

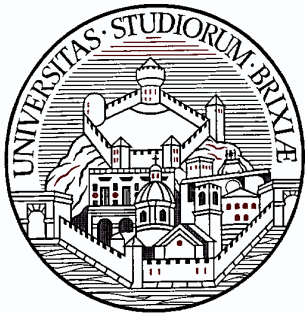
# Elements of distributed defeat status computation

---

**Massimiliano Giacomin**

giacomin@ing.unibs.it

<http://bsing.ing.unibs.it/~giacomin/>



DEA - Dipartimento di Elettronica per l'Automazione  
Università degli Studi di Brescia (Italy)

ABSTRACT CONCEPTS IN ARGUMENTATION  
Bahia Blanca, Argentina, 2008

# Outline of the talk

---

- Why distributed defeat status computation may be interesting?
- How much difficult is it?
- Two solutions (under some restrictive conditions)

# Outline of the talk

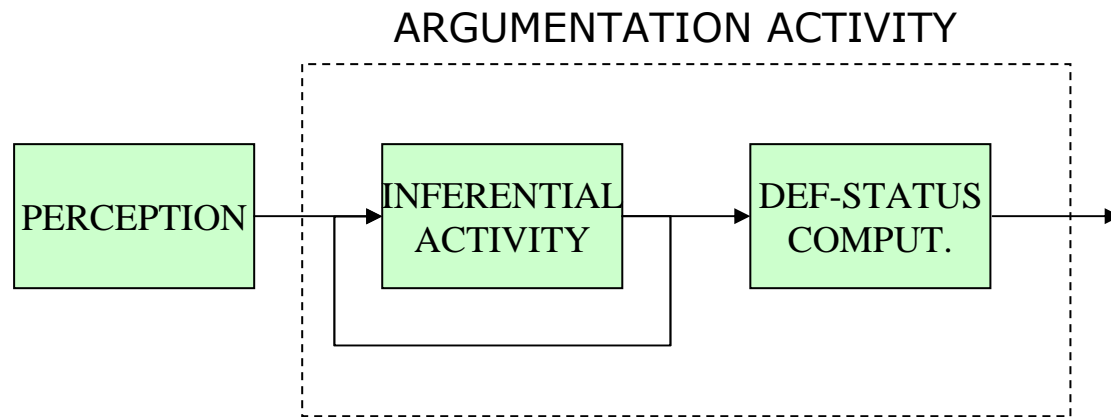
---

- Why distributed defeat status computation may be interesting?
  - different architectural choices in multi agent systems
  - motivating examples
  - a computational model for distributed argumentation and the role of self-stabilization
- How much difficult is it?
- Two solutions (under some restrictive conditions)

# Centralized vs. distributed argumentation

---

## Centralized argumentation



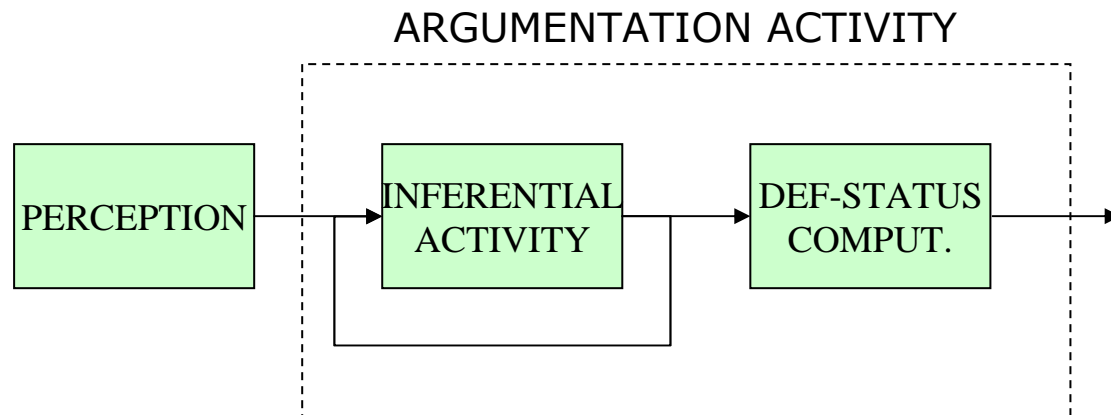
- Argumentation:
  - Inferential activity (argument construction)
  - Defeat status computation
- Centralized + Sequential

## Non centralized argumentation

- PARALLEL ARGUMENTATION: to achieve speed, reliability, ...
- DISTRIBUTED ARGUMENTATION: to apply argumentation in domains where knowledge is inherently distributed, typically in MAS where:
  - agents discuss (and argue) on opinions about the world
  - agents discuss (and argue) about actions, goals, resources...
  - ...

# Centralized vs. distributed argumentation

## Centralized argumentation



- Argumentation:
  - Inferential activity (argument construction)
  - Defeat status computation
- Centralized + Sequential

## Non centralized argumentation

- ~~• PARALLEL ARGUMENTATION: to achieve speed, reliability, ...~~
- DISTRIBUTED ARGUMENTATION: to apply argumentation in domains where knowledge is inherently distributed, typically in MAS where:
  - agents discuss (and argue) on opinions about the world
  - agents discuss (and argue) about actions, goals, resources...
  - ...

# Alternative architectures

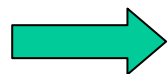
---

## Agents activities

- Argumentation activity
  - produce arguments (inferential activity)
  - compute the defeat status
- Information exchange

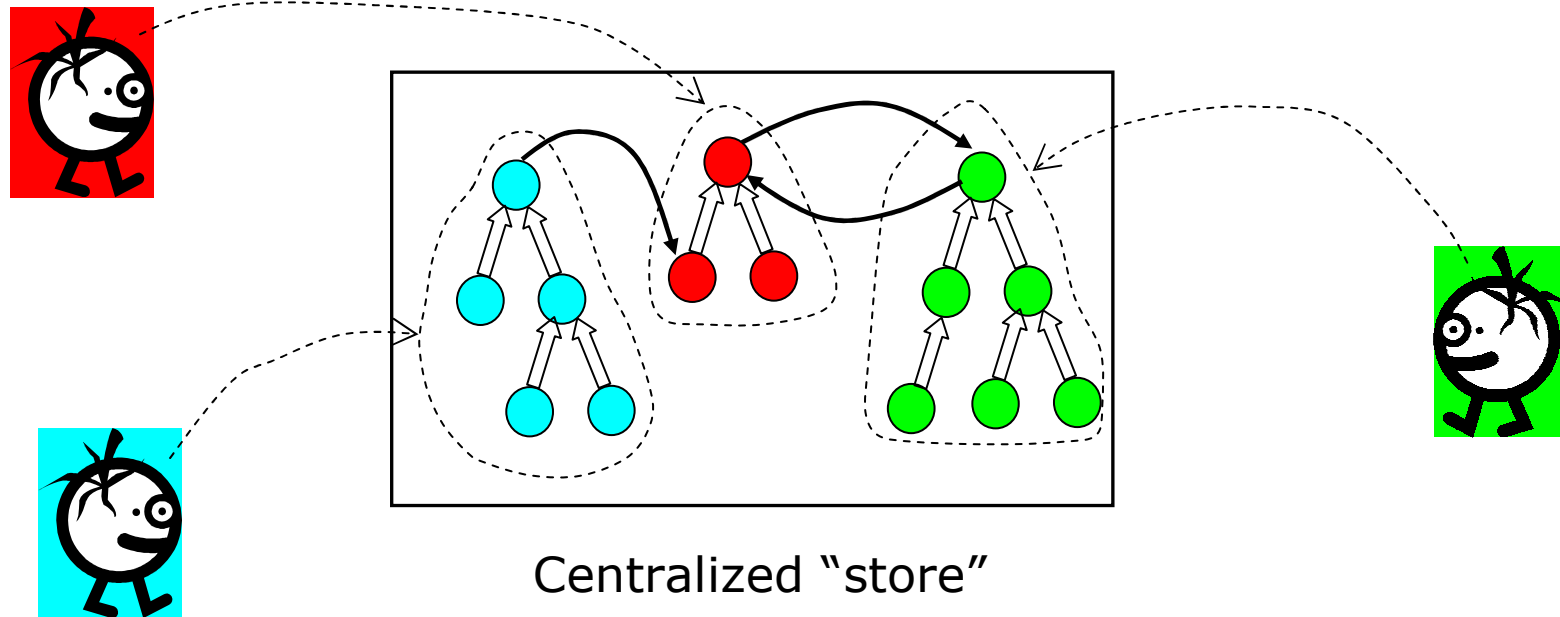
## Information exchange

- managed by a centralized entity  
vs. on a peer-to-peer basis
- always involving entire arguments  
vs. selected information about arguments,  
e.g. conclusions or attack relationships



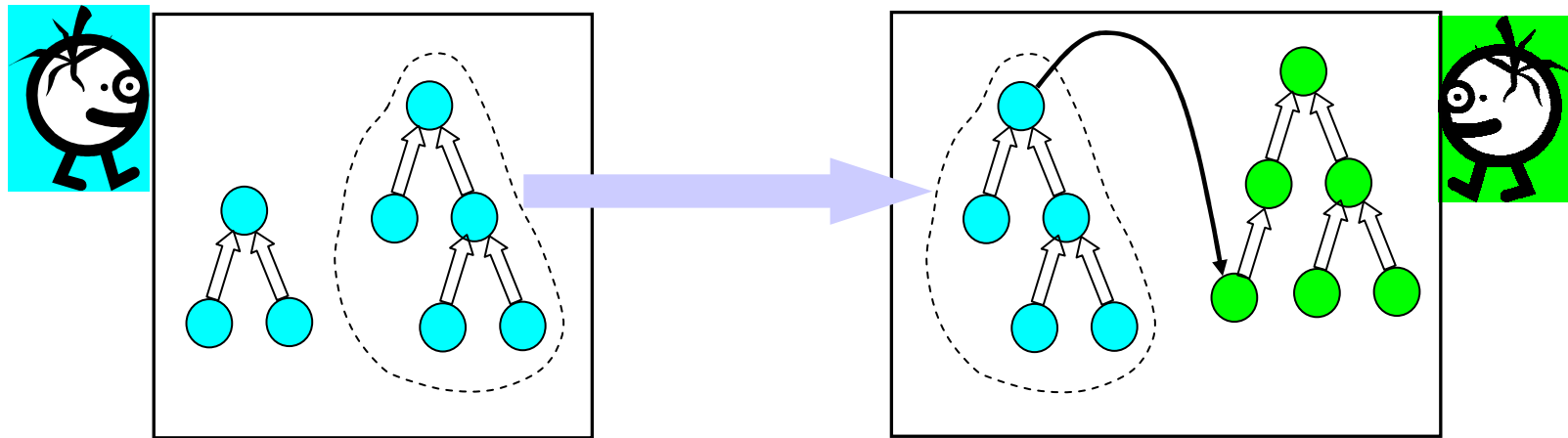
Three possible architectural models can be identified, influencing the way defeat status computation is carried out

# Exploiting a centralized structure



- Agents interact through a centralized "store" collecting all exchanged arguments
  - Agents contribute with entire arguments  
(see e.g. the notion of Risk Agora [McBurney & Parson '01])
- ➔ Defeat status computation carried out in a centralized way (algorithm running on the arguments' graph in the store)

# Peer-to-peer exchange of entire arguments



- Agents interact on a peer to peer basis
- Agents provide justification for their own claims, i.e. they exchange entire arguments

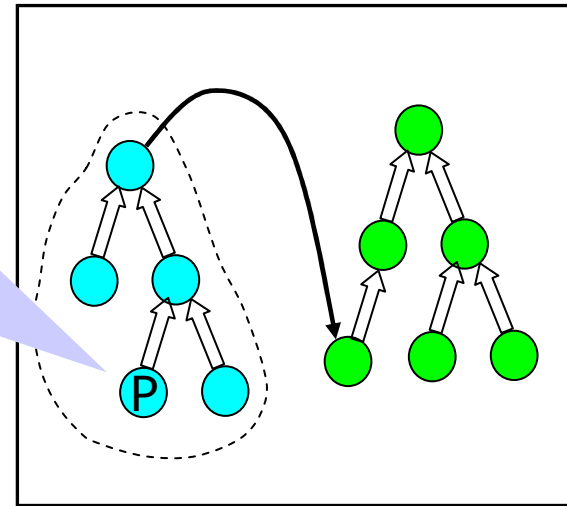
This is the typical model considered in MAS (see e.g. Parsons et al '98)

➔ Each agent carries out its own defeat status computation, e.g. recomputing it each time an argument is received: centralized defeat status computation is replicated within each agent.

# A fully distributed model



**P** is true



- Agents interact on a peer to peer basis
- Agents exchange not necessarily complete arguments but also just conclusions or attack relationships.

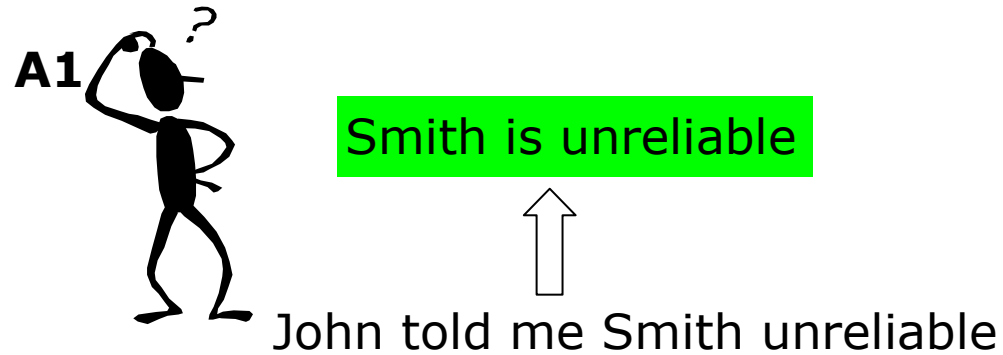
Why? Self-interest, privacy reasons, different competence domains, acquiring justifications may be impractical, ...

 Defeat status computation is now distributed...

# The Problem of Stabilization: an Example

---

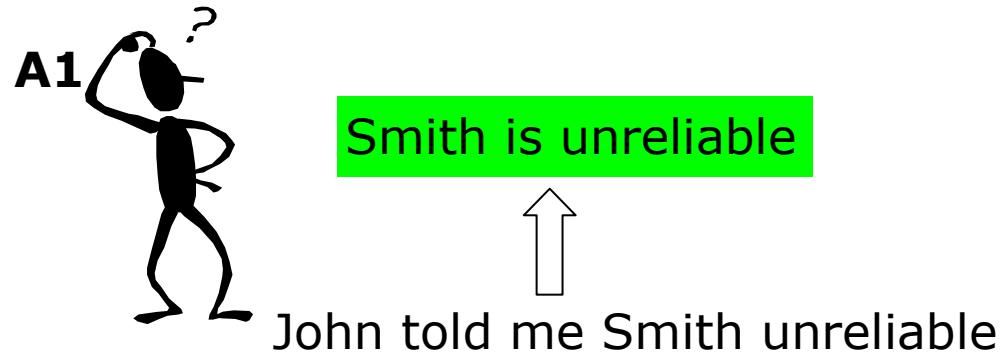
Interest about Smith reliability.



# The Problem of Stabilization: an Example

---

Interest about Smith reliability. Is John reliable?

