Model checking CTL

Natasha Alechina Brian Logan

Utrecht University

n.a.alechina@uu.nl b.s.logan@uu.nl

Local model checking

Model checking problem

$$M, q \quad \overset{?}{\models} \quad \varphi$$

state in the model — temporal formula

- this problem is sometimes called **local model checking**: it checks whether a formula is true locally (in a particular state)
- implemented as a function $mcheck(M, q, \varphi)$ such that:

$$mcheck(M, q, \varphi) = \begin{cases} true & \text{if} \quad M, q \models \varphi \\ false & \text{else} \end{cases}$$

Global model checking

- however it is more efficient to work with sets of states

Global model checking

We want to implement the function of two arguments M and ϕ

$$mcheck(M, \varphi) = \{q \in St \mid M, q \models \varphi\}$$

set of states q where
ϕ is true with q in the model M

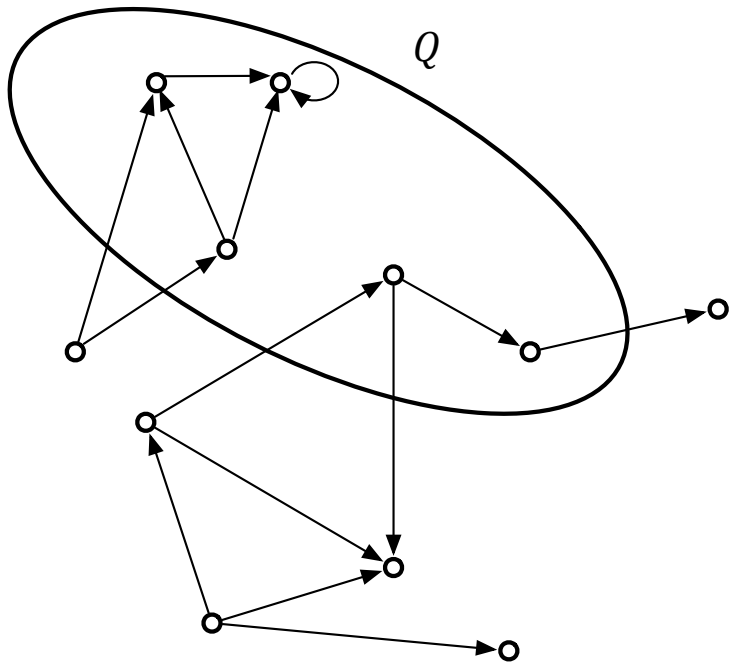
- often no harder than local model checking
- from now on, we will refer to global model checking as simply "model checking"

Model Checking for CTL

- as explained in the previous lecture, paths satisfying CTL formulae can be constructed incrementally, using fixpoint computations
- for “next time”, we simply take the appropriate pre-image
- for “long-term” properties, we repeatedly take the pre-image to compute the fixpoint
- computing the pre-image is the key building block in computing the fixpoint

Existential pre-image

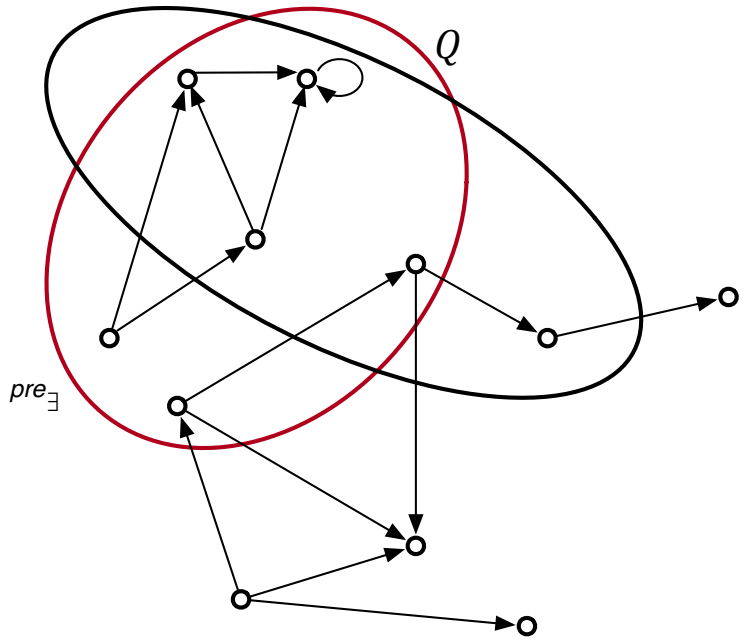
The existential pre-image of a set of states Q, $pre_{\exists}(Q)$ is all states q which have some state $q' \in Q$ as a successor



$$pre_{\exists}(Q) = \{q \mid \exists q'.q \longrightarrow q' \ \& \ q' \in Q\}$$

Existential pre-image

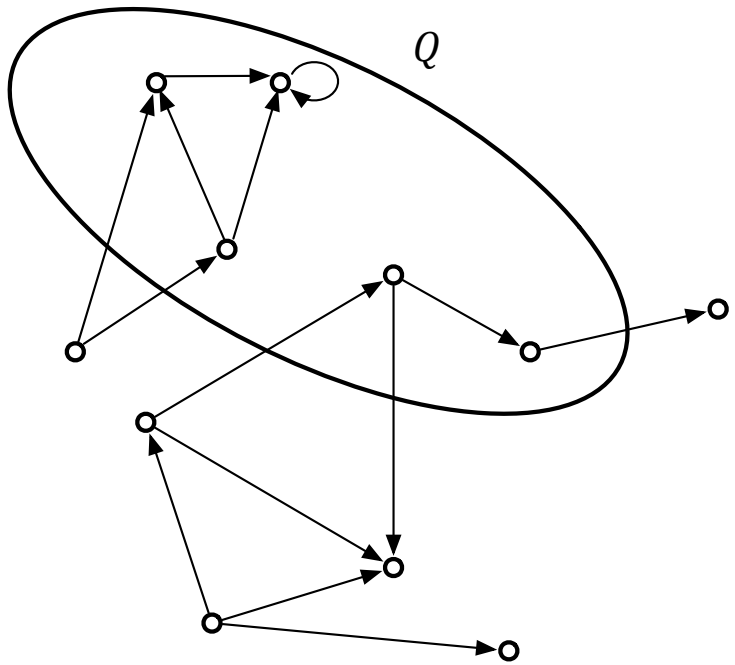
The existential pre-image of a set of states Q, $pre_{\exists}(Q)$ is all states q which have some state $q' \in Q$ as a successor



$$pre_{\exists}(Q) = \{q \mid \exists q'.q \longrightarrow q' \ \& \ q' \in Q\}$$

