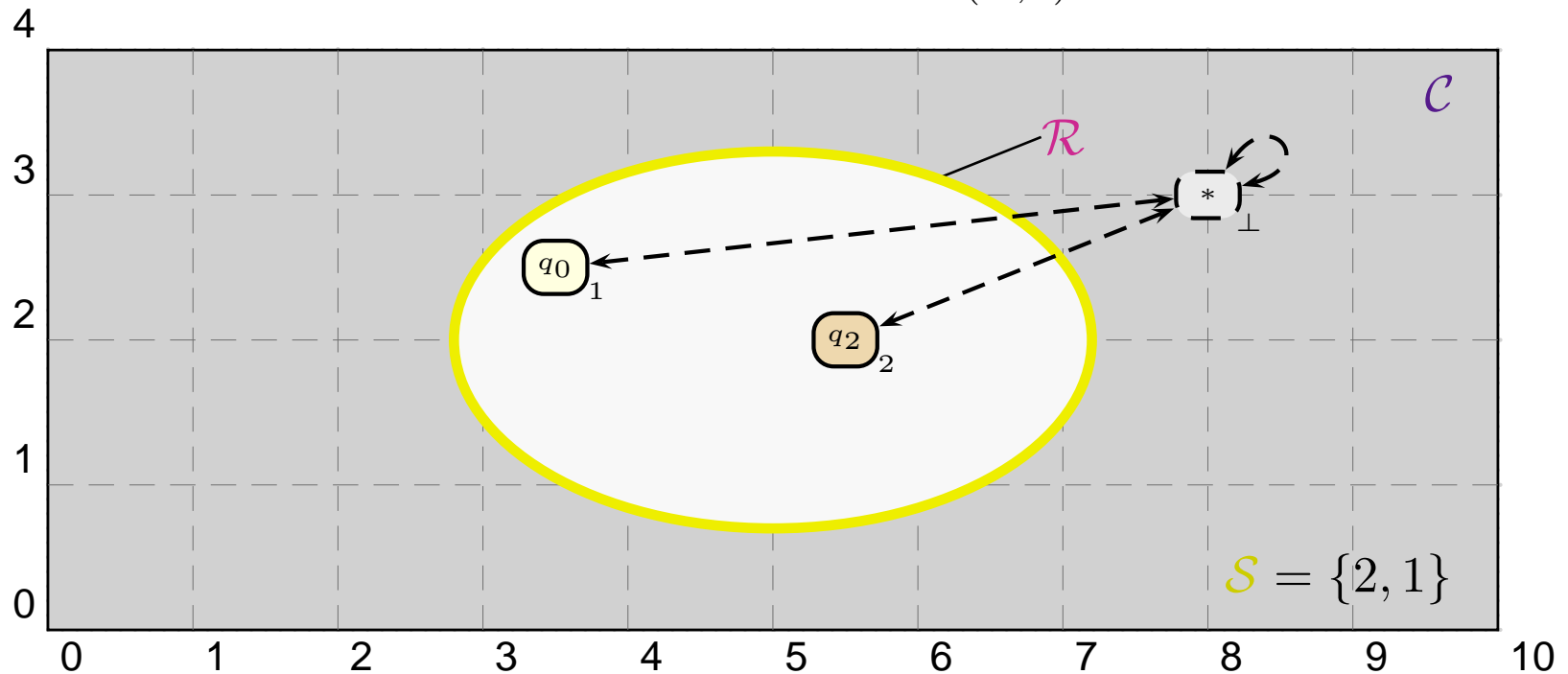
- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.



- Instead of fixing a spotlight $\mathcal{S} \subseteq \mathcal{A}$, we obtain a varying spotlight

$$\mathcal{S}(\mathcal{R}, \mathcal{I}) = \{a \in \mathcal{A} \mid \text{pos}_{\mathcal{I}}(a) \in \mathcal{R}\}$$

- The spotlight abstraction of a state $\alpha_{\mathcal{S}(\mathcal{R}, \mathcal{I})}(\mathcal{I})$ remains the same.



- In the abstract system, we now have two kinds of abstract impacts:
 1. message interference (as before), and
 2. materialisation (new!).

Intro

Abstraction

Agents

Areas

End

We have different possibilities to choose the area \mathcal{R} .

Intro

Abstraction

Agents

Areas

End

We have different possibilities to choose the area \mathcal{R} .