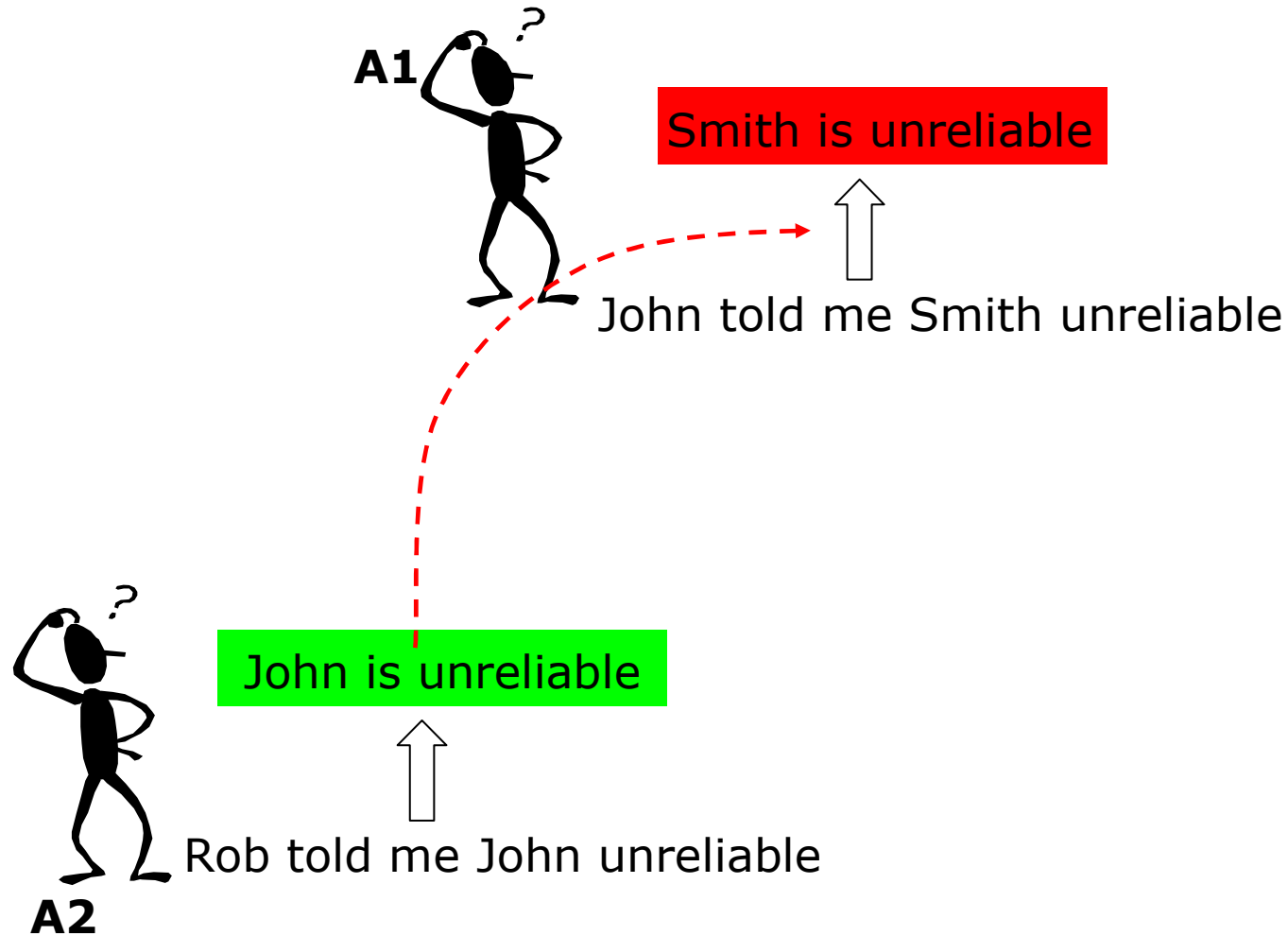


OK, BUT... WHAT ABOUT JOHN?

# The Problem of Stabilization: an Example

---

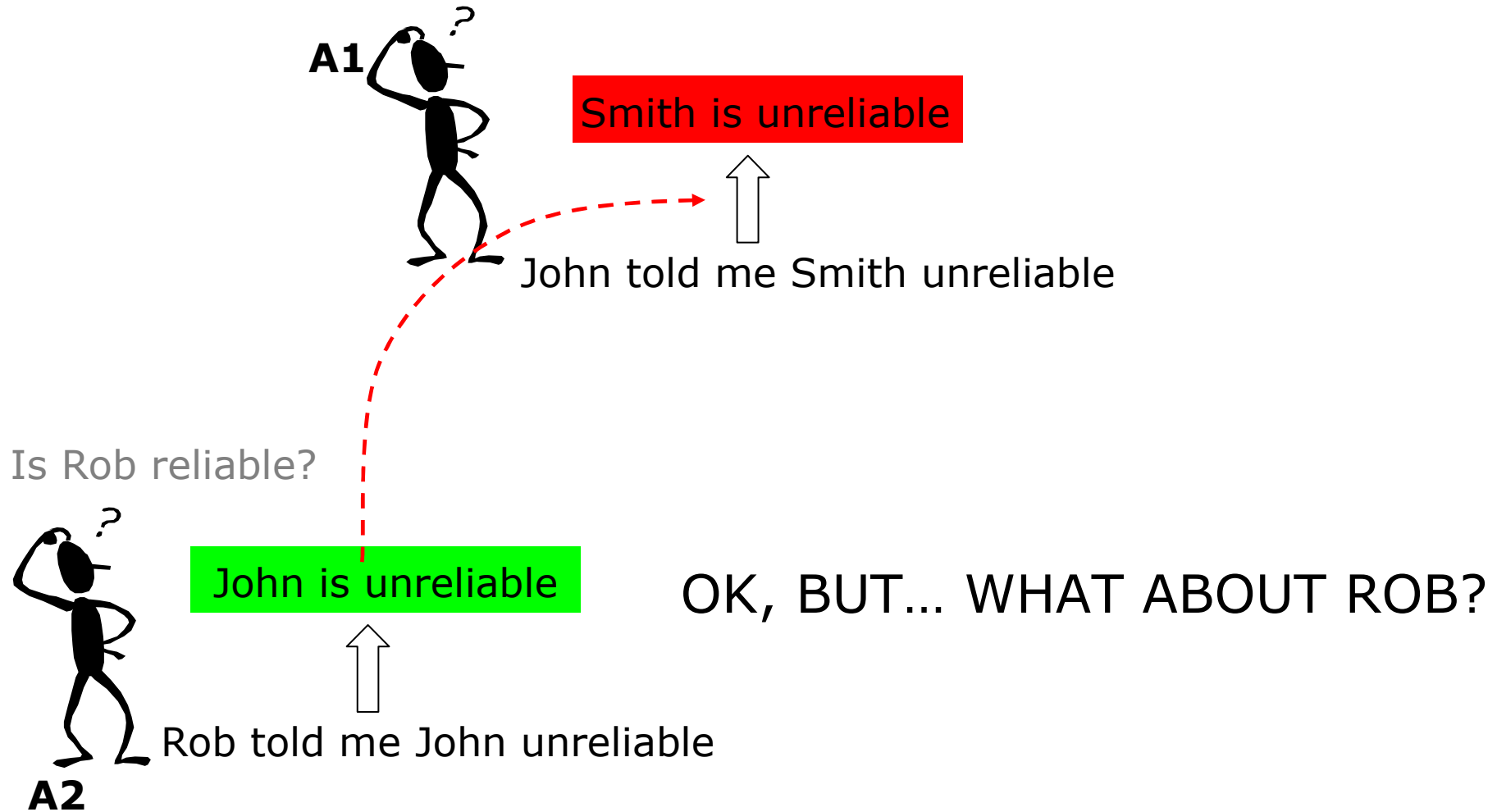
Interest about Smith reliability. Is John reliable?



# The Problem of Stabilization: an Example

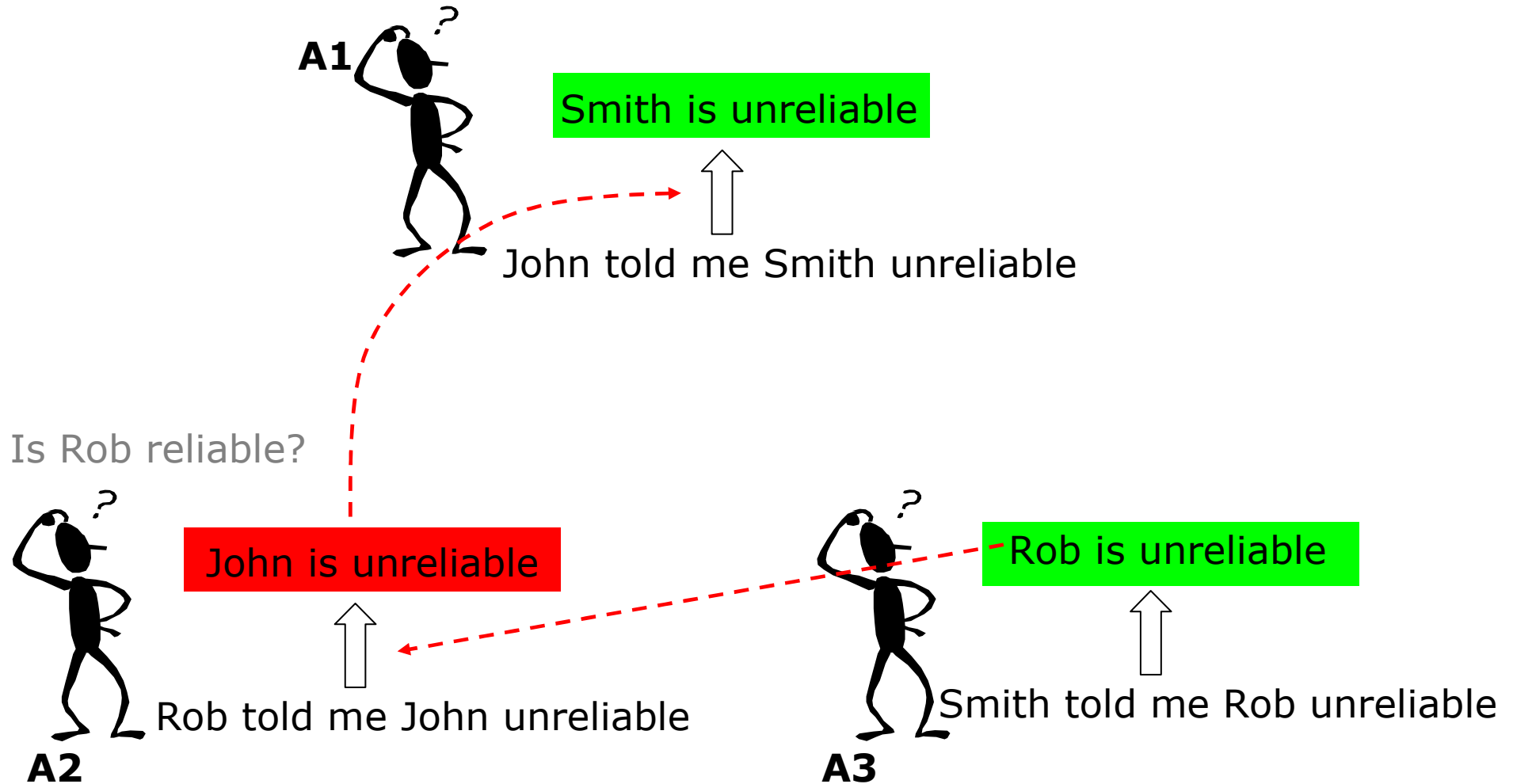
---

Interest about Smith reliability. Is John reliable?



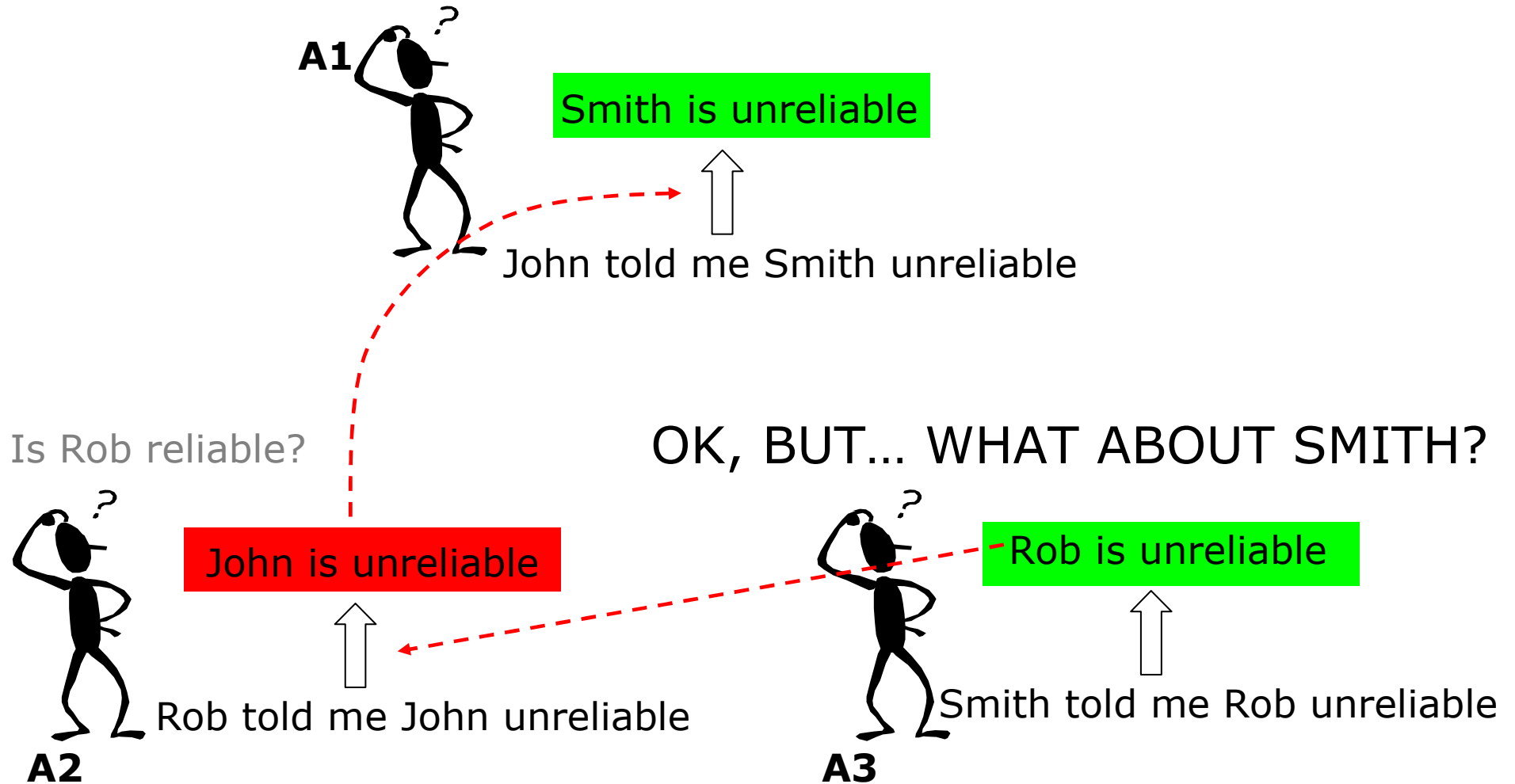
# The Problem of Stabilization: an Example

Interest about Smith reliability. Is John reliable?



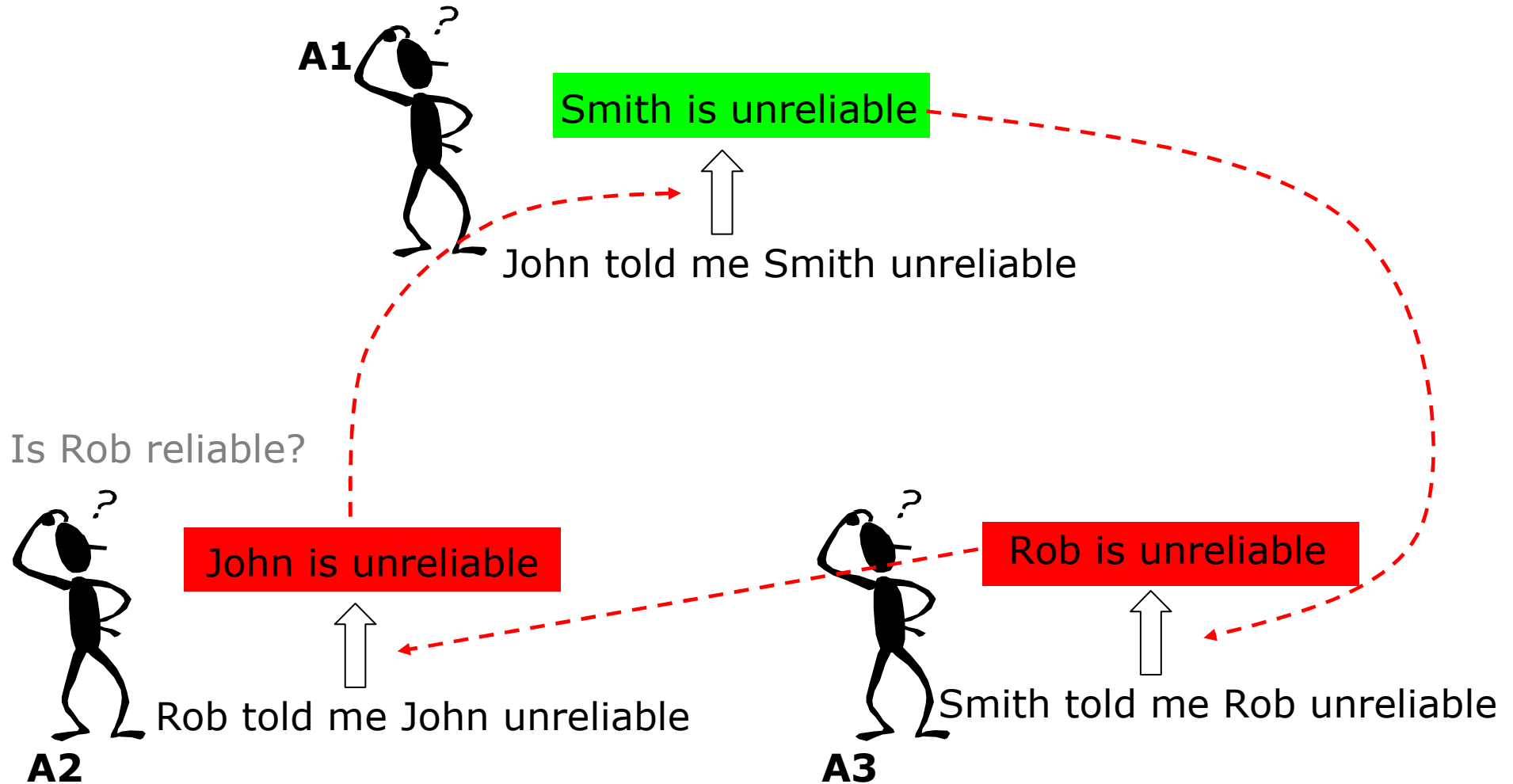
# The Problem of Stabilization: an Example

Interest about Smith reliability. Is John reliable?