Universal pre-image

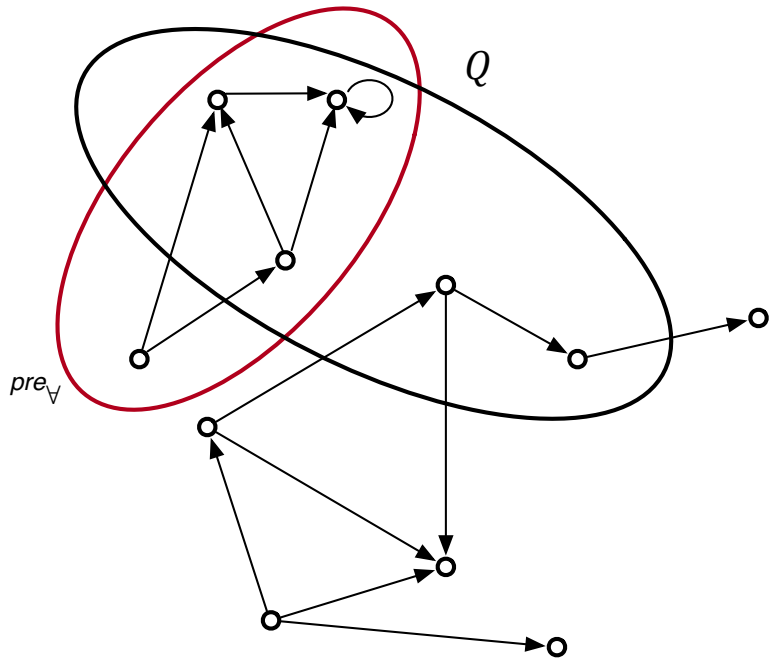
The universal pre-image of a set of states Q, $pre_{\forall}(Q)$ is all states q where all successor states q' are in Q



$$pre_{\forall}(Q) = \{q \mid \forall q'.q \longrightarrow q' \Rightarrow q' \in Q\}$$

Universal pre-image

The universal pre-image of a set of states Q, $pre_\forall(Q)$ is all states q where all successor states q' are in Q



$$pre_\forall(Q) = \{q \mid \forall q'.q \longrightarrow q' \Rightarrow q' \in Q\}$$

CTL model checking algorithm

To implement $mcheck(M, \varphi_0)$, we proceed as follows:

- compute the set of subformulas of φ_0, $Sub(\varphi_0)$
- the elements of $Sub(\varphi_0)$ are in order of complexity, atomic propositions first, then negations of atomic formulas, and so on, ending with φ_0 itself
- label states of M with each subformula $\varphi' \in Sub(\varphi_0)$ in turn, so that when we reach φ', states are already labelled with the subformulas of φ'
- labelling of states for each subformula proceeds by cases, one for each logical connective

function MCHECK(M, φ_0)

for $\varphi' \in Sub(\varphi_0)$ **do**

case $\varphi' = p$

$[\varphi']_M \leftarrow \mathcal{V}(p)$

case $\varphi' = \neg\psi$

$[\varphi']_M \leftarrow St \setminus [\psi]_M$

case $\varphi' = \psi_1 \wedge \psi_2$

$[\varphi']_M \leftarrow [\psi_1]_M \cap [\psi_2]_M$

case $\varphi' = \psi_1 \vee \psi_2$

$[\varphi']_M \leftarrow [\psi_1]_M \cup [\psi_2]_M$

case $\varphi' = EX\psi$

$[\varphi']_M \leftarrow pre_\exists([\psi]_M)$

case $\varphi' = EG\,\psi$

$Q_1 \leftarrow St; \quad Q_2 \leftarrow [\psi]_M$

while $Q_1 \not\subseteq Q_2$ **do**

$Q_1 \leftarrow Q_2; \quad Q_2 \leftarrow pre_\exists(Q_1) \cap Q_1$

$[\varphi']_M \leftarrow Q_1$

case $\varphi' = E\psi_1\, U\, \psi_2$

$Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow [\psi_2]_M$

while $Q_2 \not\subseteq Q_1$ **do**

$Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre_\exists(Q_1) \cap [\psi_1]_M$

$[\varphi']_M \leftarrow Q_1$

case $\varphi' = AX\psi$

$[\varphi']_M \leftarrow pre_\forall([\psi]_M)$

Only difference with E is that
for A, we use the universal pre-image

For example, for AX's case as we want to
ensure that we only label states
where all the successors are ψ states