1. Focus on “**danger zones**”, e.g. junctions, highway exits
 - Establishes properties of the kind
“for all agents there is no collision in *this area*”

Intro

Abstraction

Agents

Areas

End

We have different possibilities to choose the area \mathcal{R} .

1. Focus on “**danger zones**”, e.g. junctions, highway exits
 - Establishes properties of the kind
“for all agents there is no collision in *this area*”
2. Exploit **symmetric agents on symmetric spaces**.
 - If the behaviour of the agents is *independent* of its identity, and the focussed areas are *representative* for the overall space, we may be able to establish
“for all agents there is no collision *anywhere*”

Intro

Abstraction

Agents

Areas

End

We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Intro

Abstraction

Agents

Areas

End

We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

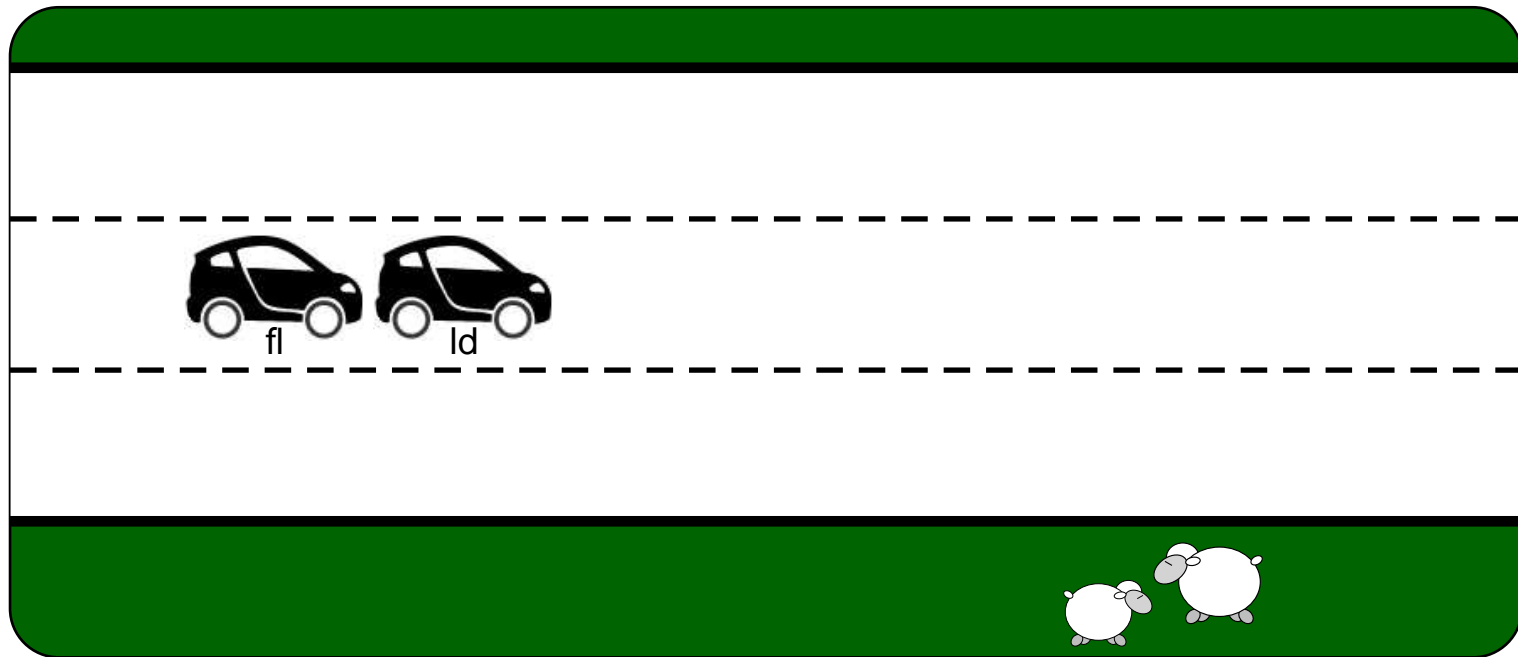
Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .

We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .