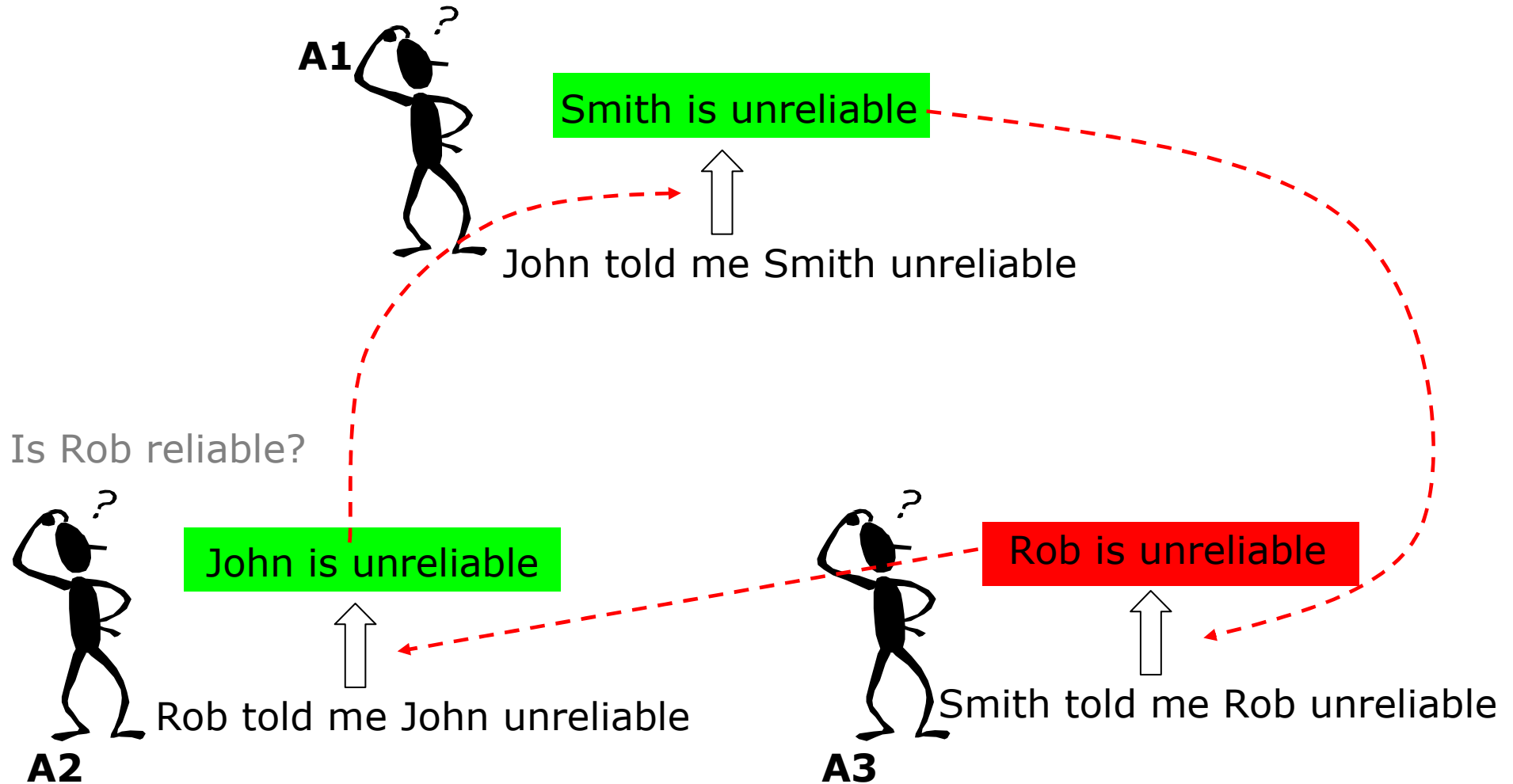
# The Problem of Stabilization: an Example

Interest about Smith reliability. Is John reliable?



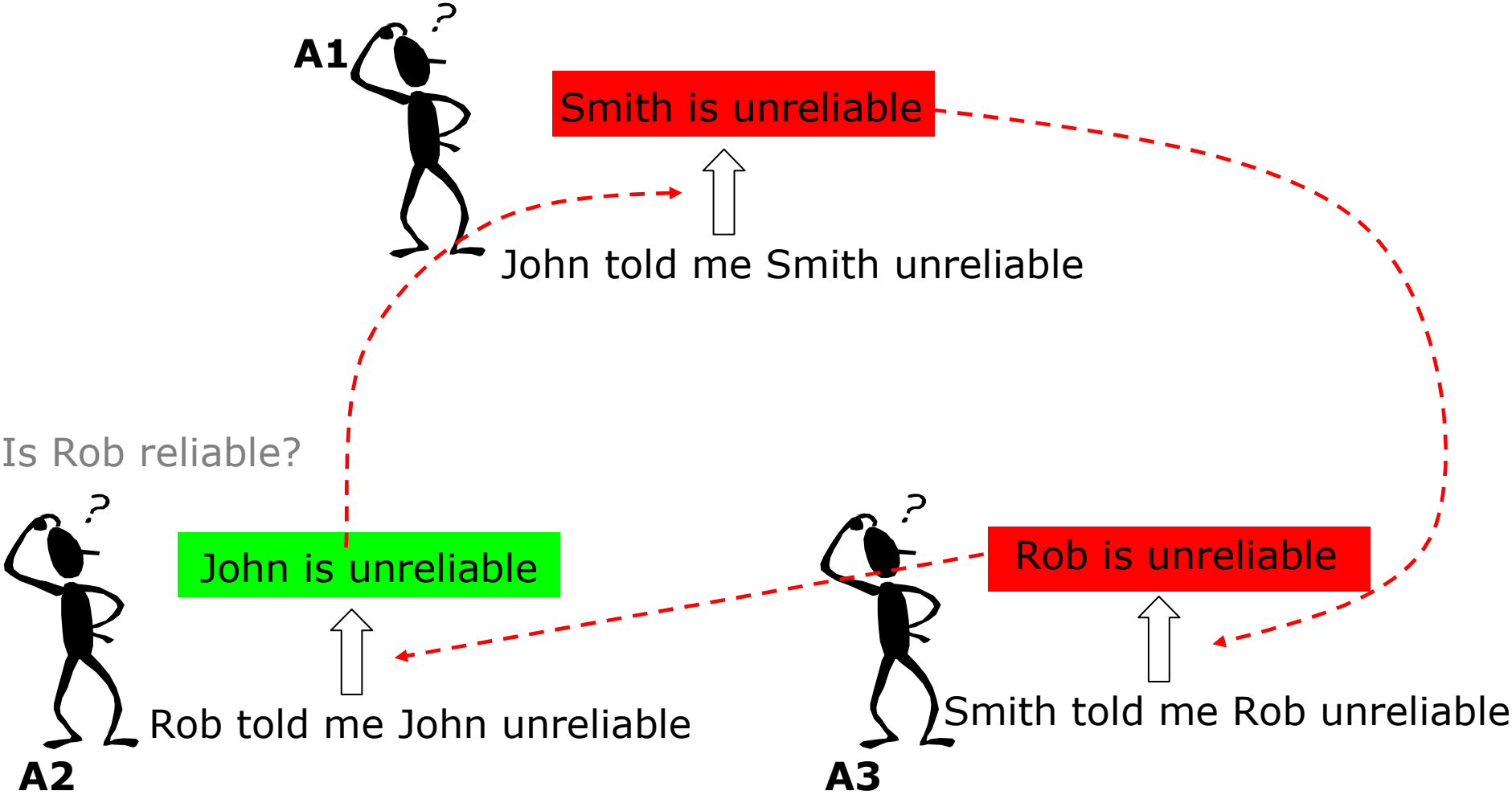
# The Problem of Stabilization: an Example

Interest about Smith reliability. Is John reliable?



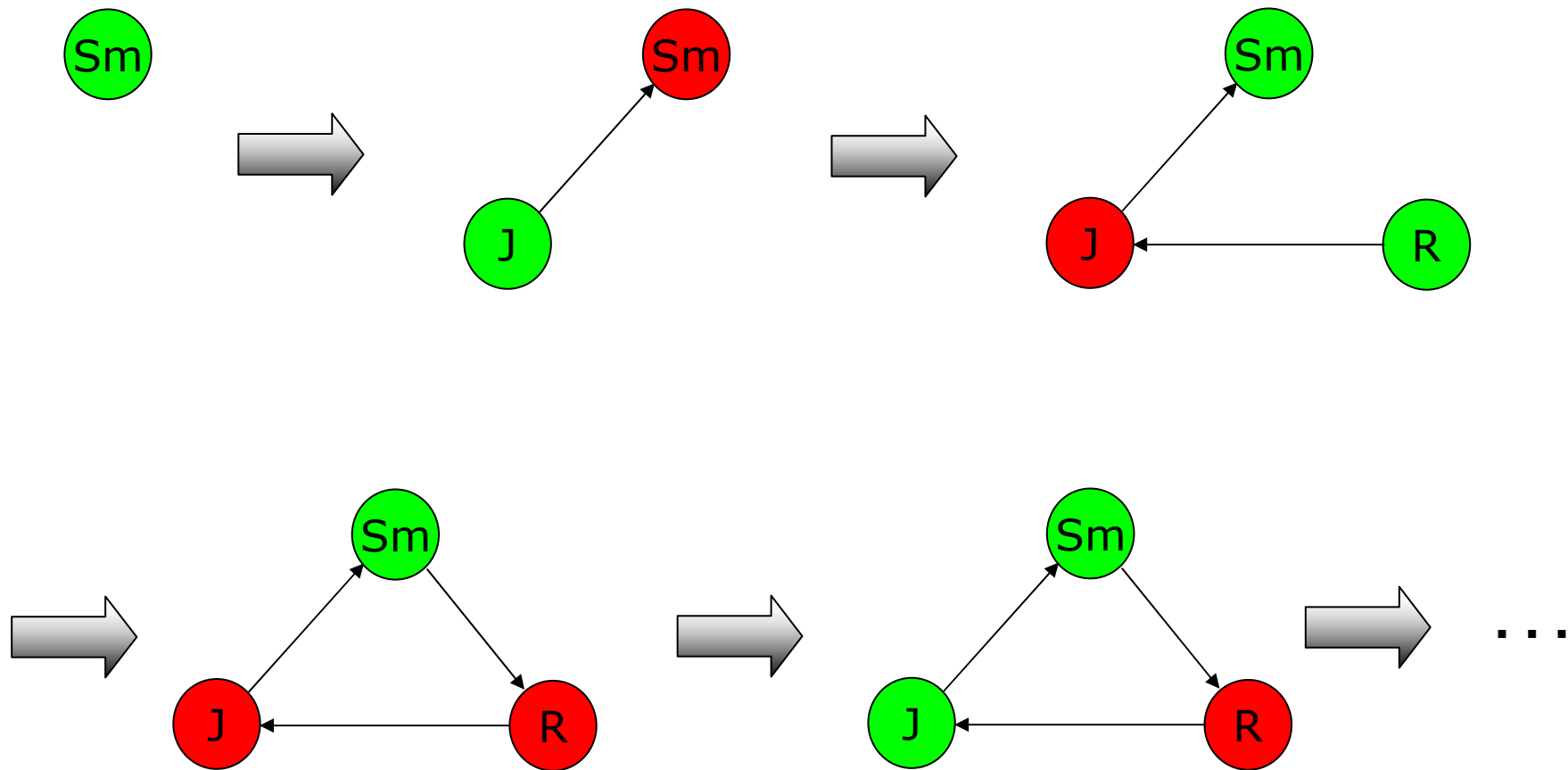
# The Problem of Stabilization: an Example

Interest about Smith reliability. Is John reliable?



## The Problem of Stabilization: an Example (2)

---



# The Problem of Stabilization: a Food Example

---

A<sub>1</sub>

bread

pasta

GC1(pasta, mushrooms)

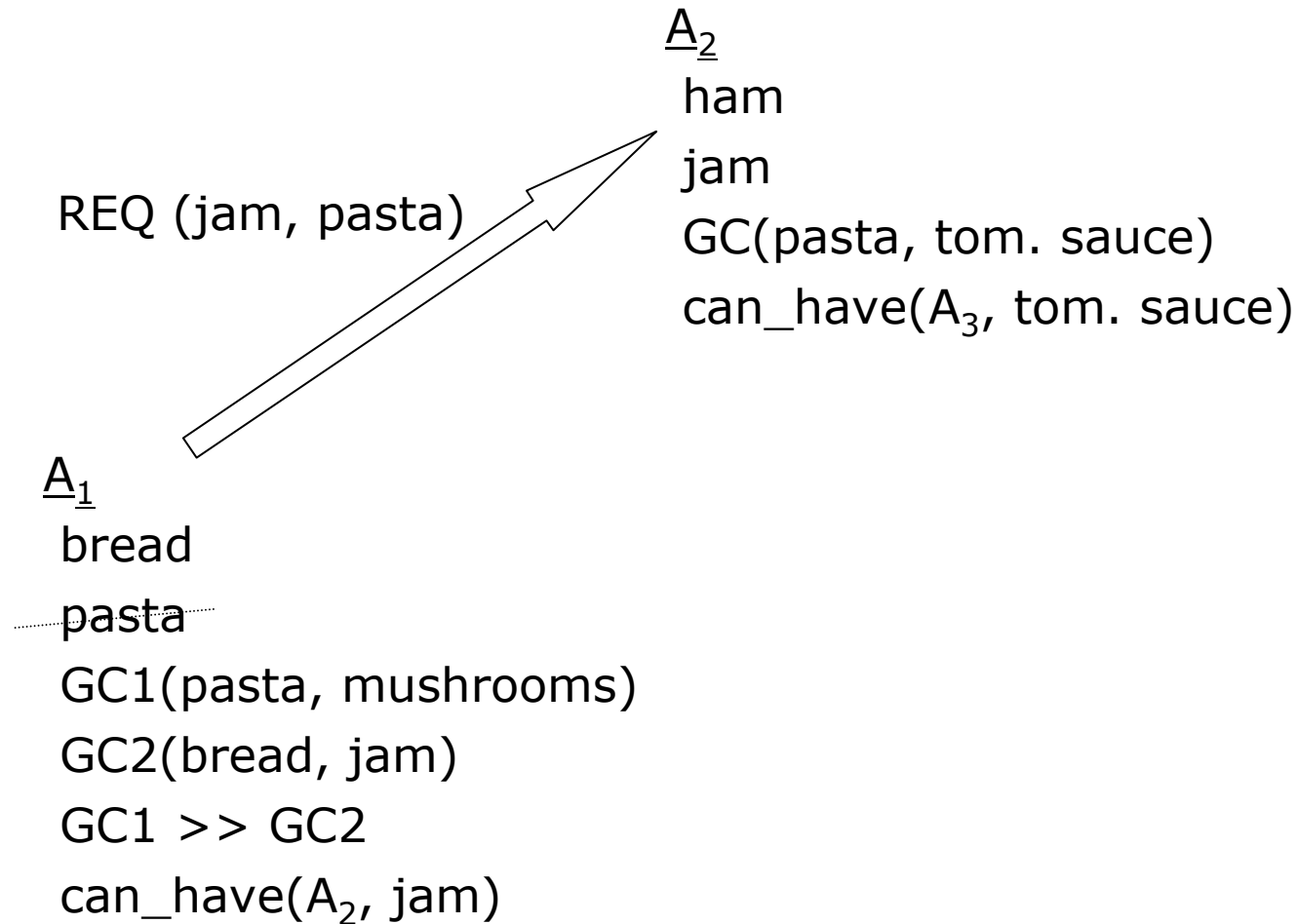
GC2(bread, jam)

GC1 >> GC2

can\_have(A<sub>2</sub>, jam)