case $\varphi' = AG\,\psi$

$Q_1 \leftarrow St; \quad Q_2 \leftarrow [\psi]_M$

while $Q_1 \not\subseteq Q_2$ **do**

$Q_1 \leftarrow Q_2; \quad Q_2 \leftarrow pre_\forall(Q_1) \cap Q_1$

$[\varphi']_M \leftarrow Q_1$

case $\varphi' = A\psi_1\,U\,\psi_2$

$Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow [\psi_2]_M$

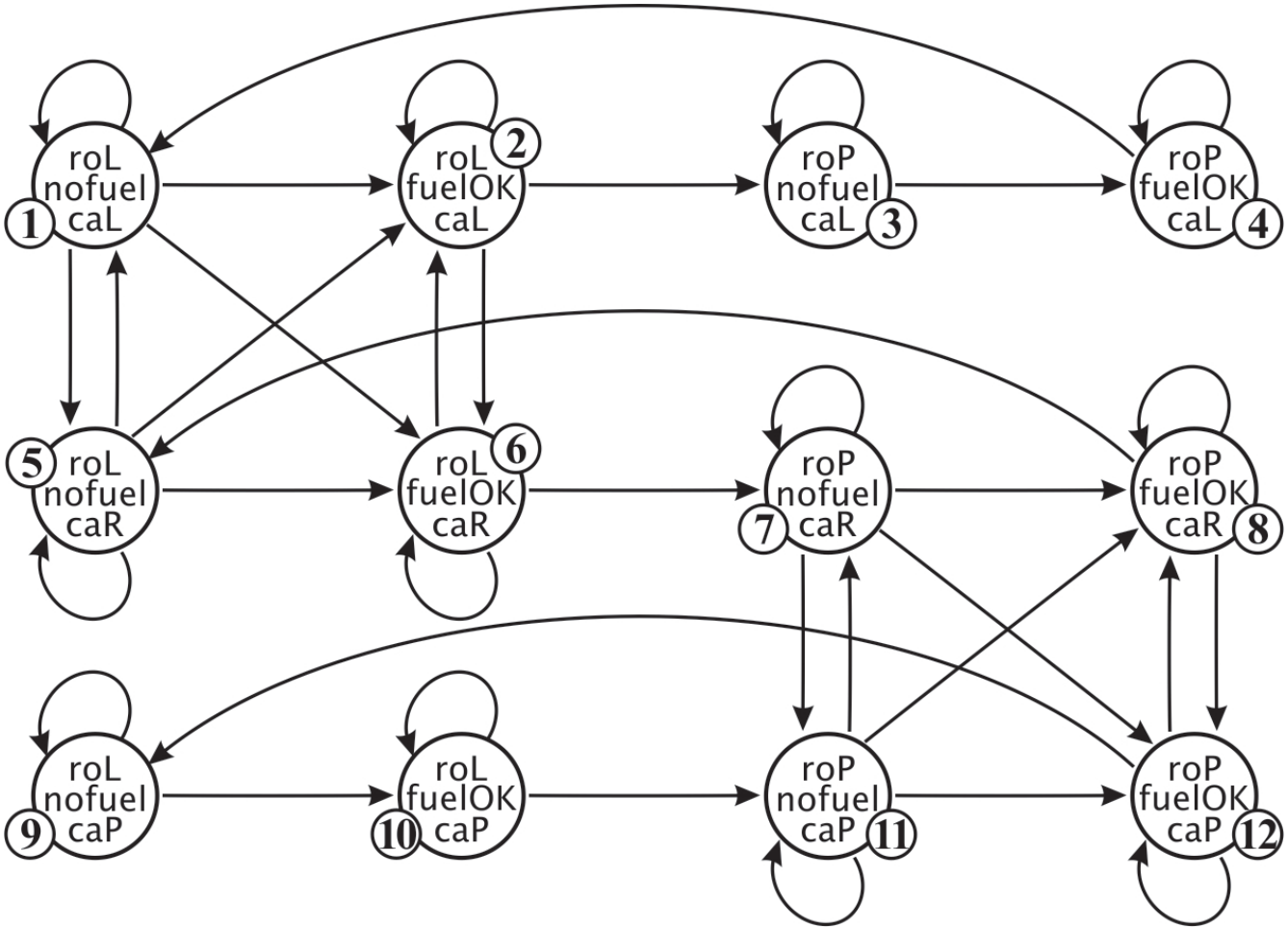
while $Q_2 \not\subseteq Q_1$ **do**

$Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre_\forall(Q_1) \cap [\psi_1]_M$

$[\varphi']_M \leftarrow Q_1$

Example: Simple Rocket

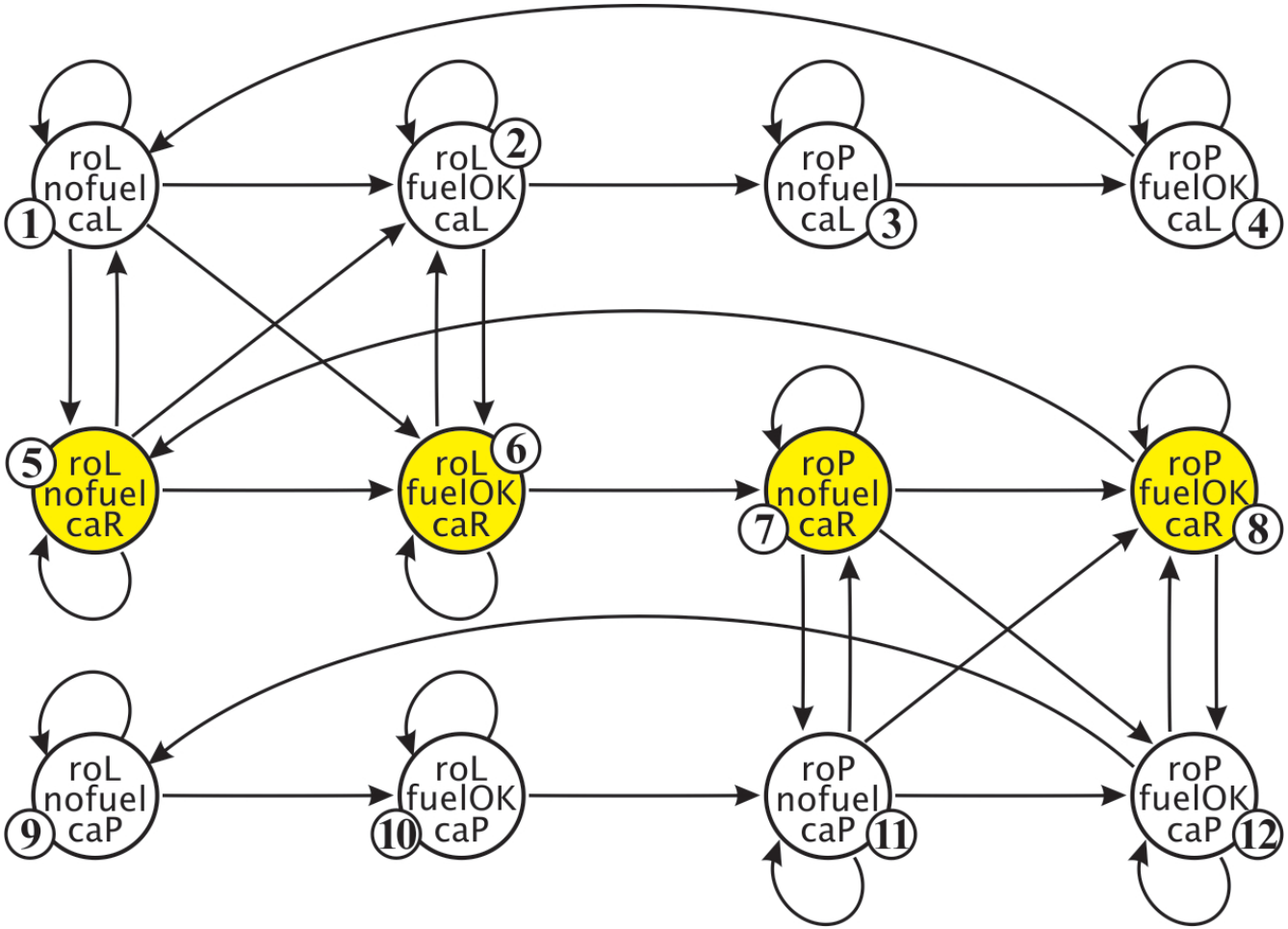
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