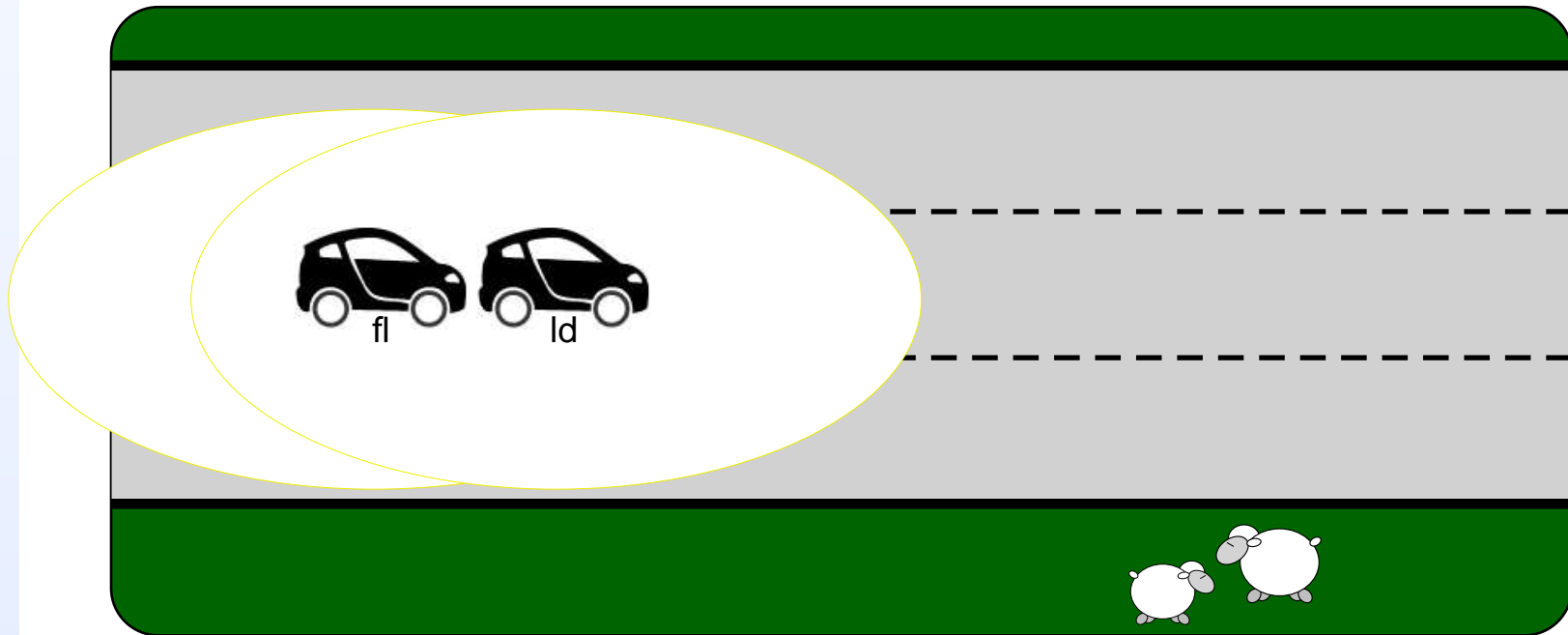


We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .