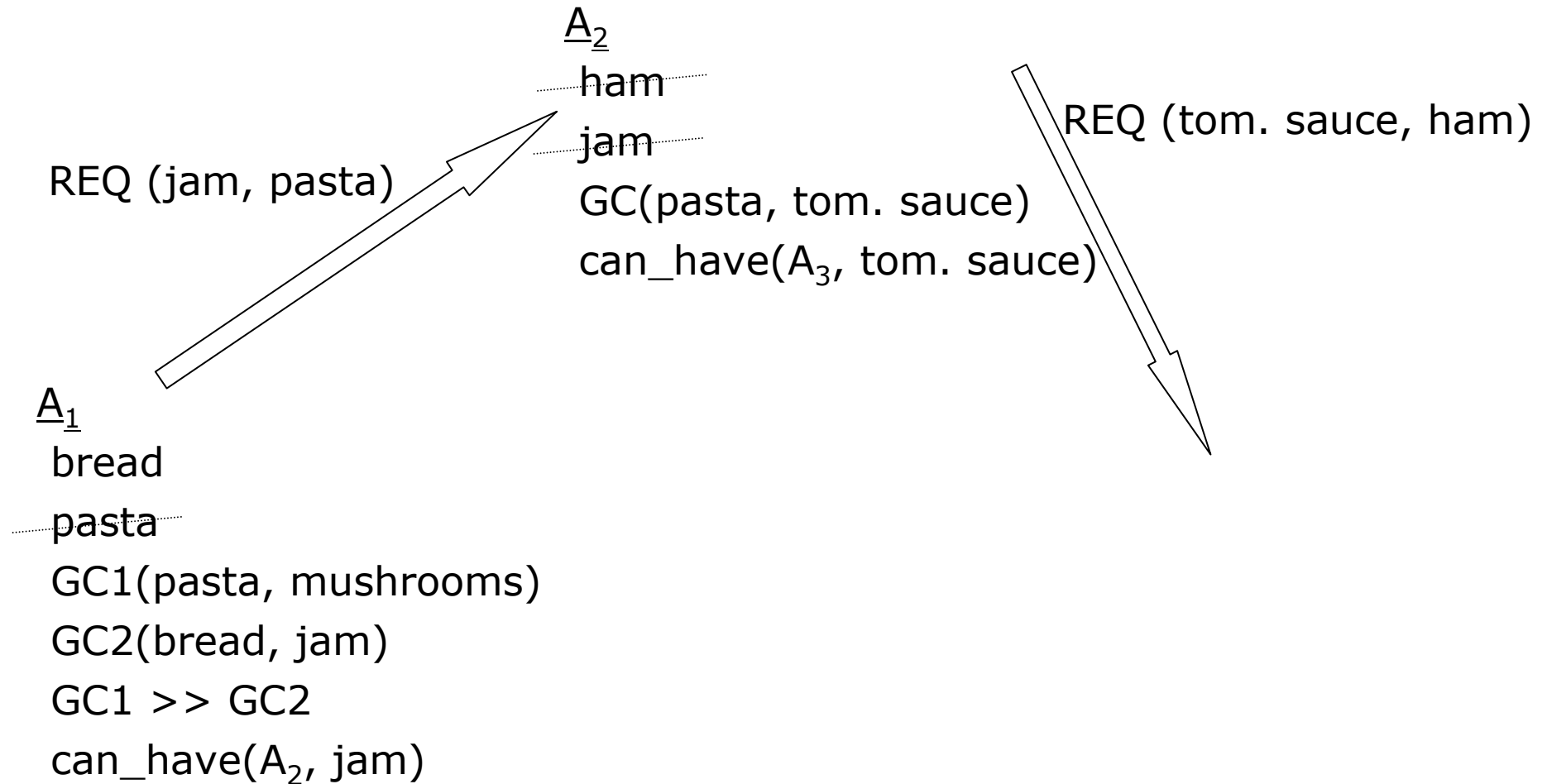
# The Problem of Stabilization: a Food Example

---



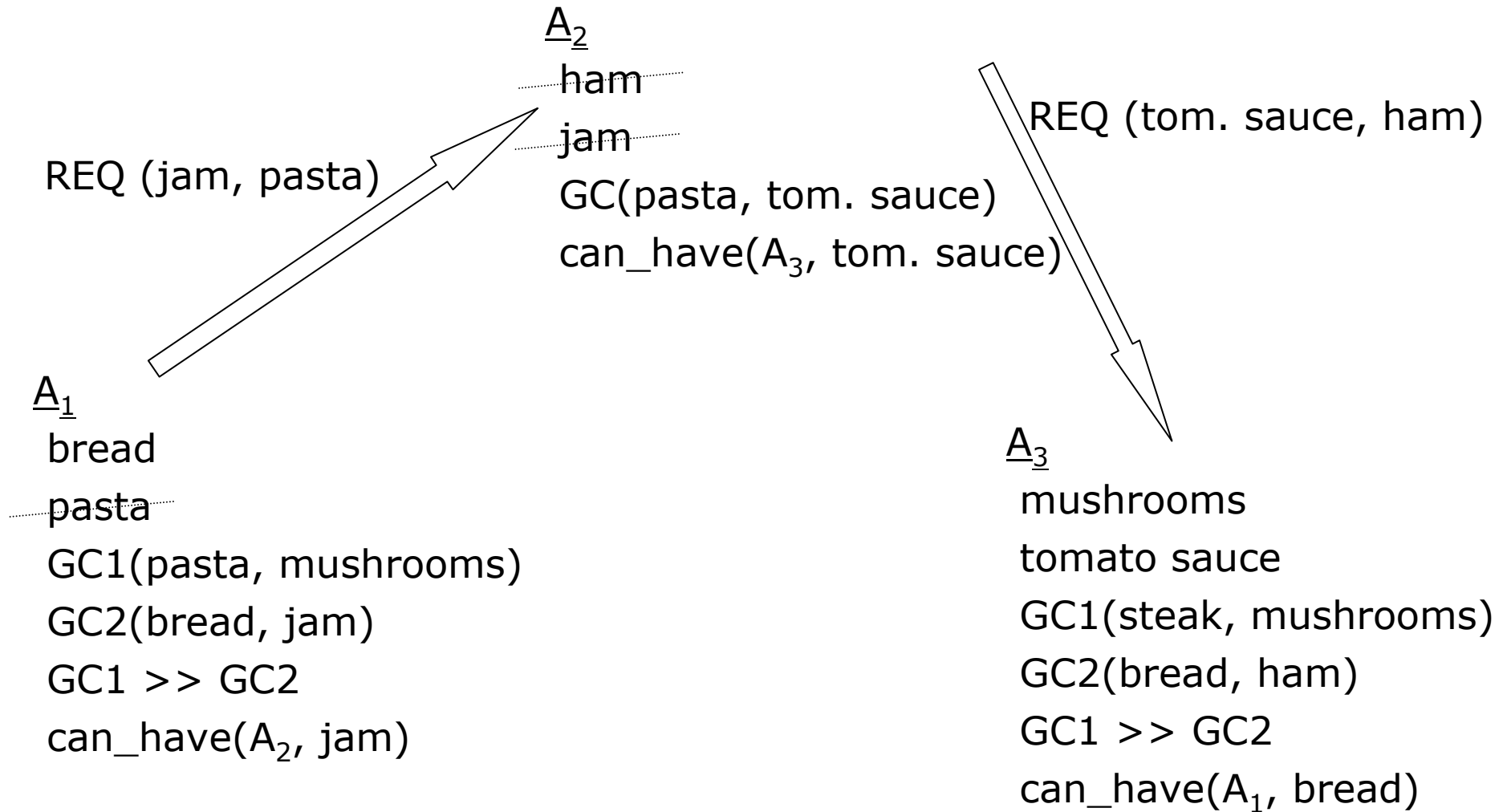
# The Problem of Stabilization: a Food Example

---



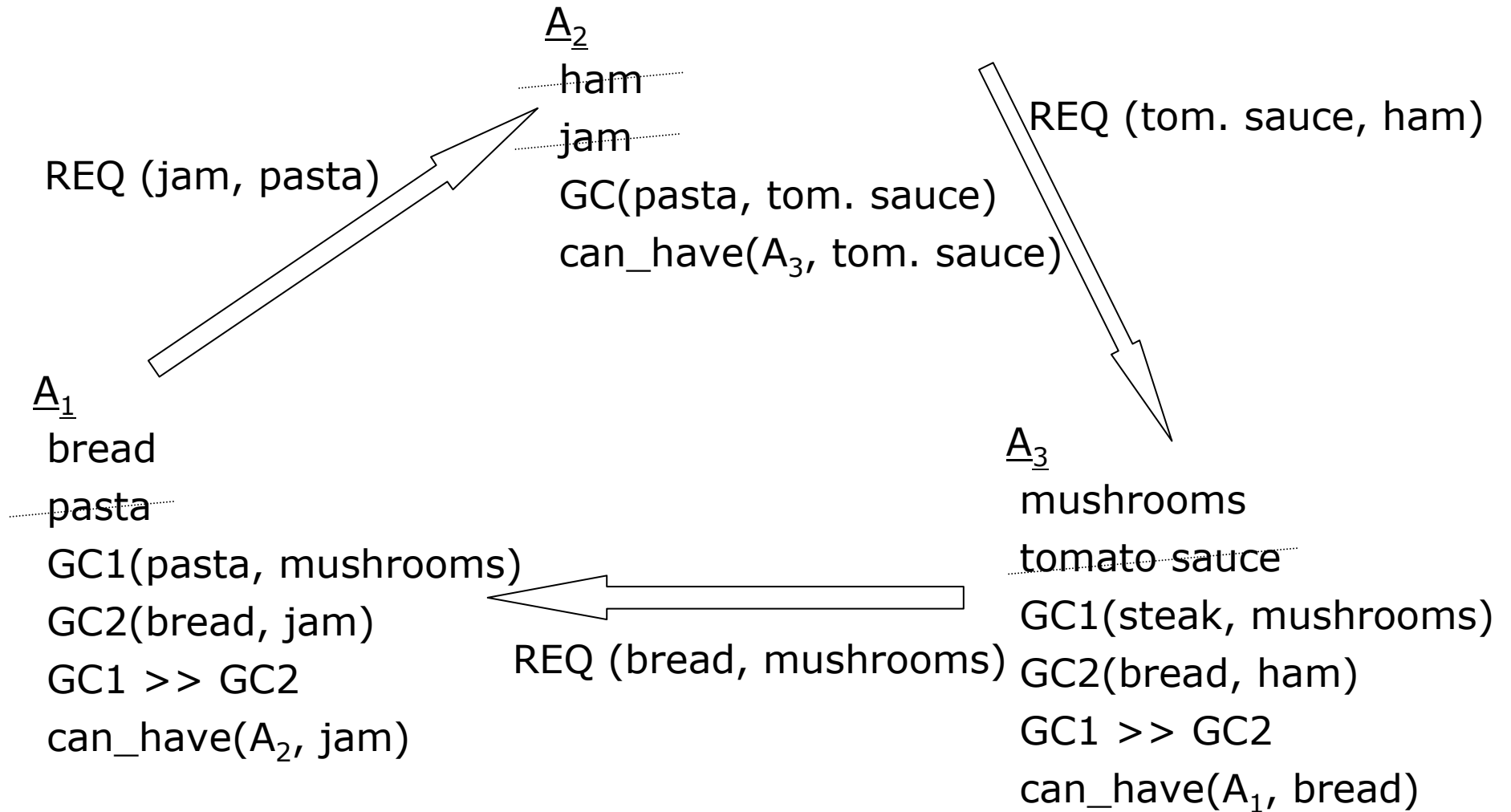
# The Problem of Stabilization: a Food Example

---



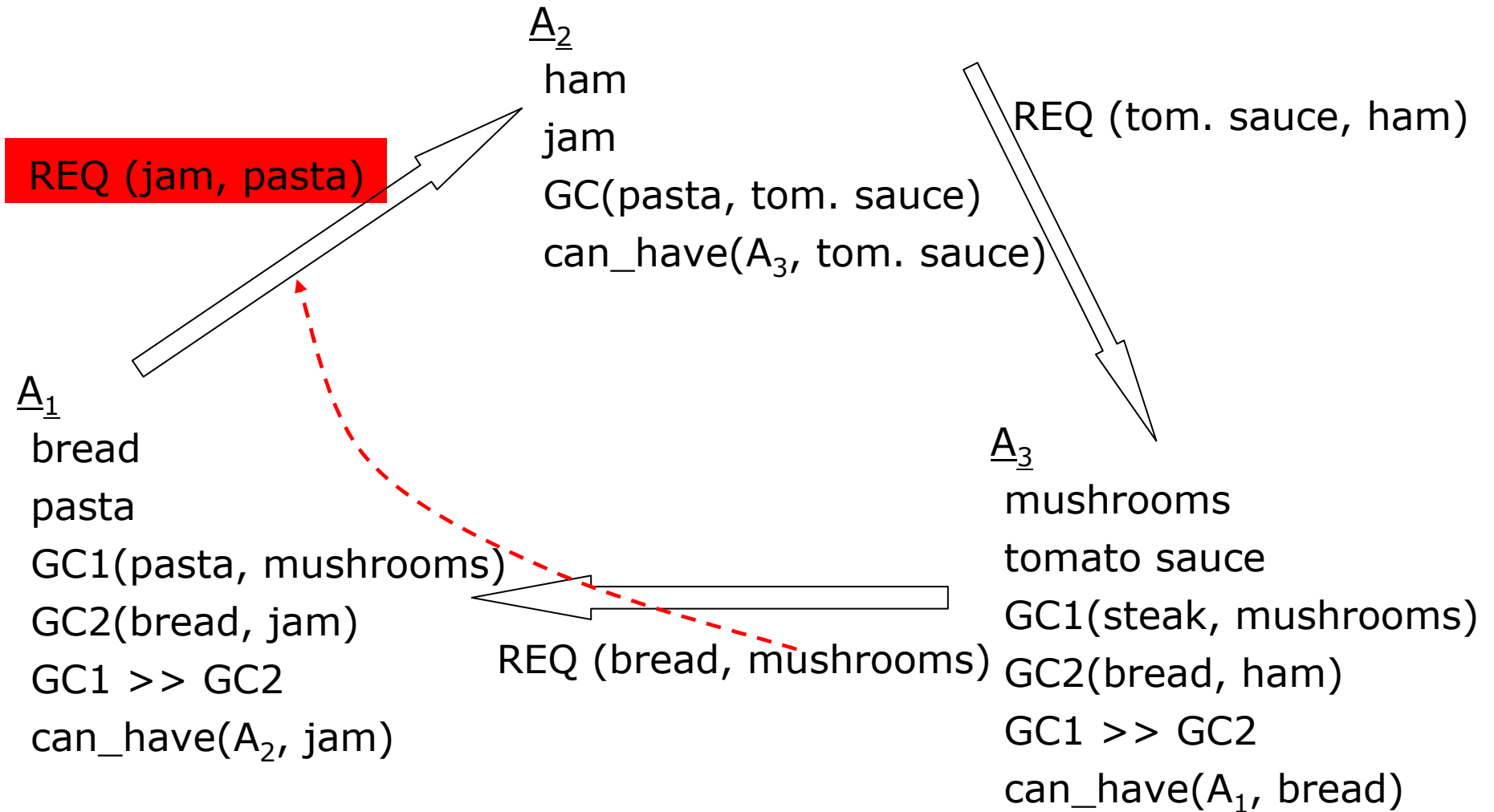
# The Problem of Stabilization: a Food Example