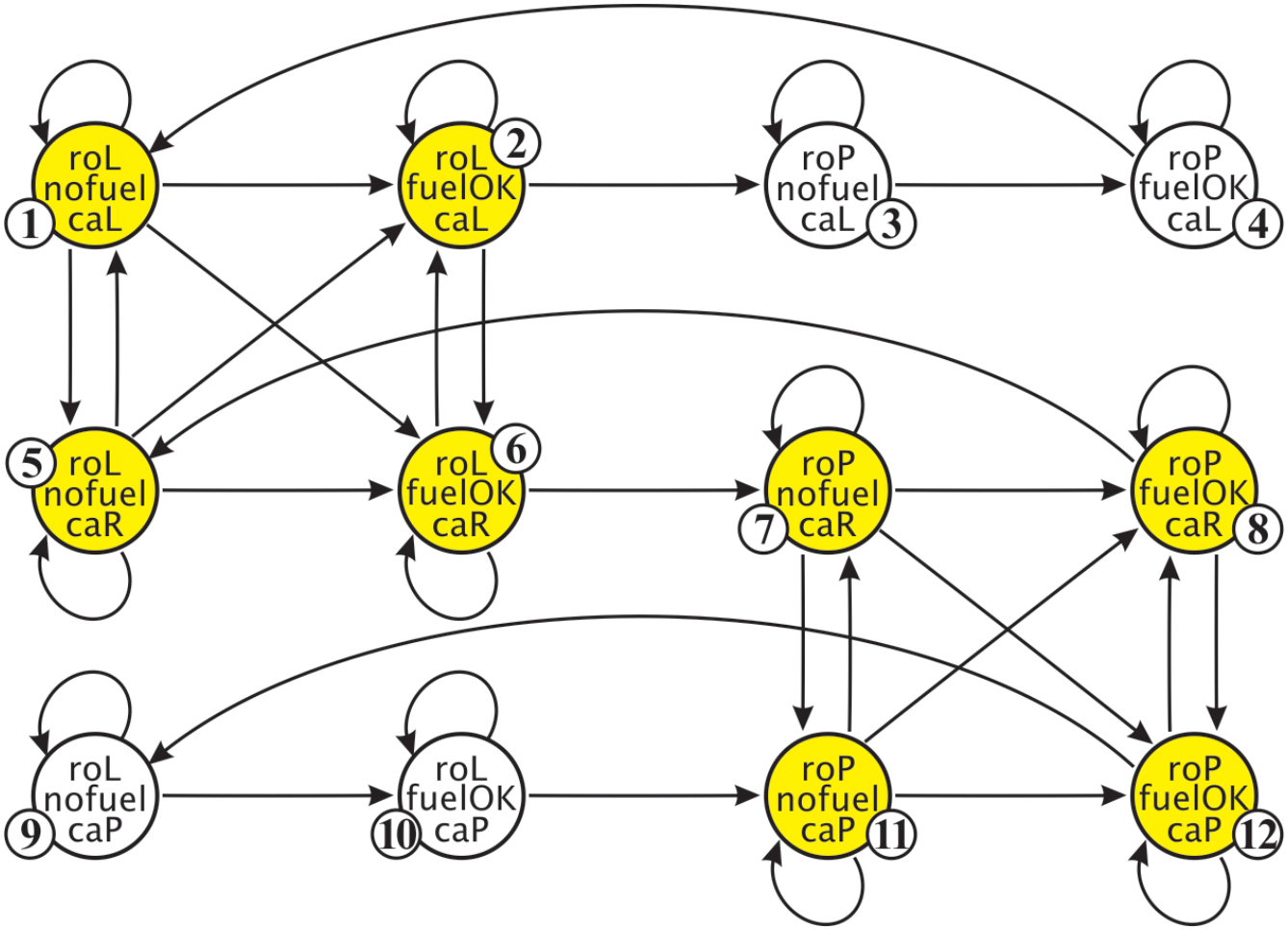
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

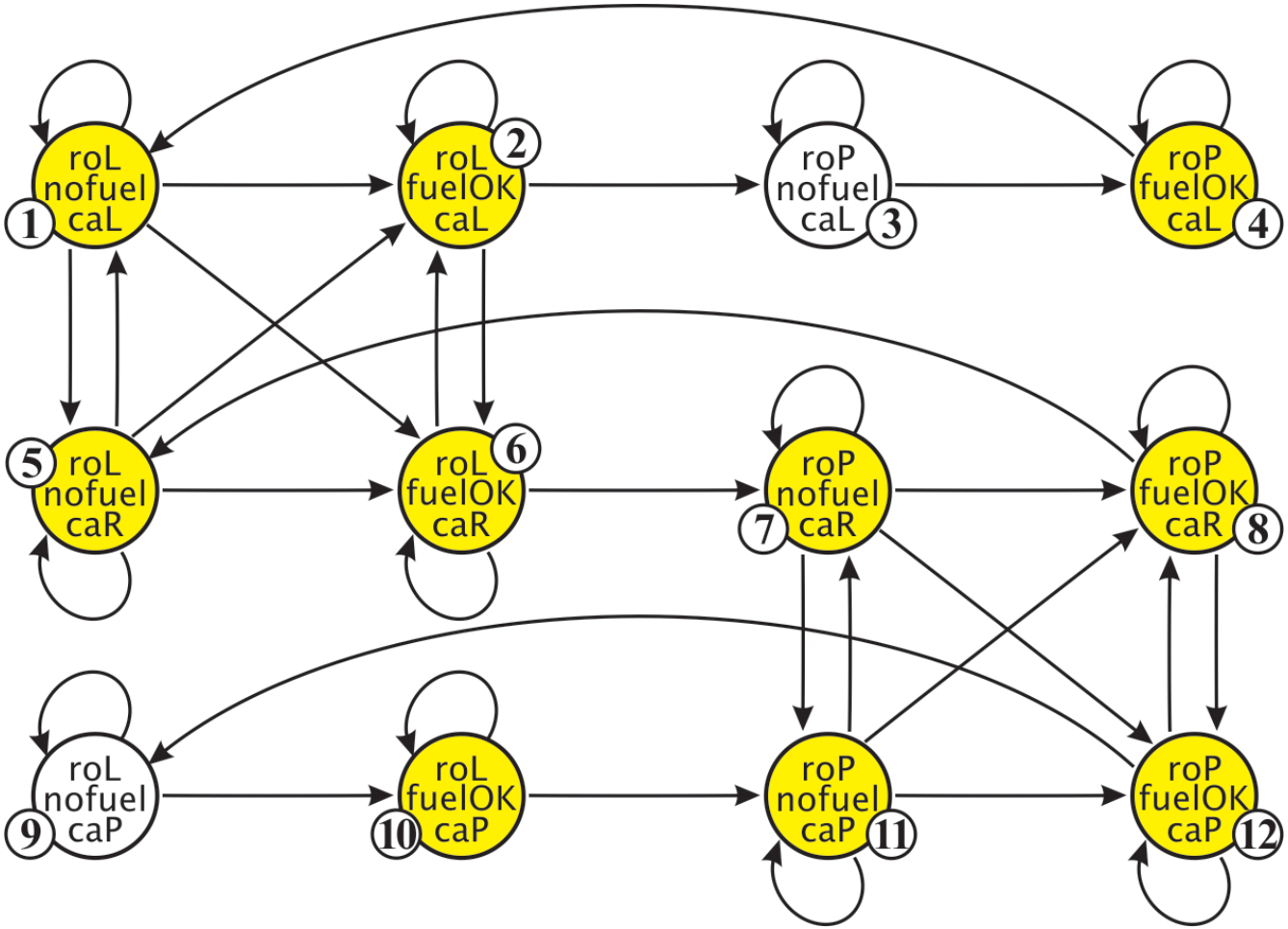
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