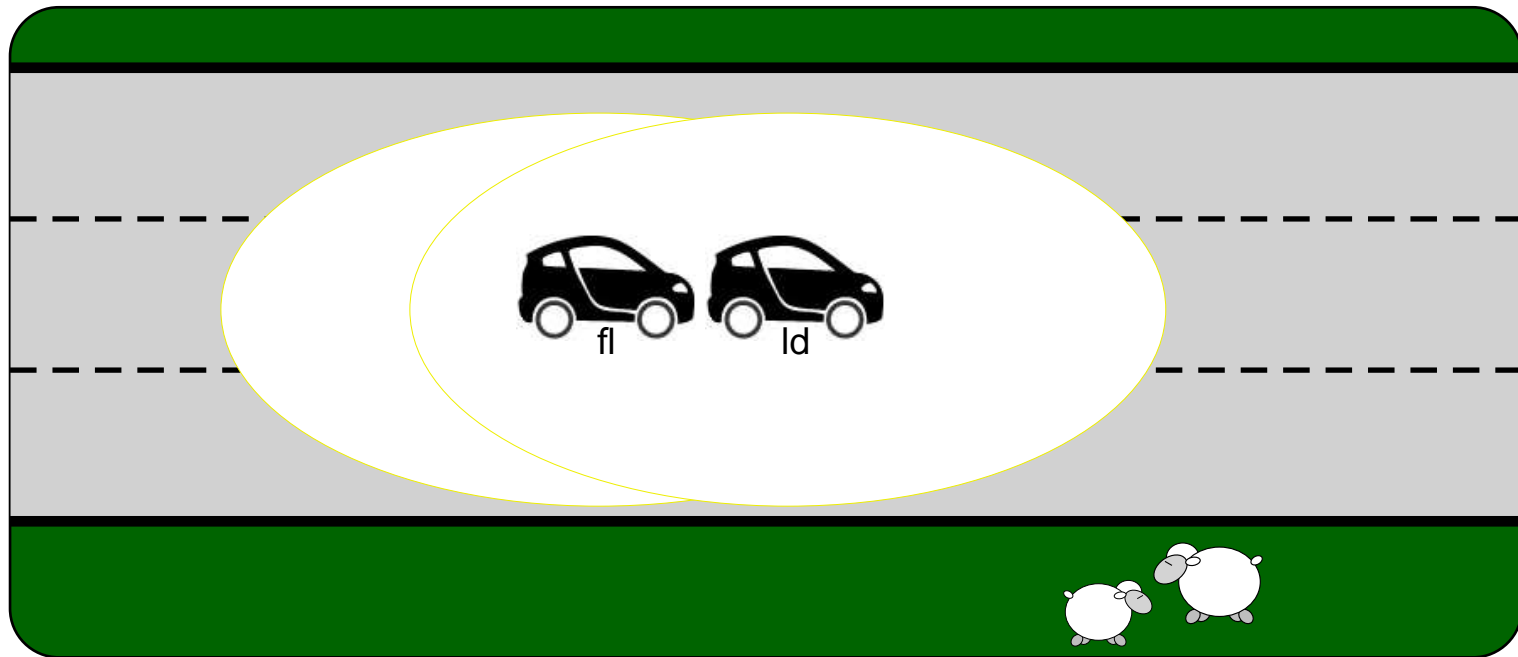


We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .

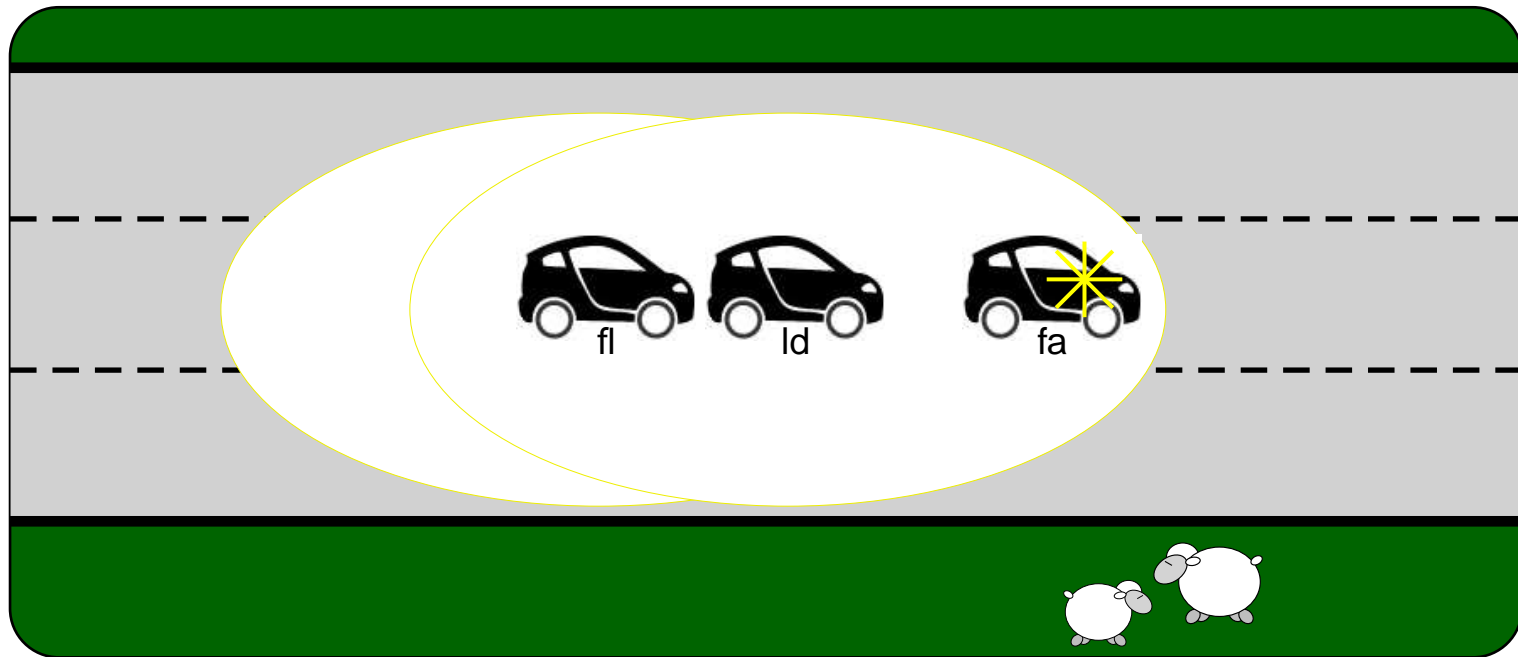


We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .

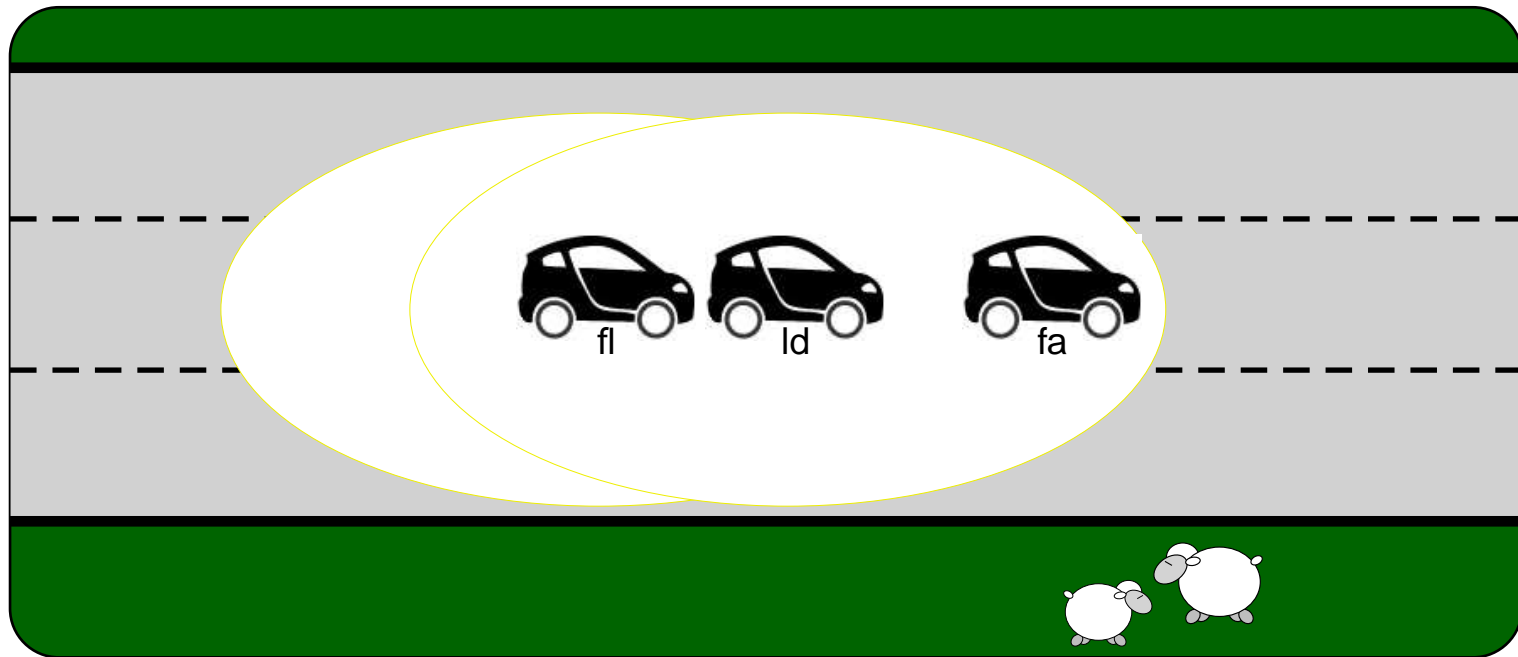


We have different possibilities to choose the area \mathcal{R} .