---

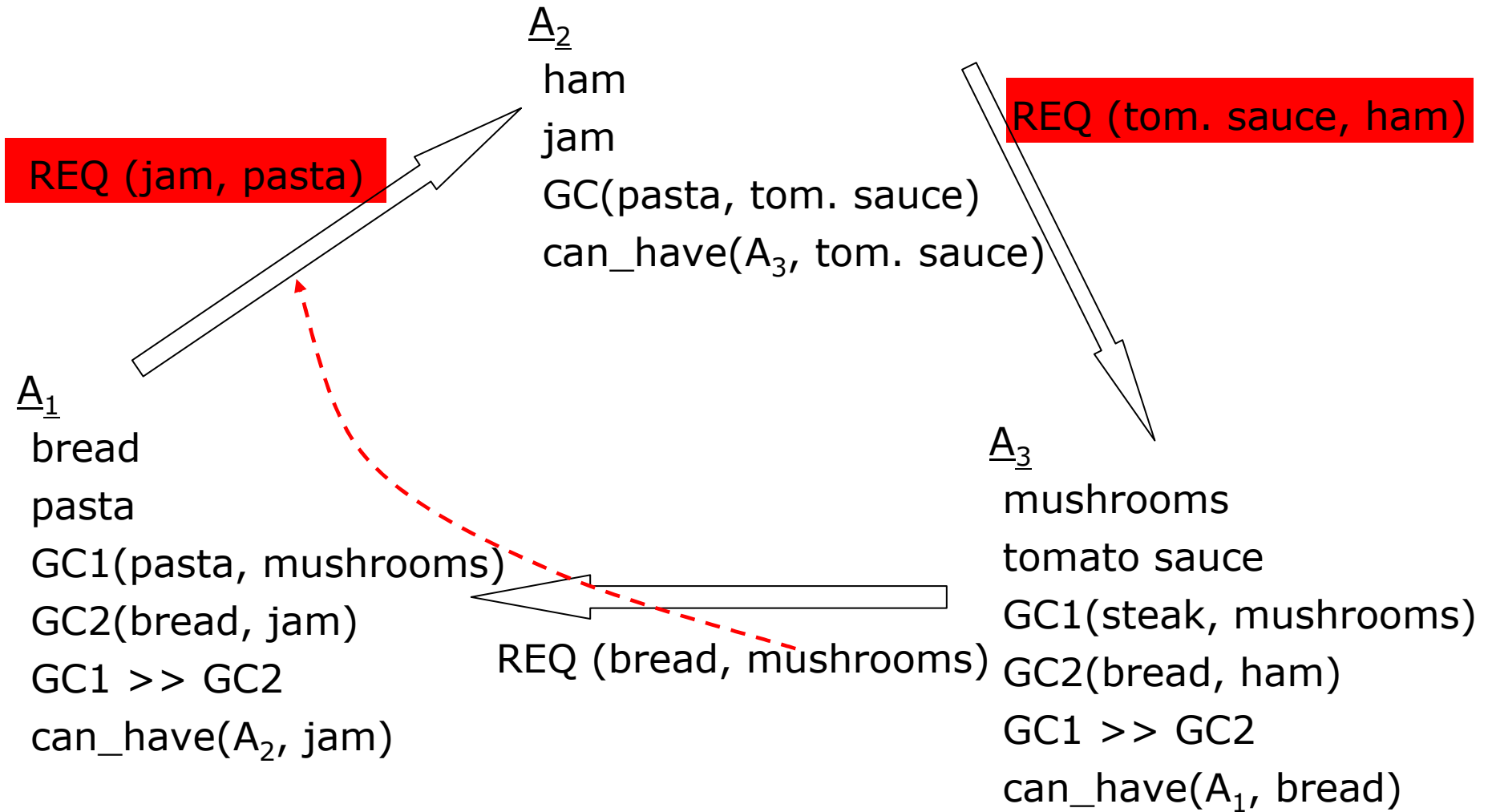


# The Problem of Stabilization: a Food Example

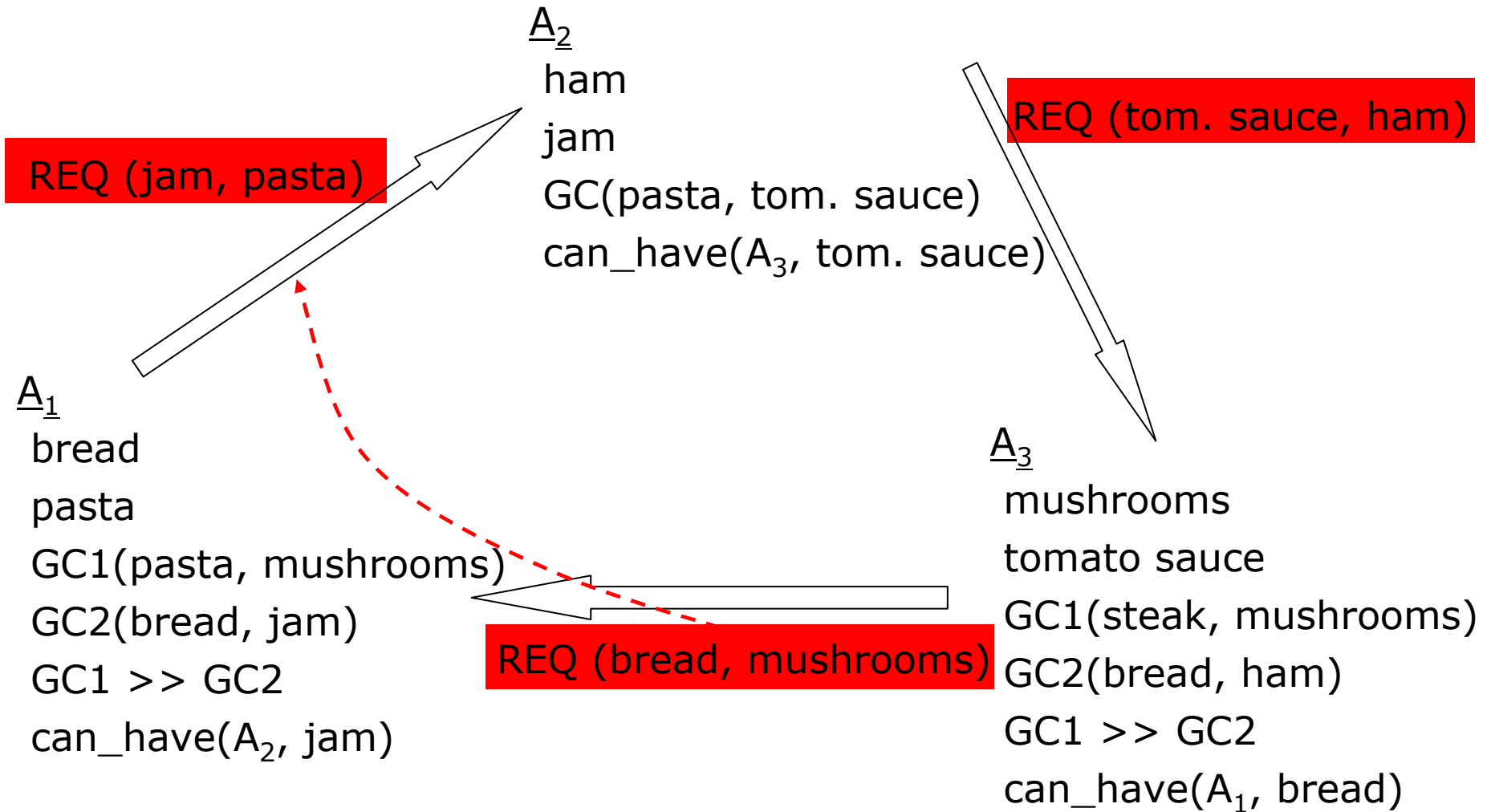
---



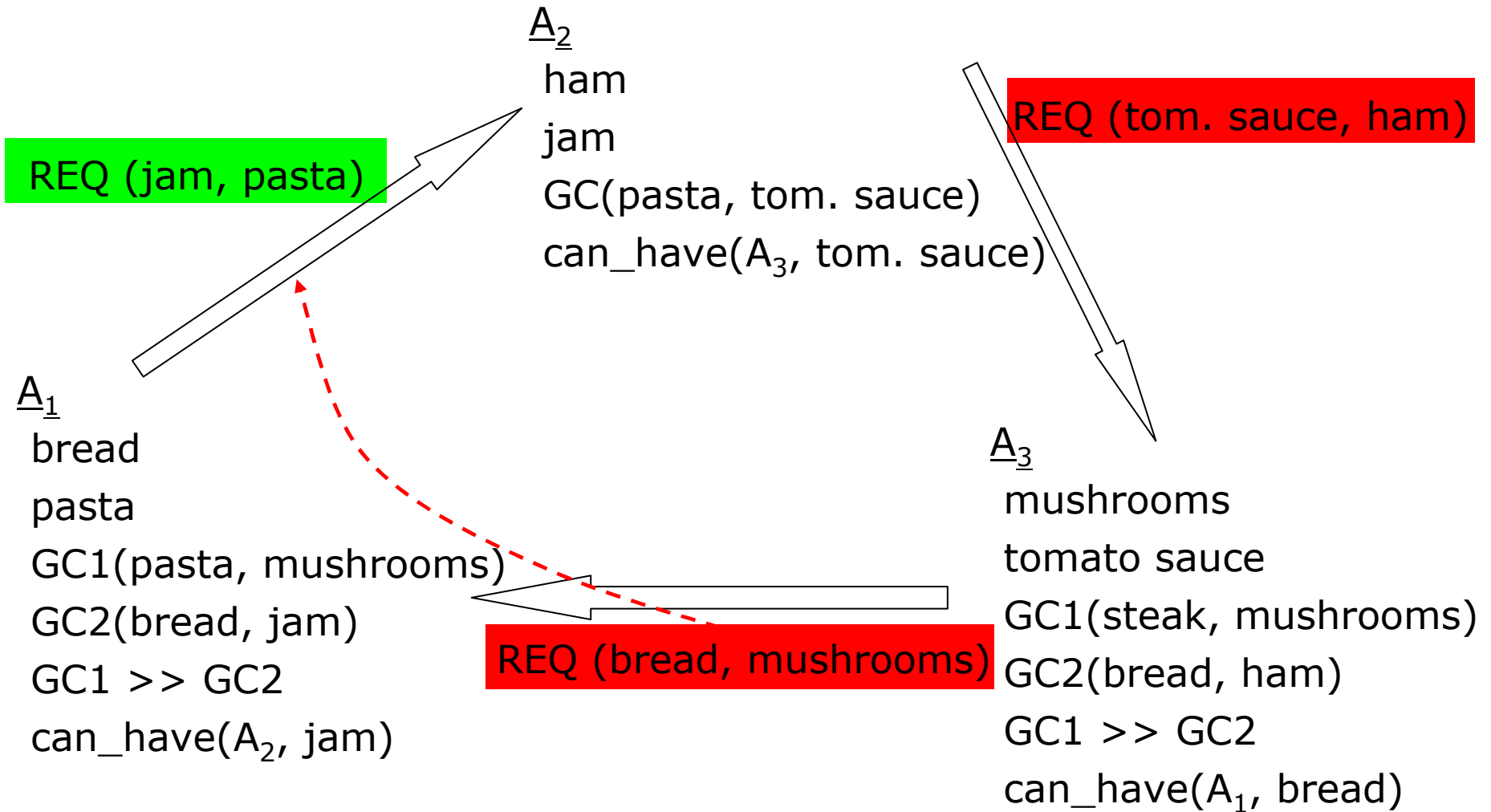
# The Problem of Stabilization: a Food Example



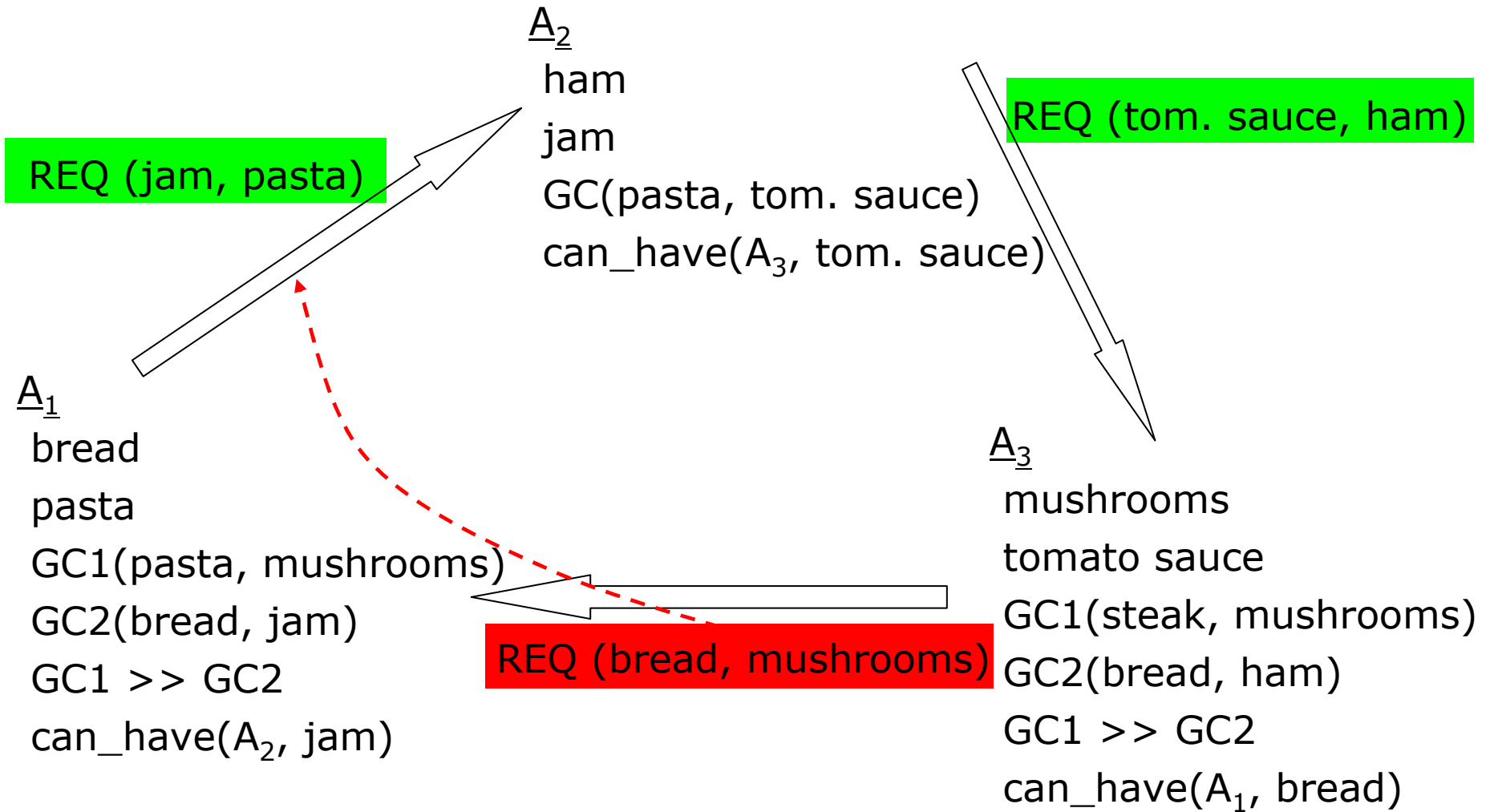
# The Problem of Stabilization: a Food Example



# The Problem of Stabilization: a Food Example



# The Problem of Stabilization: a Food Example

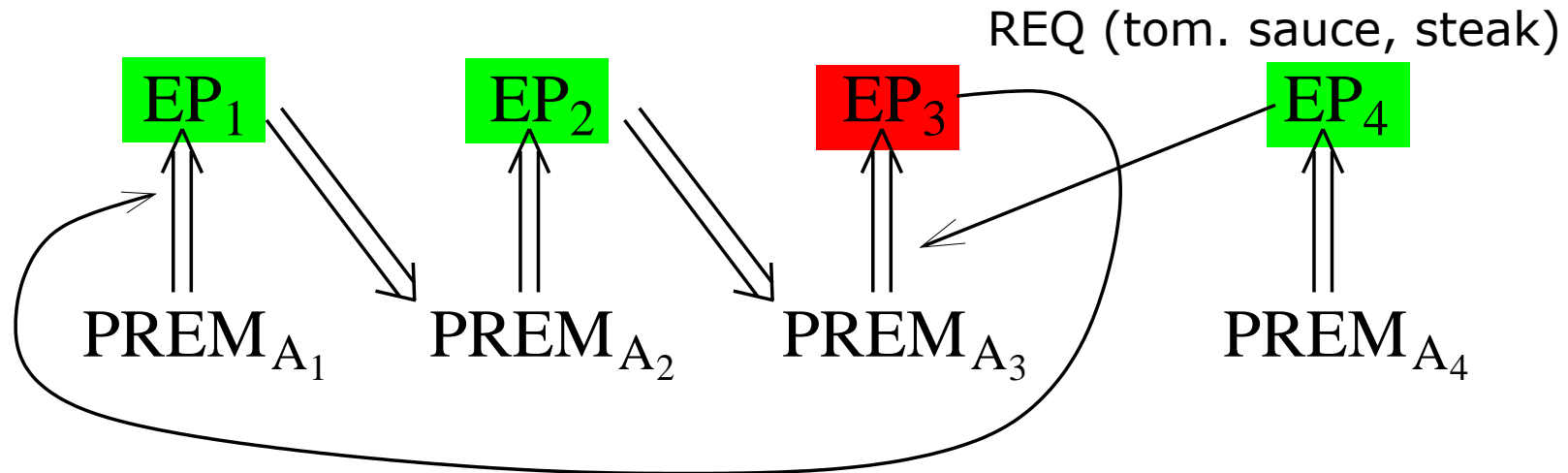


THE SONG REMAINS THE SAME...