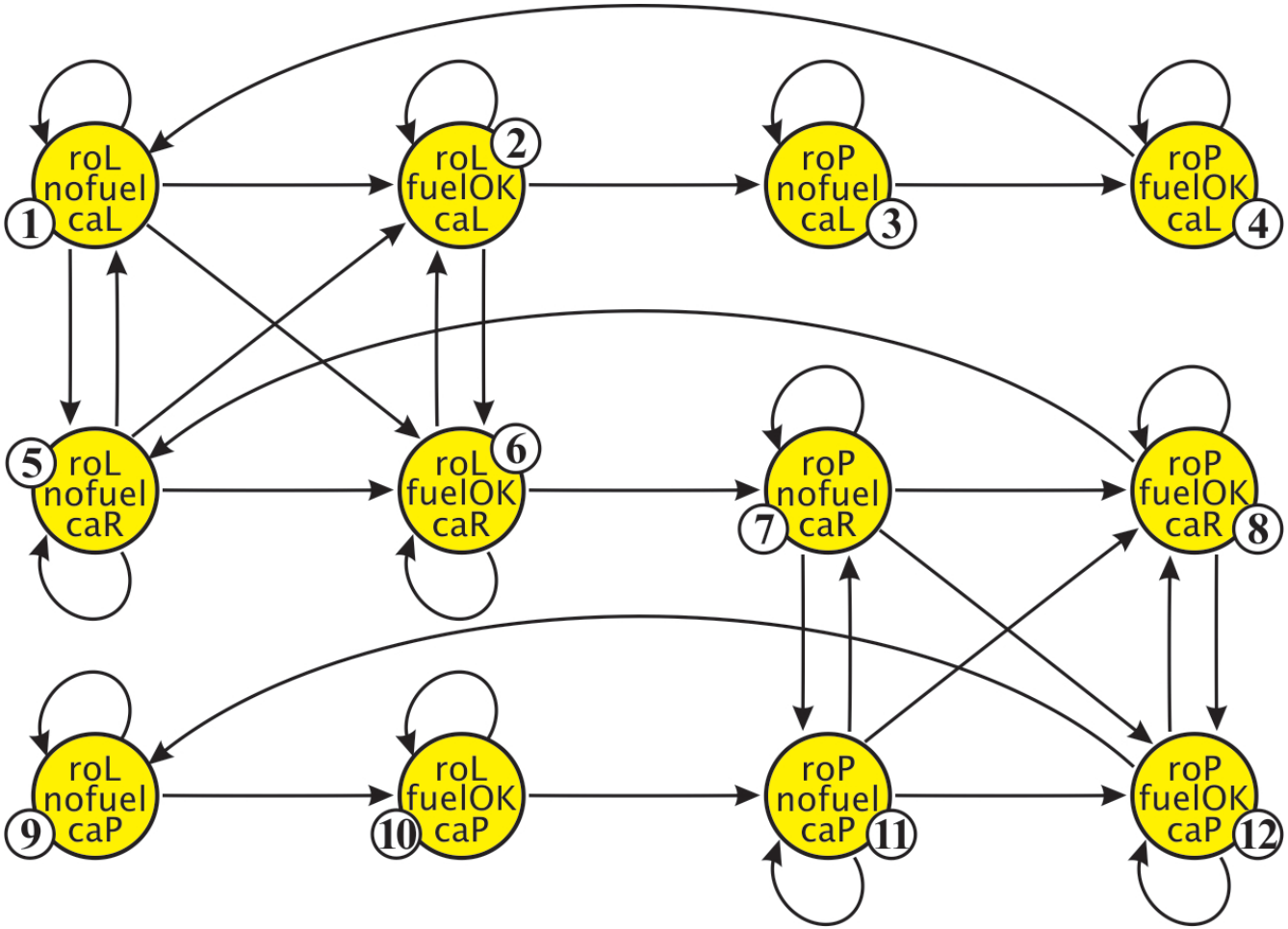
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

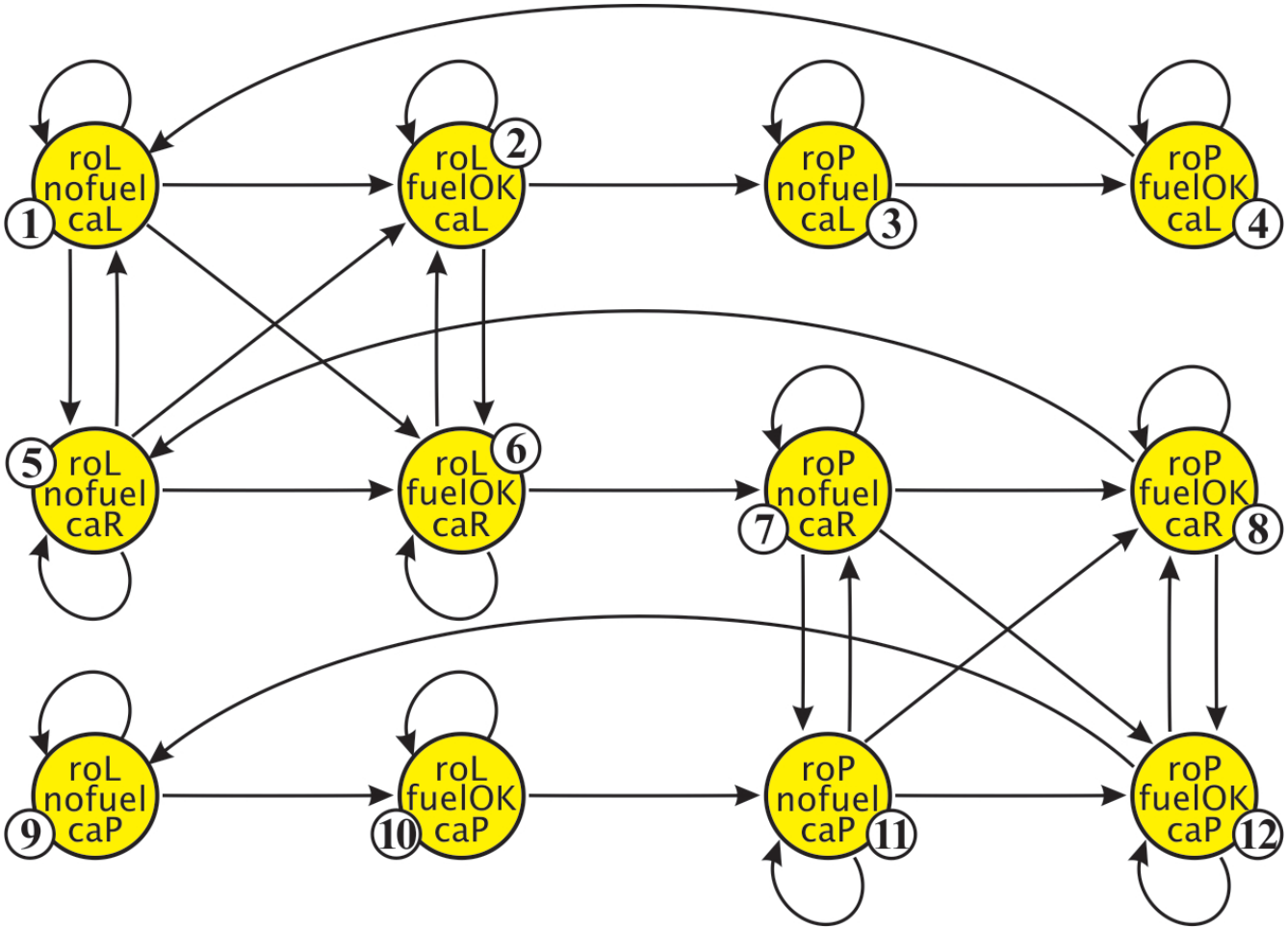
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