3. Move the area depending on the positions of $Ego \subseteq \mathcal{A}$:

$$\mathcal{R}(Ego, \mathcal{I}, r) = \{x \in \mathcal{C} \mid \exists a \in Ego : d(\text{pos}_{\mathcal{I}}(a), x) \leq r\}$$

Note that $Ego \subseteq \mathcal{S}(\mathcal{R}(Ego, \mathcal{I}, r), \mathcal{I})$ for any Ego, r and \mathcal{I} .



Intro

Abstraction

Agents

Areas

End

- Abstract impacts (message interference and materilisation) can be validated by increasing the set of ego agents *Ego* (which in turn increases the area, which in turn increases the spotlight).

Intro

Abstraction

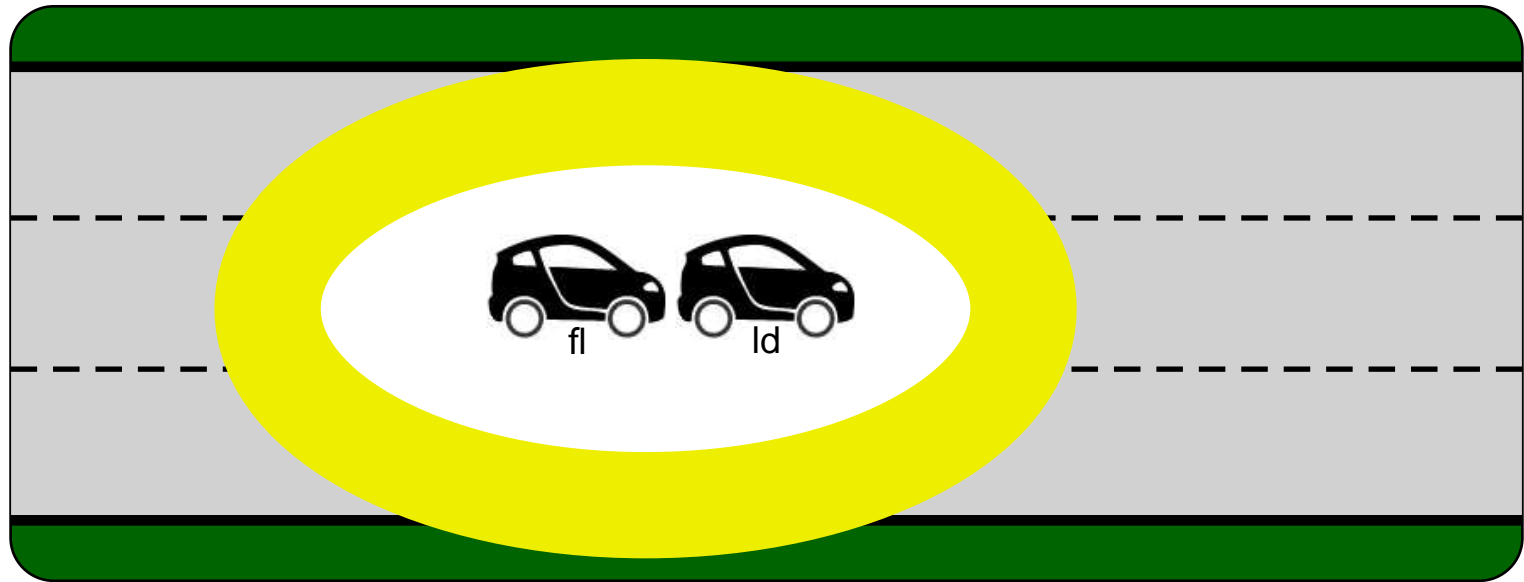
Agents

Areas

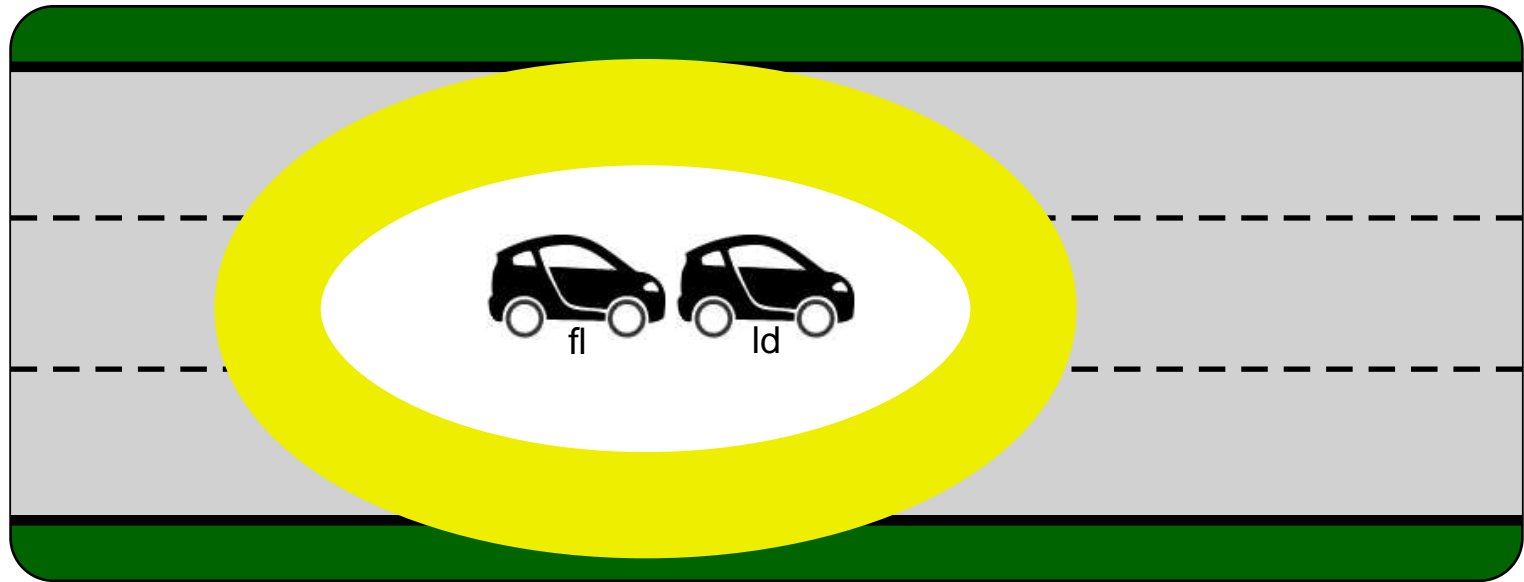
End

- Abstract impacts (message interference and materialisation) can be validated by increasing the set of ego agents *Ego* (which in turn increases the area, which in turn increases the spotlight).
- Materialisation can be refined by constraints from the dynamics of the underlying model.

- Abstract impacts (message interference and materilisation) can be validated by increasing the set of ego agents *Ego* (which in turn increases the area, which in turn increases the spotlight).
- Materialisation can be refined by constraints from the dynamics of the underlying model.



- Abstract impacts (message interference and materilisation) can be validated by increasing the set of ego agents *Ego* (which in turn increases the area, which in turn increases the spotlight).
- Materialisation can be refined by constraints from the dynamics of the underlying model.



- In addition to the CEGAR refinement, spurious message interferences can be suppressed by choosing a sufficiently large radius r .