# The Problem of Stabilization: Observations

---

- The stabilization problem we consider concerns defeat status computation, not the construction of arguments
- The presence of a cycle does not give rise per se to a stability problem



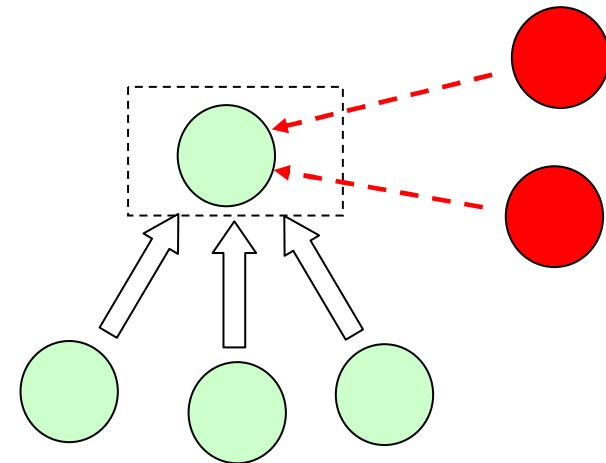
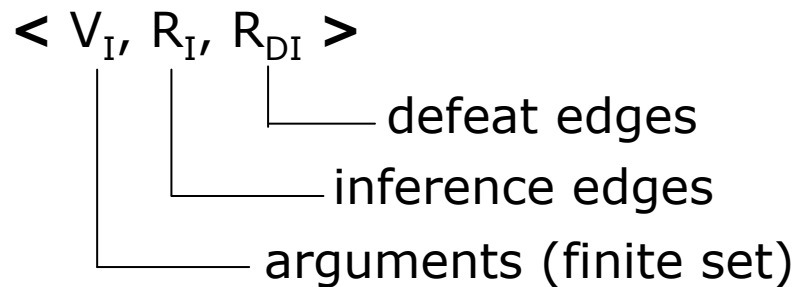
- The stabilization problem does not arise in architectural models that are centralized or where entire arguments are exchanged

# A distributed model for argumentation

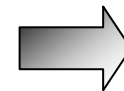
## A model for argumentation in multi agent systems

- no assumptions on the number of involved agents
- distributed model: no centralized control
- no need to communicate the underlying justification

## Distributed inference graph



- Every node is a process
- Information about direct defeaters and immediate subarguments
- Dynamic graph
- No synchronism



**Need for a self-stabilizing algorithm**

Starting from any initial state, always terminates to a legal state in a finite amount of time

# Outline of the talk

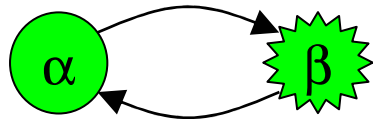
---

- Why distributed defeat status computation may be interesting?
- How much difficult is it?
  - exploring tentative solutions
  - impossibility theorems
- Two solutions (under some restrictive conditions)

# Coherence Conditions: a Tentative Approach

---

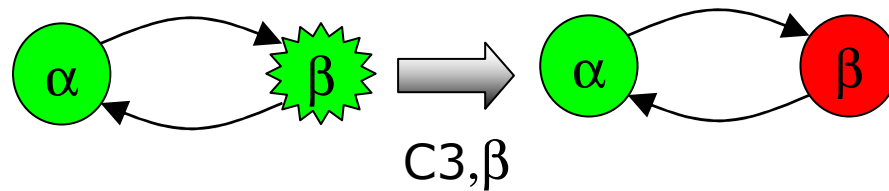
- C1:  $\text{parents}(\alpha) = \emptyset \Rightarrow L(\alpha) = \text{UNDEF}$
- C2:  $\forall \beta \in \text{parents}(\alpha) L(\beta) = \text{DEF} \Rightarrow L(\alpha) = \text{UNDEF}$
- C3:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{UNDEF} \Rightarrow L(\alpha) = \text{DEF}$
- C4:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{PROV}$  and  $\forall \beta \in \text{parents}(\alpha) L(\beta) \neq \text{UNDEF} \Rightarrow L(\alpha) = \text{PROV}$



# Coherence Conditions: a Tentative Approach

---

- C1:  $\text{parents}(\alpha) = \emptyset \Rightarrow L(\alpha) = \text{UNDEF}$
- C2:  $\forall \beta \in \text{parents}(\alpha) L(\beta) = \text{DEF} \Rightarrow L(\alpha) = \text{UNDEF}$
- C3:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{UNDEF} \Rightarrow L(\alpha) = \text{DEF}$
- C4:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{PROV}$  and  $\forall \beta \in \text{parents}(\alpha) L(\beta) \neq \text{UNDEF} \Rightarrow L(\alpha) = \text{PROV}$

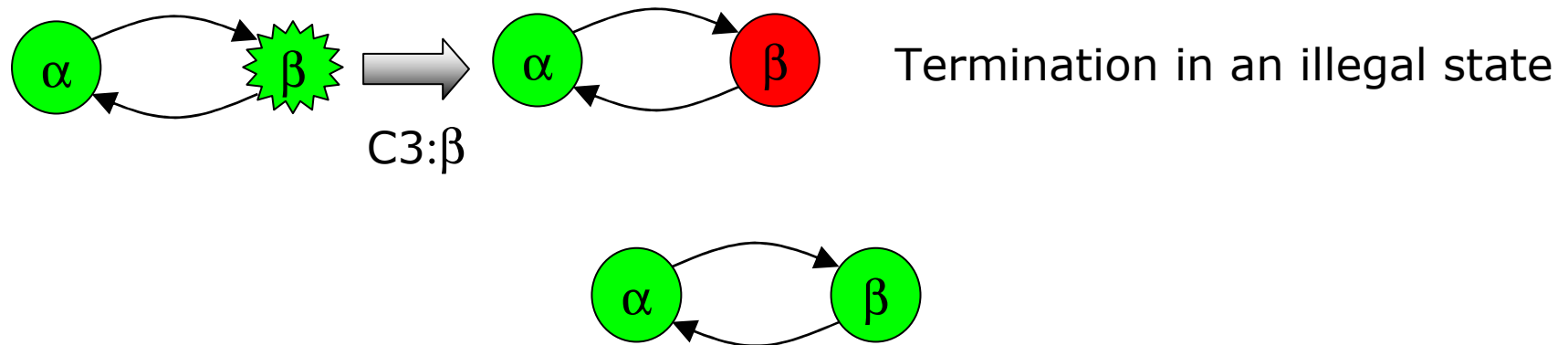


Termination in an illegal state

# Coherence Conditions: a Tentative Approach

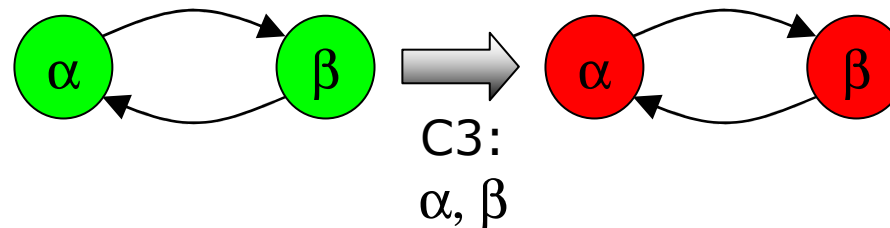
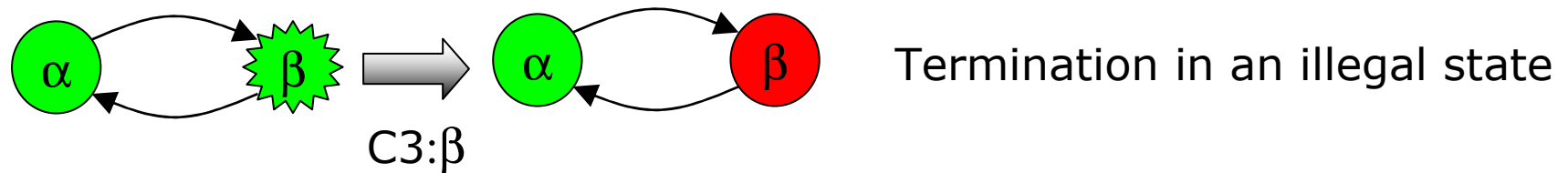
---

- C1:  $\text{parents}(\alpha) = \emptyset \Rightarrow L(\alpha) = \text{UNDEF}$
- C2:  $\forall \beta \in \text{parents}(\alpha) L(\beta) = \text{DEF} \Rightarrow L(\alpha) = \text{UNDEF}$
- C3:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{UNDEF} \Rightarrow L(\alpha) = \text{DEF}$
- C4:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{PROV}$  and  $\forall \beta \in \text{parents}(\alpha) L(\beta) \neq \text{UNDEF} \Rightarrow L(\alpha) = \text{PROV}$



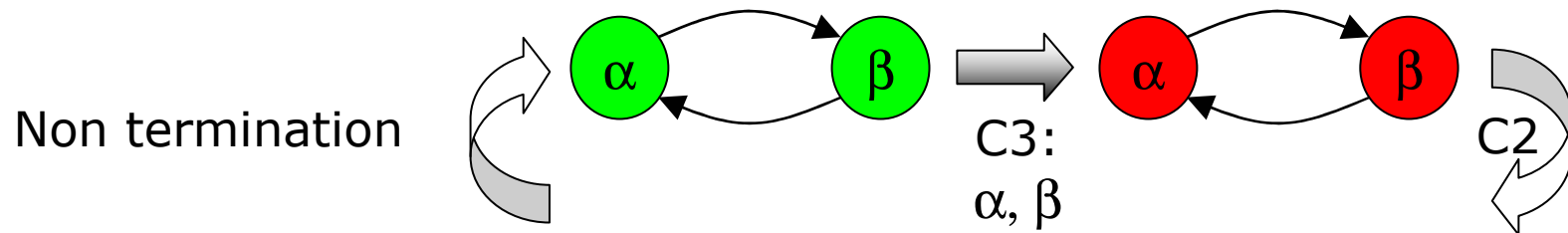
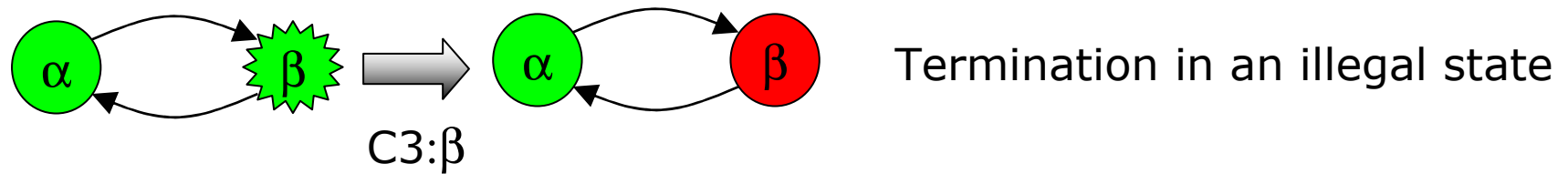
# Coherence Conditions: a Tentative Approach

- C1:  $\text{parents}(\alpha) = \emptyset \Rightarrow L(\alpha) = \text{UNDEF}$
- C2:  $\forall \beta \in \text{parents}(\alpha) L(\beta) = \text{DEF} \Rightarrow L(\alpha) = \text{UNDEF}$
- C3:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{UNDEF} \Rightarrow L(\alpha) = \text{DEF}$
- C4:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{PROV}$  and  $\forall \beta \in \text{parents}(\alpha) L(\beta) \neq \text{UNDEF} \Rightarrow L(\alpha) = \text{PROV}$



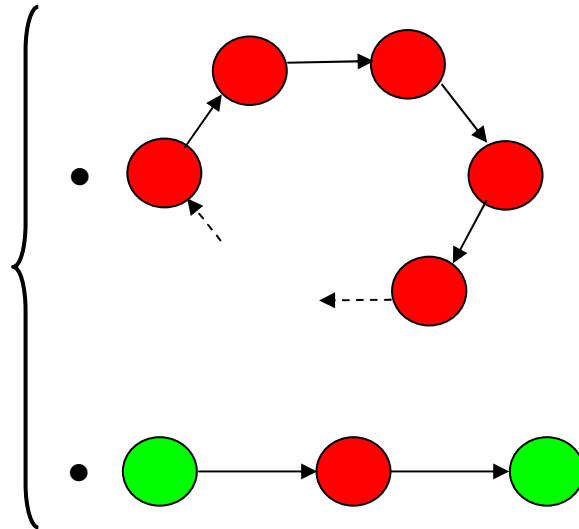
# Coherence Conditions: a Tentative Approach

- C1:  $\text{parents}(\alpha) = \emptyset \Rightarrow L(\alpha) = \text{UNDEF}$
- C2:  $\forall \beta \in \text{parents}(\alpha) L(\beta) = \text{DEF} \Rightarrow L(\alpha) = \text{UNDEF}$
- C3:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{UNDEF} \Rightarrow L(\alpha) = \text{DEF}$
- C4:  $\exists \beta \in \text{parents}(\alpha) L(\beta) = \text{PROV}$  and  $\forall \beta \in \text{parents}(\alpha) L(\beta) \neq \text{UNDEF} \Rightarrow L(\alpha) = \text{PROV}$



# An impossibility result

For any "valid" semantics, i.e. such that

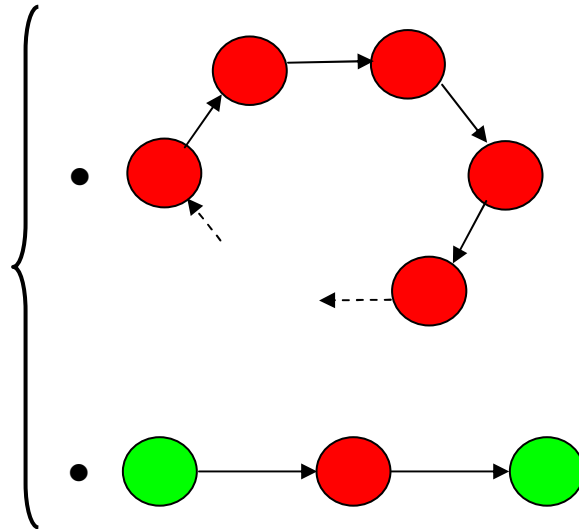


NO SELF-STABILIZING ALGORITHM EXISTS, EVEN IN CASE OF A CENTRALIZED SCHEDULER!

WHY?

# An impossibility result

For any "valid" semantics, i.e. such that



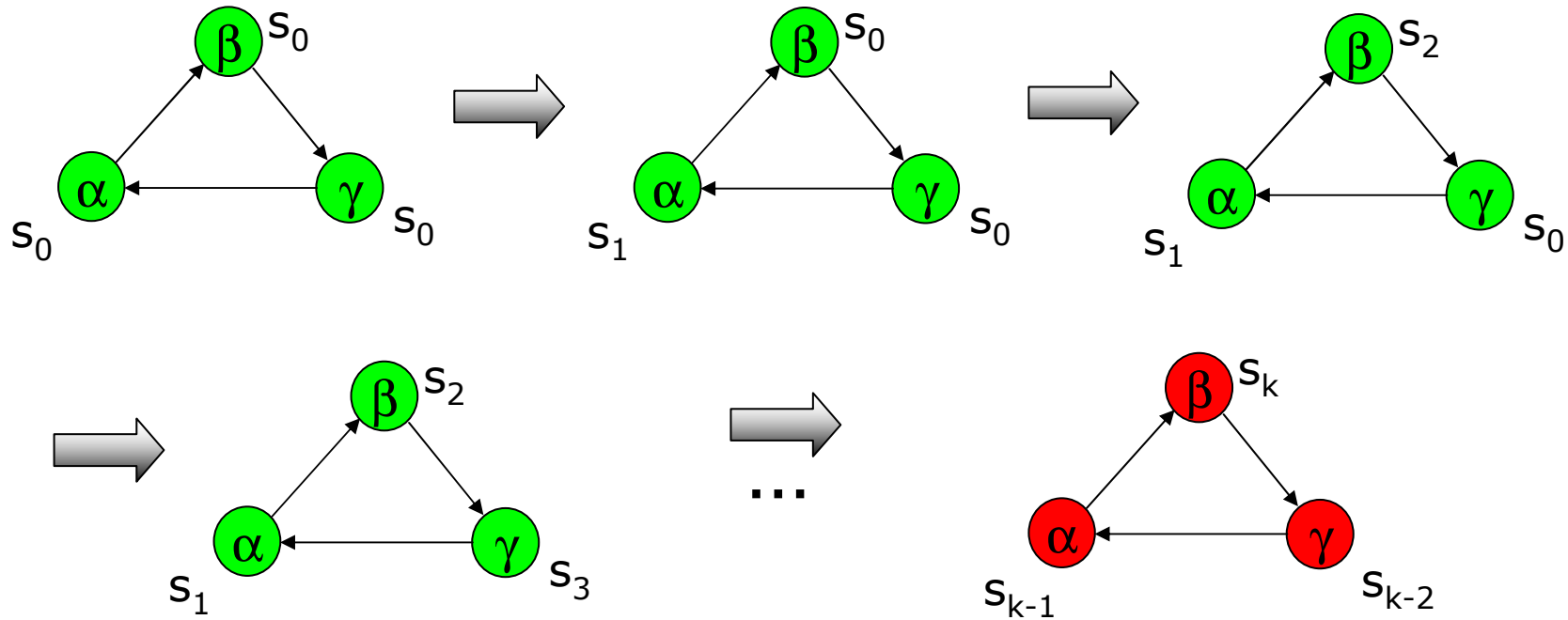
NO SELF-STABILIZING ALGORITHM EXISTS, EVEN IN CASE OF A CENTRALIZED SCHEDULER!