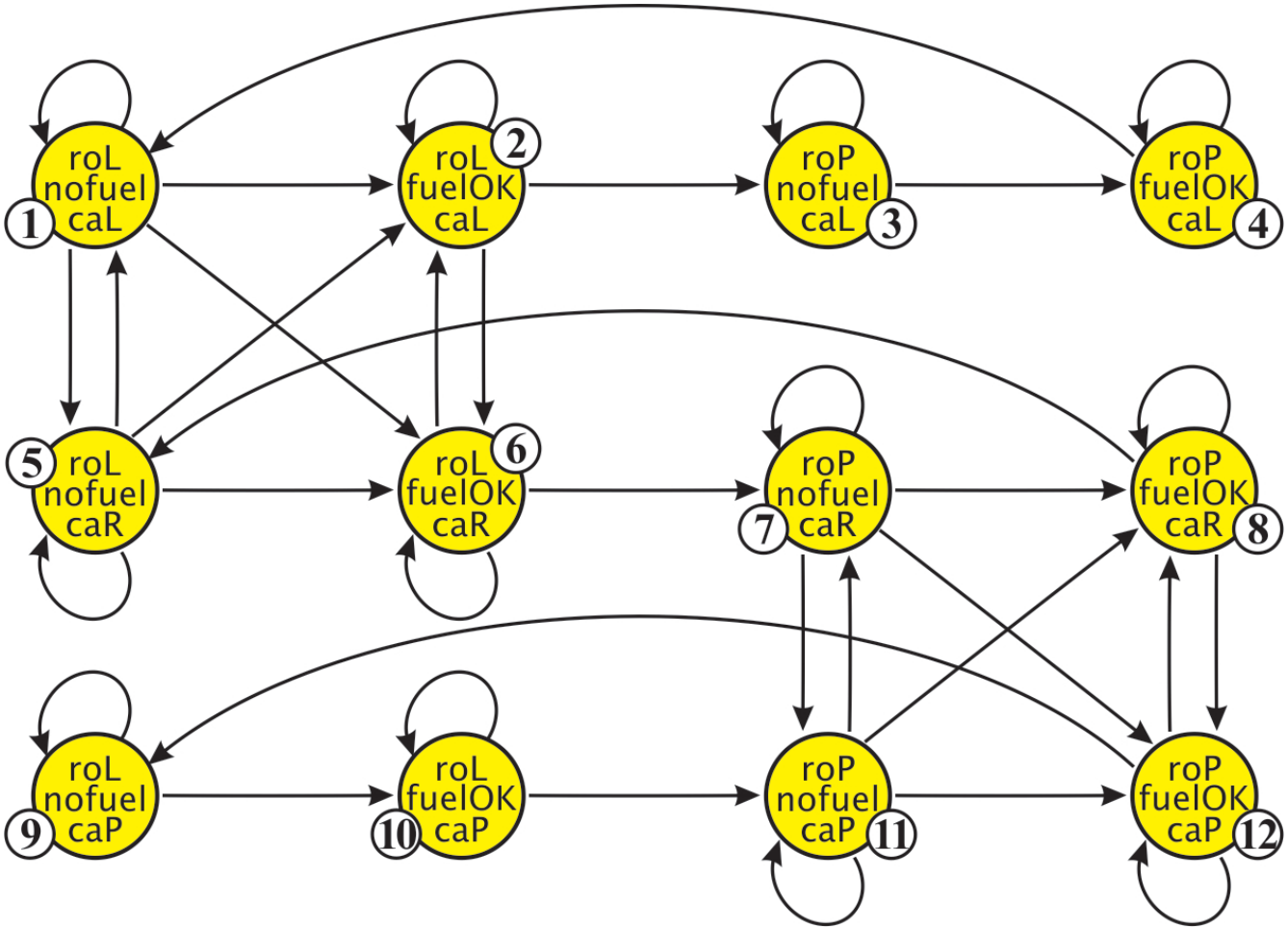
Compute $mcheck(M, EF\,\text{caR})$



$EF\,\text{caR}$

Example: Simple Rocket

Compute $mcheck(M, EF\,\text{caR})$