Intro

Abstraction

Agents

Areas

End

- The presented variant of spotlight abstraction goes back to the “Data-Type Reduction” technique of McMillan.

Intro

Abstraction

Agents

Areas

End

- The presented variant of spotlight abstraction goes back to the “Data-Type Reduction” technique of McMillan.
- Usage of DTR for the reduction of unbounded processes to finite instances.

Intro

Abstraction

Agents

Areas

End

- The presented variant of spotlight abstraction goes back to the “Data-Type Reduction” technique of McMillan.
- Usage of DTR for the reduction of unbounded processes to finite instances.
- The “spotlight principle” is also applied for other abstraction techniques for unbounded systems of systems.

Intro

Abstraction

Agents

Areas

End

- The presented variant of spotlight abstraction goes back to the “Data-Type Reduction” technique of McMillan.
- Usage of DTR for the reduction of unbounded processes to finite instances.
- The “spotlight principle” is also applied for other abstraction techniques for unbounded systems of systems.
- The treatment of unbounded space by DTR looks promising, in particular for obtaining new kinds of refinement.

- The presented variant of spotlight abstraction goes back to the “Data-Type Reduction” technique of McMillan.
- Usage of DTR for the reduction of unbounded processes to finite instances.
- The “spotlight principle” is also applied for other abstraction techniques for unbounded systems of systems.
- The treatment of unbounded space by DTR looks promising, in particular for obtaining new kinds of refinement.

End.