## WHY?

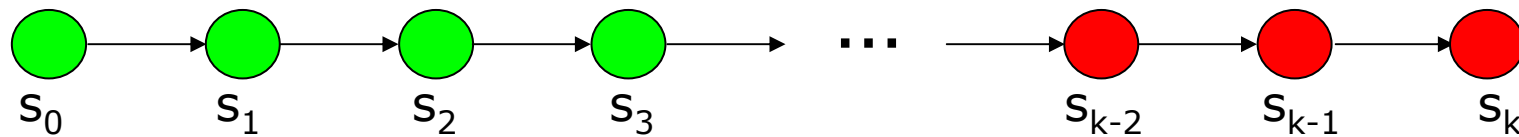
**Inherent impossibility of handling chains AND cycles correctly**

# An impossibility result (2)

$S_0$ : stable for initial nodes



but then this chain with this initial state is stable!

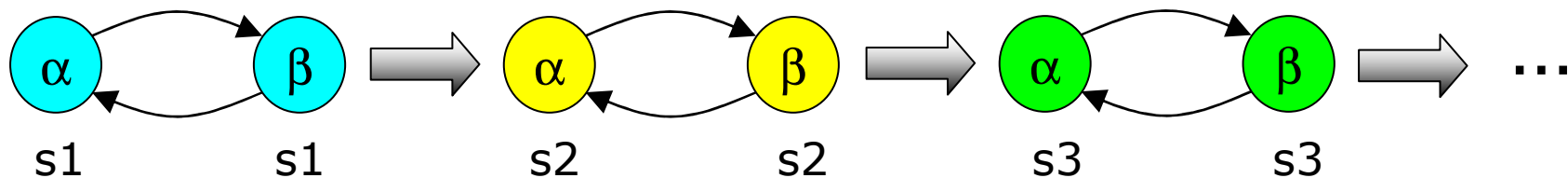


# What to do?

## Two investigation lines pursued:

- 1) topological restriction (only rebutting defeat):
  - correct algorithm
- 2) approximated algorithm such that:
  - always terminates for any inference graph
  - final state is correct under topological restrictions
  - incorrect results adhere to a "cautiousness criterion"

## Another impossibility result against multiple status semantics



Impossibility of symmetry breaking in uniform networks

 We consider the **GROUND SEMANTICS**

# Outline of the talk

---

- Why distributed defeat status computation may be interesting?
- How much difficult is it?
- Two solutions (under some restrictive conditions)

# The Case of Rebutting Defeat: a Specific Algorithm (1)

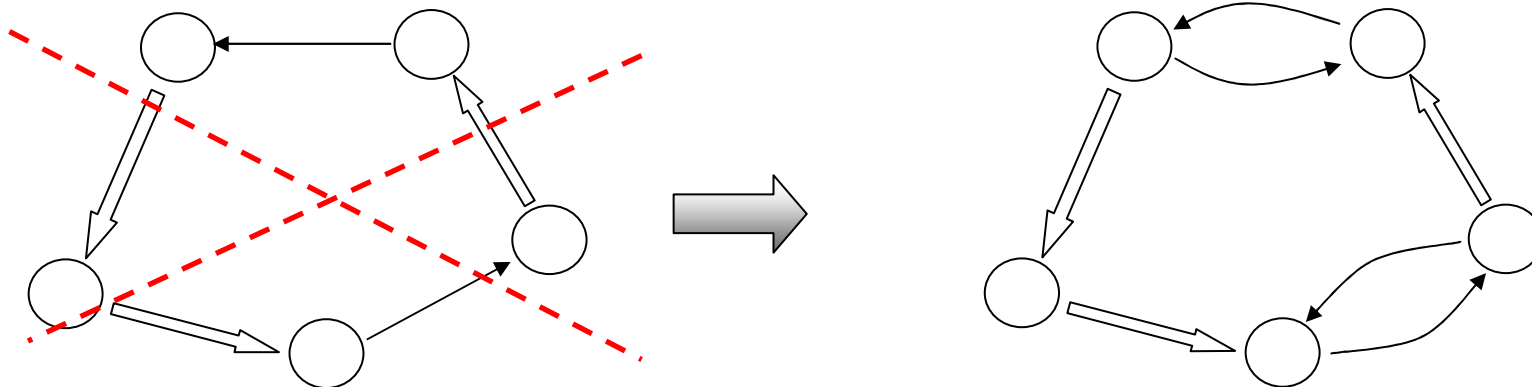
---

## Rebutting defeat

$\forall \underline{\alpha} \in \text{sub}(\alpha), \text{strength}(\underline{\alpha}) \geq \text{strength}(\alpha)$

$R_{DI} = \{ (\alpha, \beta) : \text{concl}(\alpha) = \neg \text{concl}(\beta) \wedge \text{strength}(\alpha) \geq \text{strength}(\beta) \}$

## Specific Property of the Topology of the Inference Graph



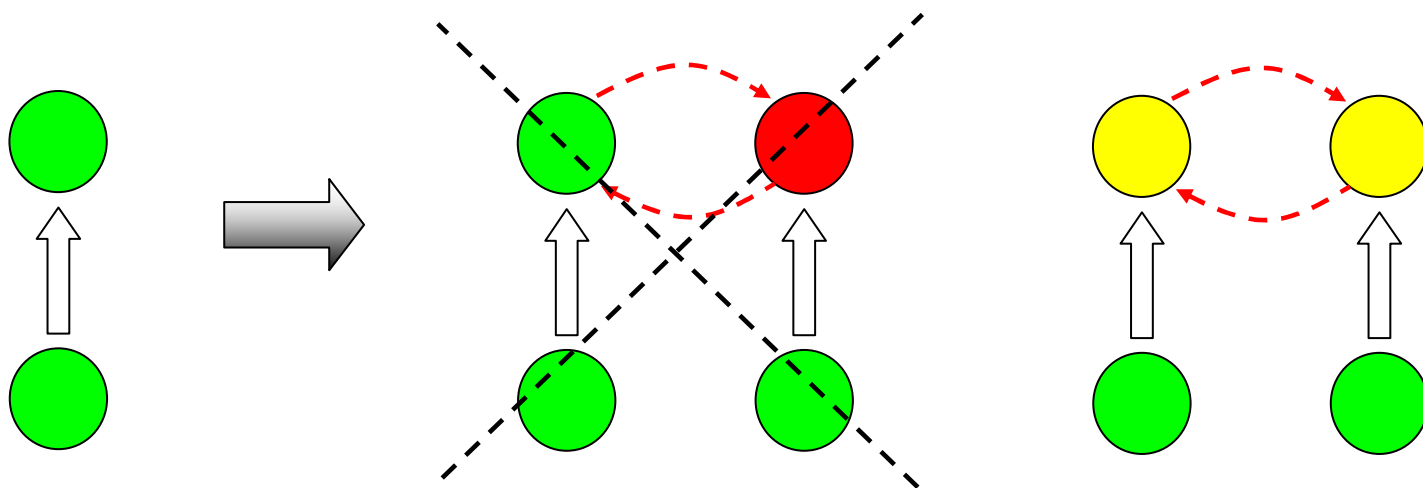
# The Case of Rebutting Defeat: a Specific Algorithm (2)

---

$$\text{d-parents}(\alpha): \begin{cases} \text{contenders}(\alpha) : \langle \beta, \alpha \rangle \in R_{DI} \text{ and } \langle \alpha, \beta \rangle \in R_{DI} \\ \text{superiors}(\alpha) : \text{d-parents}(\alpha) \setminus \text{contenders}(\alpha) \end{cases}$$

➔ A contender can not bring about an argument to be defeated

## Example



## The Case of Rebutting Defeat: a Specific Algorithm (3)

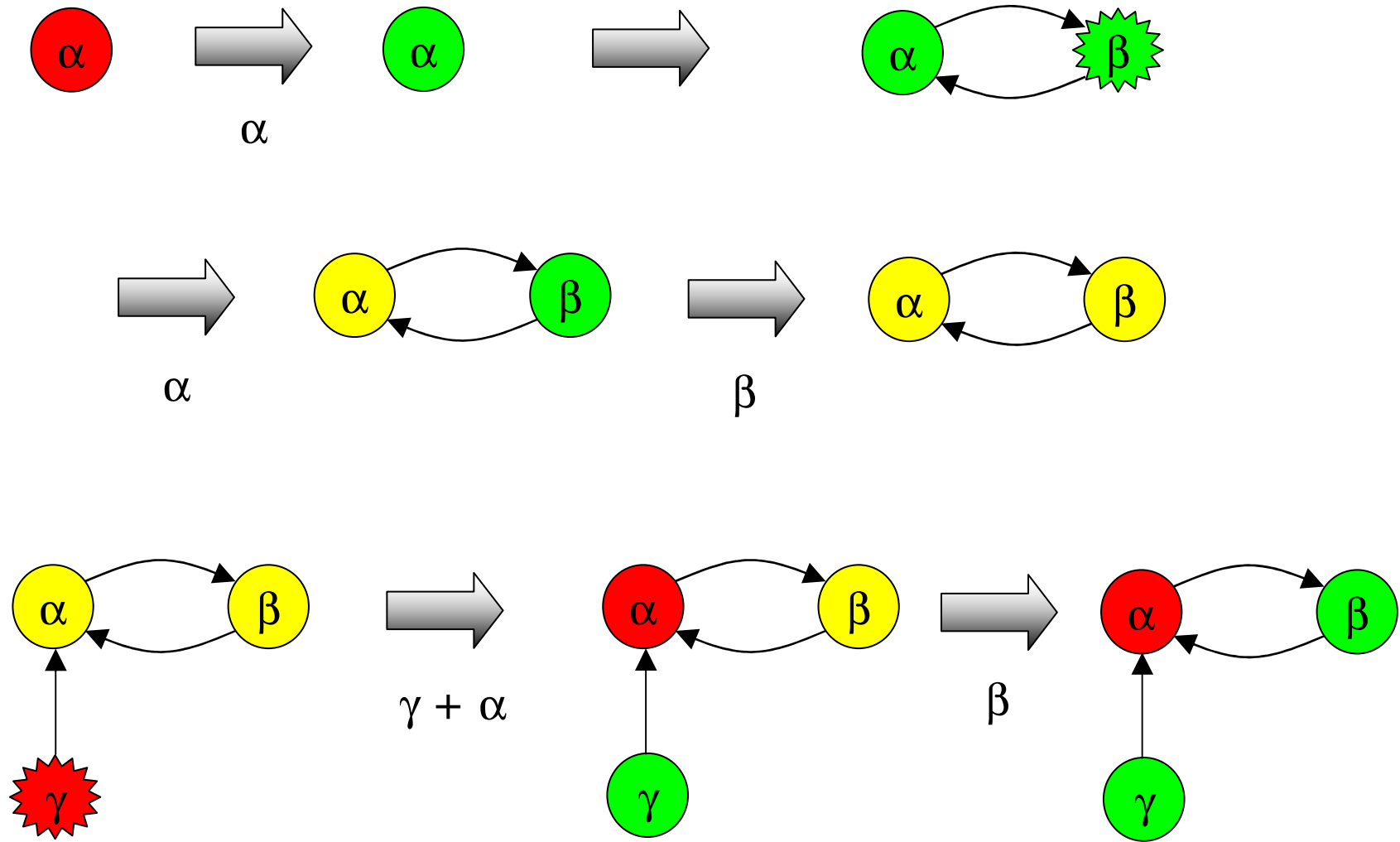
---

- $\exists \beta \in \text{IMM}(\alpha) : s[\beta] = \text{DEF} \Rightarrow s[\alpha] := \text{DEF}$
- $\exists \beta \in \text{IMM}(\alpha) : s[\beta] = \text{PROV}$  and  $\forall \gamma \in \text{IMM}(\alpha) s[\gamma] \neq \text{DEF}$ 
  - $\exists \beta \in \text{superiors}(\alpha) : s[\beta] = \text{UNDEF} \Rightarrow s[\alpha] := \text{DEF}$
  - otherwise  $\Rightarrow s[\alpha] := \text{PROV}$
- $\text{IMM}(\alpha) = \emptyset$  or  $\forall \beta \in \text{IMM}(\alpha) s[\beta] = \text{UNDEF}$ 
  - $\exists \beta \in \text{superiors}(\alpha) : s[\beta] = \text{UNDEF} \Rightarrow s[\alpha] := \text{DEF}$
  - $\exists \beta \in \text{superiors}(\alpha) : s[\beta] = \text{PROV}$  and  $\forall \gamma \in \text{superiors}(\alpha) s[\gamma] \neq \text{UNDEF}$   
 $\Rightarrow s[\alpha] := \text{PROV}$
  - $\text{superiors}(\alpha) = \emptyset$  or  $\forall \gamma \in \text{superiors}(\alpha) s[\gamma] = \text{DEF}$

$$s[\alpha] := \begin{cases} \text{UNDEF} & \text{if } \forall \gamma \in \text{contenders}(\alpha) s[\gamma] = \text{DEF} \\ \text{PROV} & \text{otherwise} \end{cases}$$