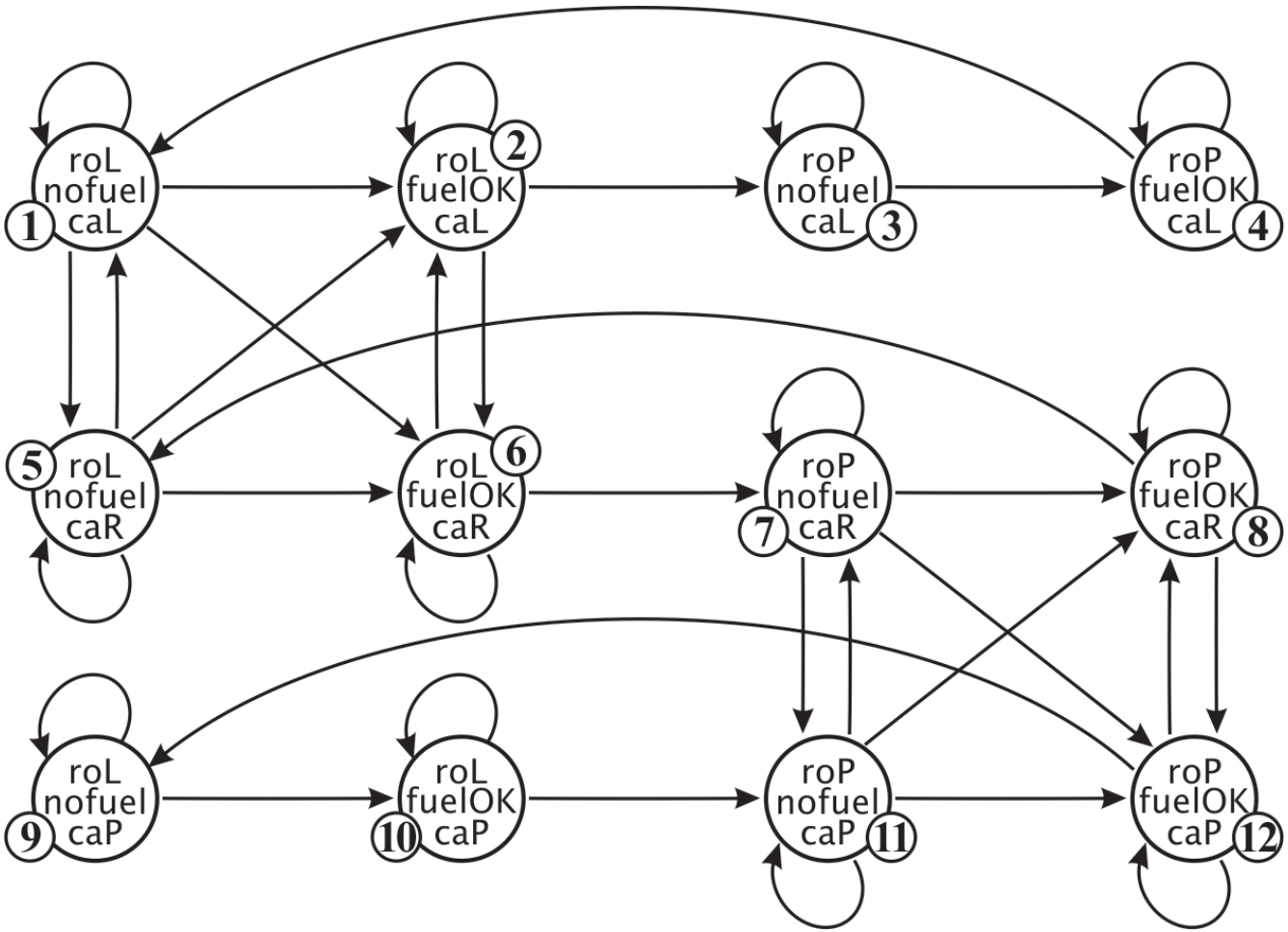


$EF\,\text{caR}$

Done!

Example: Simple Rocket

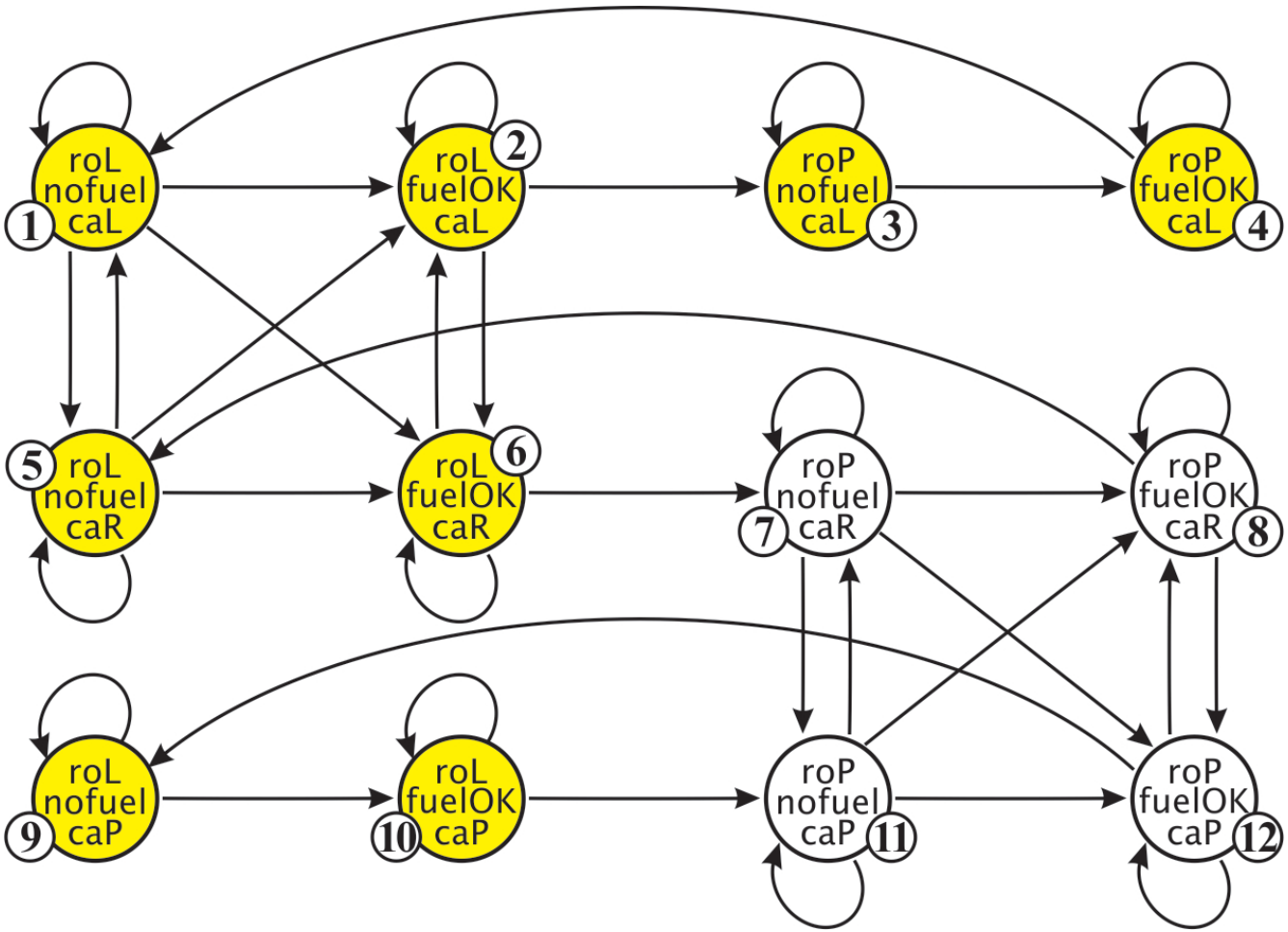
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