# The case of Rebutting Defeat: Example of Execution

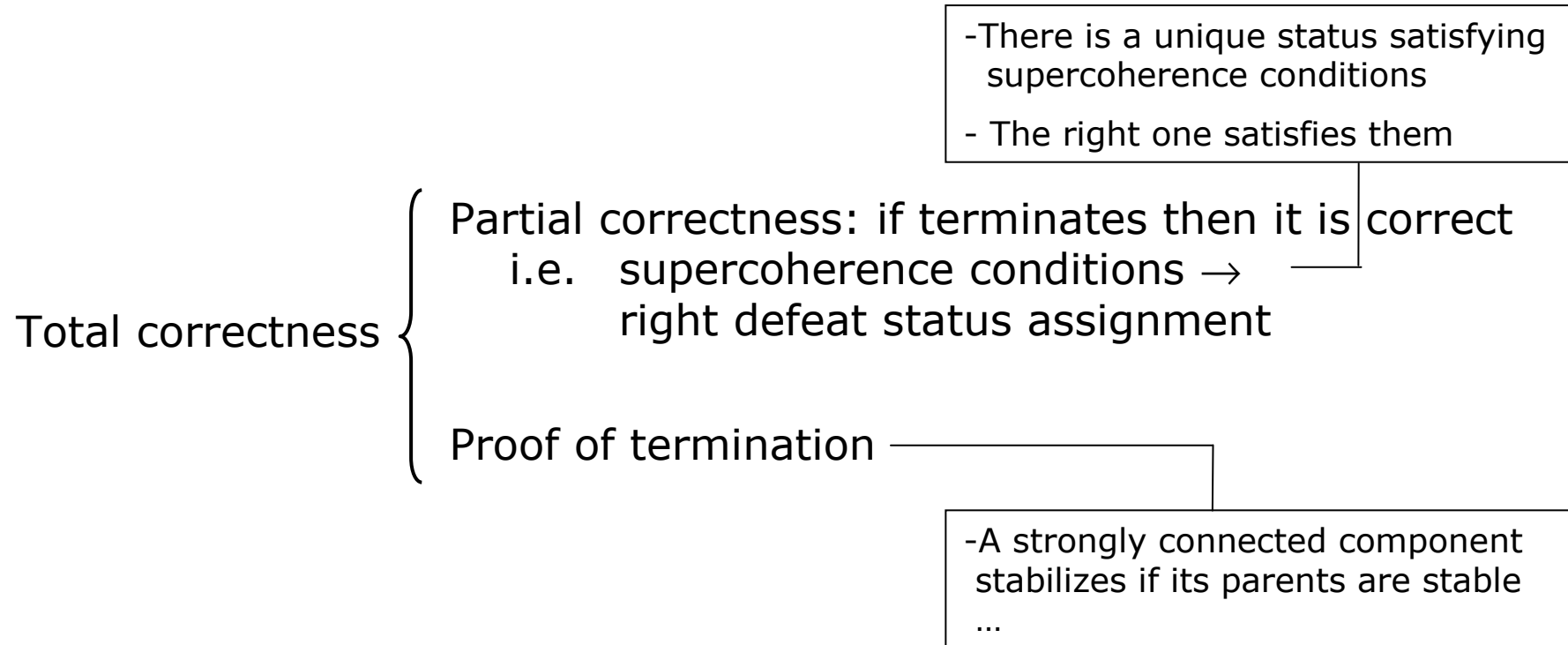
---



# Correctness

---

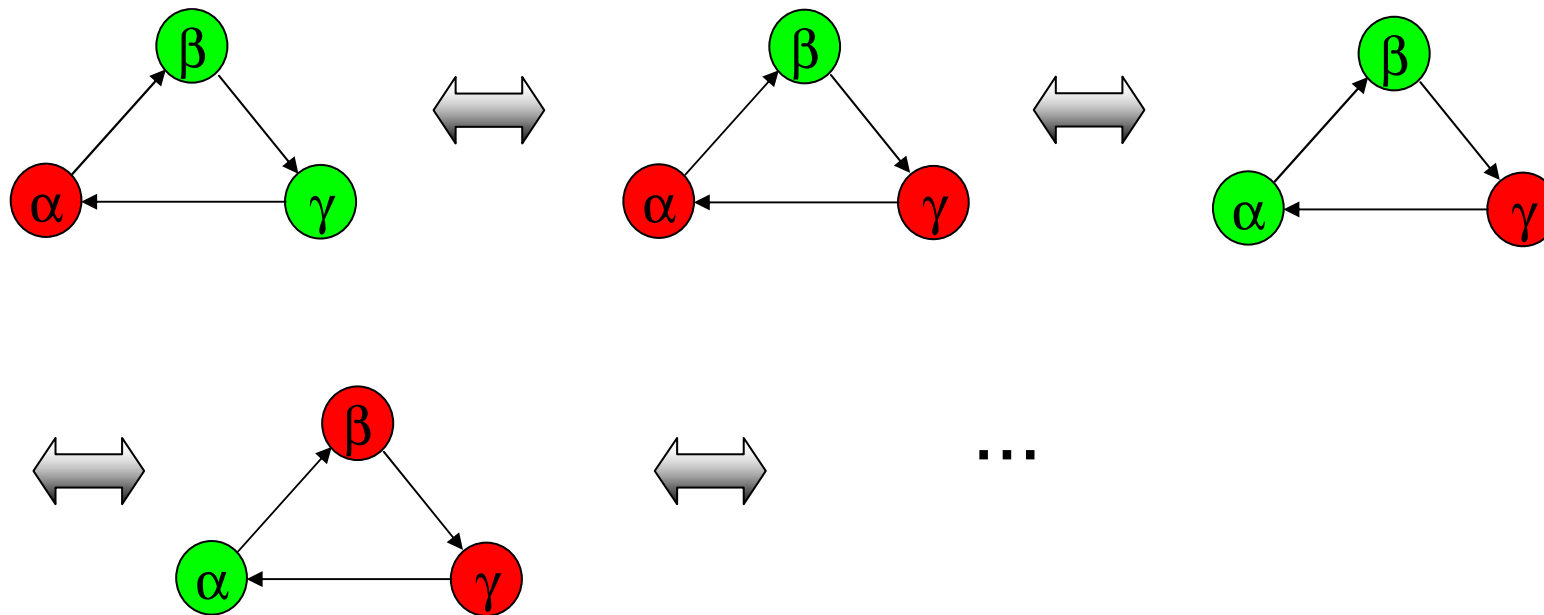
---



Correctness holds also in case of distributed schedulers

# Another example of execution

Termination is not guaranteed outside the “rebutting” family of graphs



[inside the rebutting family of graphs we are sure that any “unidirectional attack”  $\longrightarrow$  is not involved in a cycle]

# Handling the General Case: possible choices

---

## Requirements

- Termination in any case
- Right defeat status assignment for a family of graphs satisfying a specified topological restriction
- Defeat status assignment “cautiously” incorrect outside this family: some arguments may be left “undecided” (provisionally defeated)

## Two options

Cycles and chains can not be handled correctly together:

- Properly treatment of defeat cycles of arbitrary length, but inability to assign the right status to some defeat chains
- Properly treatment of defeat chains of arbitrary length, but inability to assign the right status to some defeat cycles

# Handling the General Case: possible choices

---

## Requirements

- Termination in any case
- Right defeat status assignment for a family of graphs satisfying a specified topological restriction
- Defeat status assignment “cautiously” incorrect outside this family: some arguments may be left “undecided” (provisionally defeated)

## Two options

Cycles and chains can not be handled correctly together:

- Properly treatment of defeat cycles of arbitrary length, but inability to assign the right status to some defeat chains
- Properly treatment of defeat chains of arbitrary length, but inability to assign the right status to some defeat cycles



INCOMPATIBLE WITH CAUTIOUSNESS CRITERION  
(cycles should be provisionally defeated in the right status)

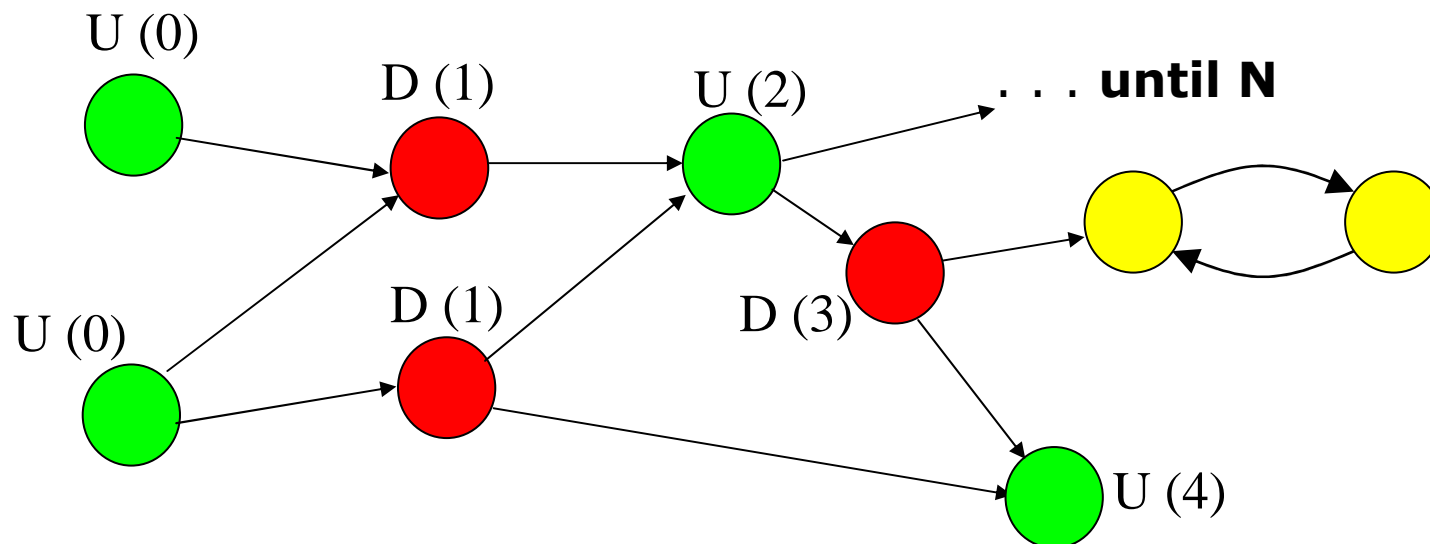
# Handling the General Case (1)

## The idea

$$s[\alpha] = \begin{cases} \text{DEF } (x) \\ \text{UNDEF } (x) \\ \text{PROV} \end{cases} \quad \text{where } x = d[\beta] + 1$$

└ "determinant" node

N: a fixed constant



## Handling the General Case (2)

---

- $\text{EXUNDEF}(\alpha) \Rightarrow$

$$s[\alpha] := \begin{cases} \text{DEF}(\min[\text{DDEF}(\alpha)]) & \text{if } \min[\text{DDEF}(\alpha)] \leq N \\ \text{PROV} & \text{otherwise} \end{cases}$$

- $\neg\text{EXUNDEF}(\alpha)$  and  $\text{EXPROV}(\alpha) \Rightarrow s[\alpha] := \text{PROV}$

- $\neg\text{EXUNDEF}(\alpha)$  and  $\neg\text{EXPROV}(\alpha) \Rightarrow$

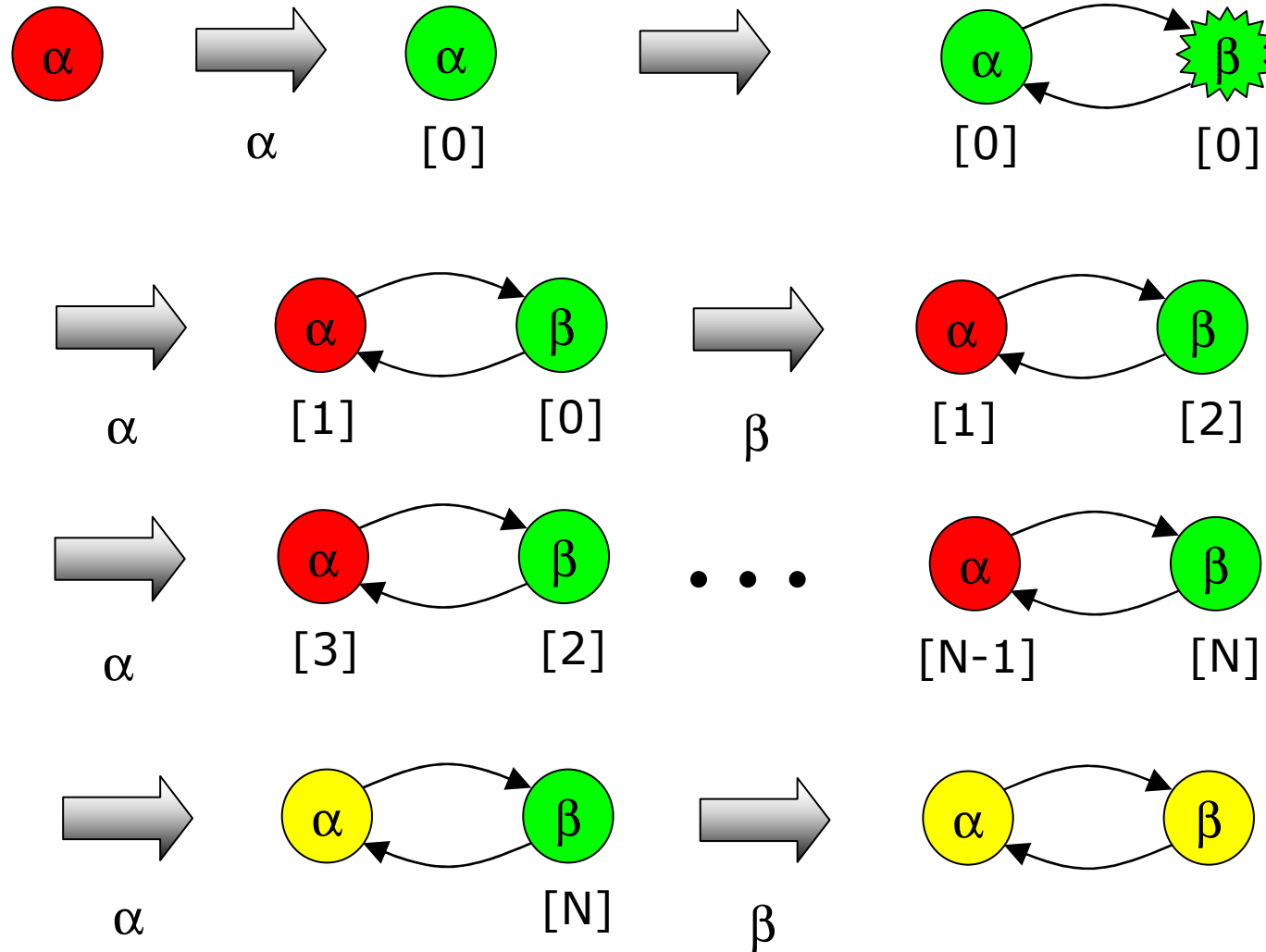
$$s[\alpha] = \begin{cases} \text{UNDEF}(\max[\text{DUND}(\alpha)]) & \text{if } \max[\text{DUND}(\alpha)] \leq N \\ \text{PROV} & \text{otherwise} \end{cases}$$

$\text{EXUNDEF}(\alpha) \equiv \exists \beta \in \text{IMM}(\alpha) : s[\beta] = \text{DEF}$  or  $\exists \beta \in \text{d-parents}(\alpha) : s[\beta] = \text{UNDEF}$

$\text{EXPROV}(\alpha) \equiv \exists \beta \in (\text{IMM}(\alpha) \cup \text{d-parents}(\alpha)) : s[\beta] = \text{PROV}$

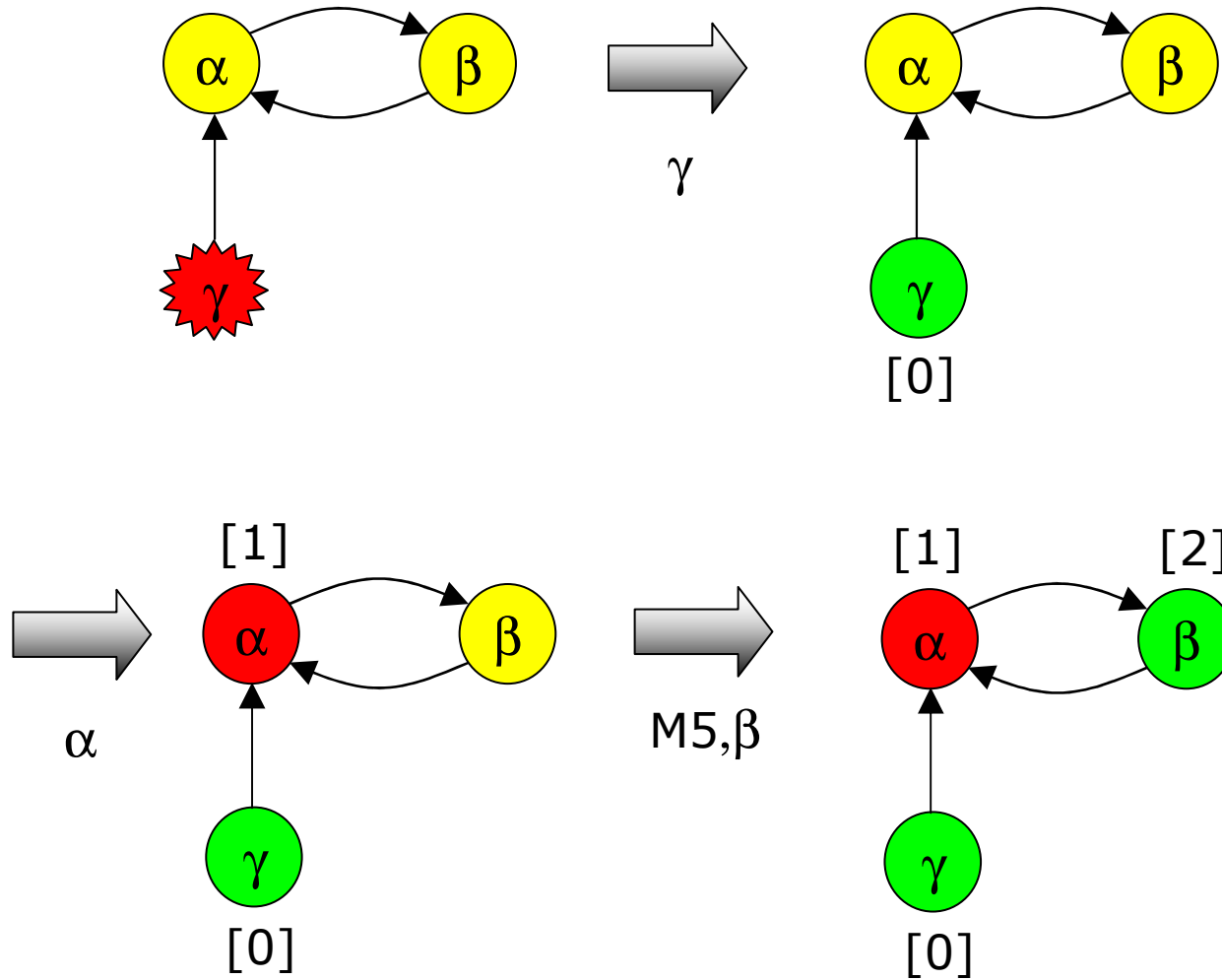
[see the paper for the definitions of  $\text{DDEF}(\alpha)$  and  $\text{DUND}(\alpha)$ ]

# Example of Execution (1)



# Example of Execution (2)

---



# Comparison between algorithms and main results

---

## ALGORITHM 1

Under topological restrictions:

termination + right defeat status assignment

round complexity:  $2n - 1$

Outside topological restrictions:

no guarantee of termination

## ALGORITHM 2

Always terminates

Defeat status assignment:

if  $N$  "high enough" w.r.t. max length of chains ( $\leftarrow N \geq n-1$ ) : RIGHT

otherwise: some nodes PROV DEF instead of DEF or UNDEF

Round complexity:  $(N+1) * MAXLENGTH \leq (N+1) * n$

## FOR BOTH ALGORITHMS

NO BETTER ROUND COMPLEXITY CAN BE ACHIEVED BY ANY

SELF-STABILIZING ALGORITHM OPERATING UNDER THE SAME CONDITIONS

MUCHAS GRACIAS  
POR VUESTRA  
ATENCIÓN

# More details and references in:

---

P. Baroni, M. Giacomin, G. Guida, "Self-stabilizing defeat status computation: dealing with conflict-management in multi-agent systems"  
*Artificial Intelligence*, vol. 165(2), 2005.