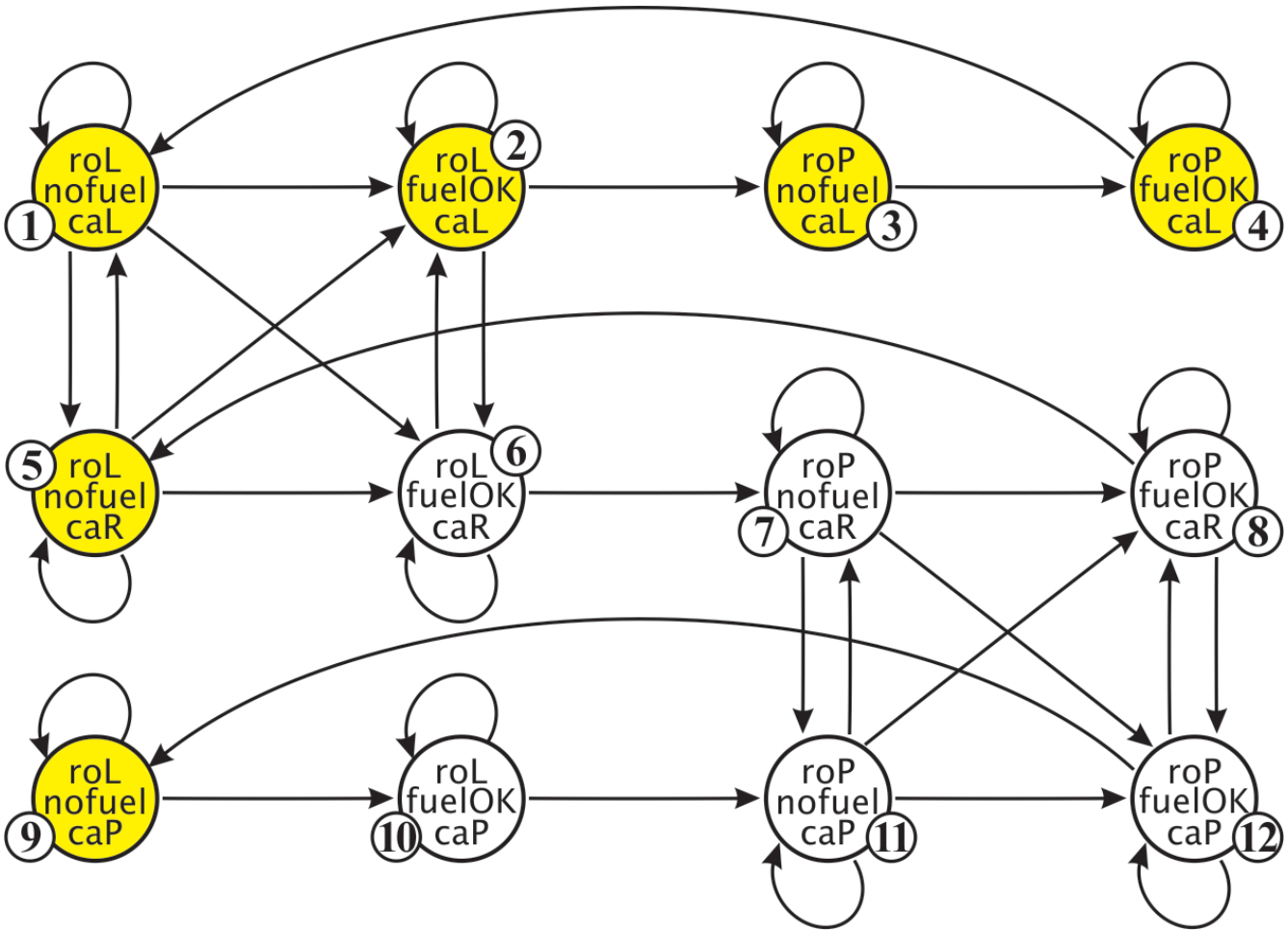
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

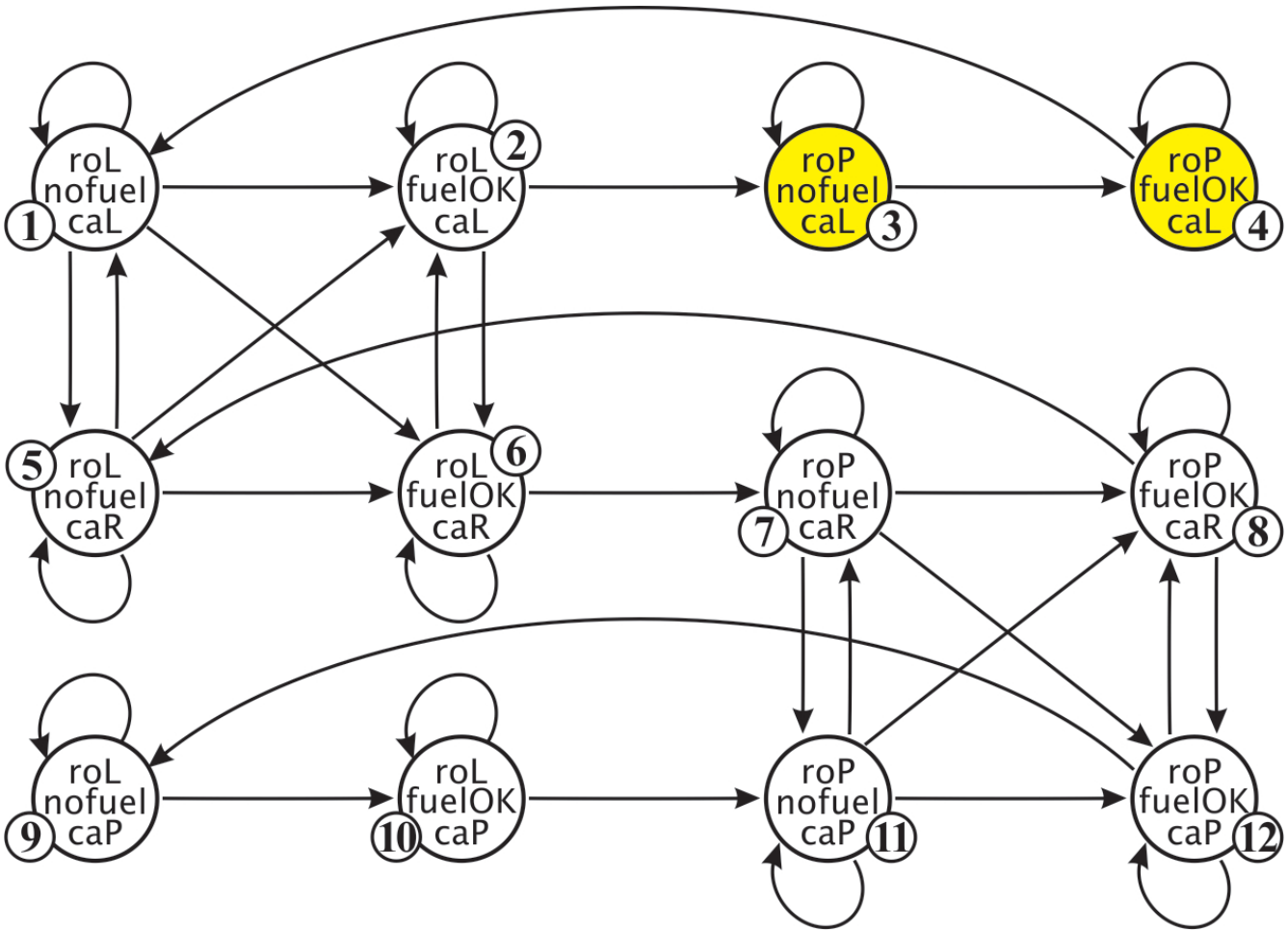
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

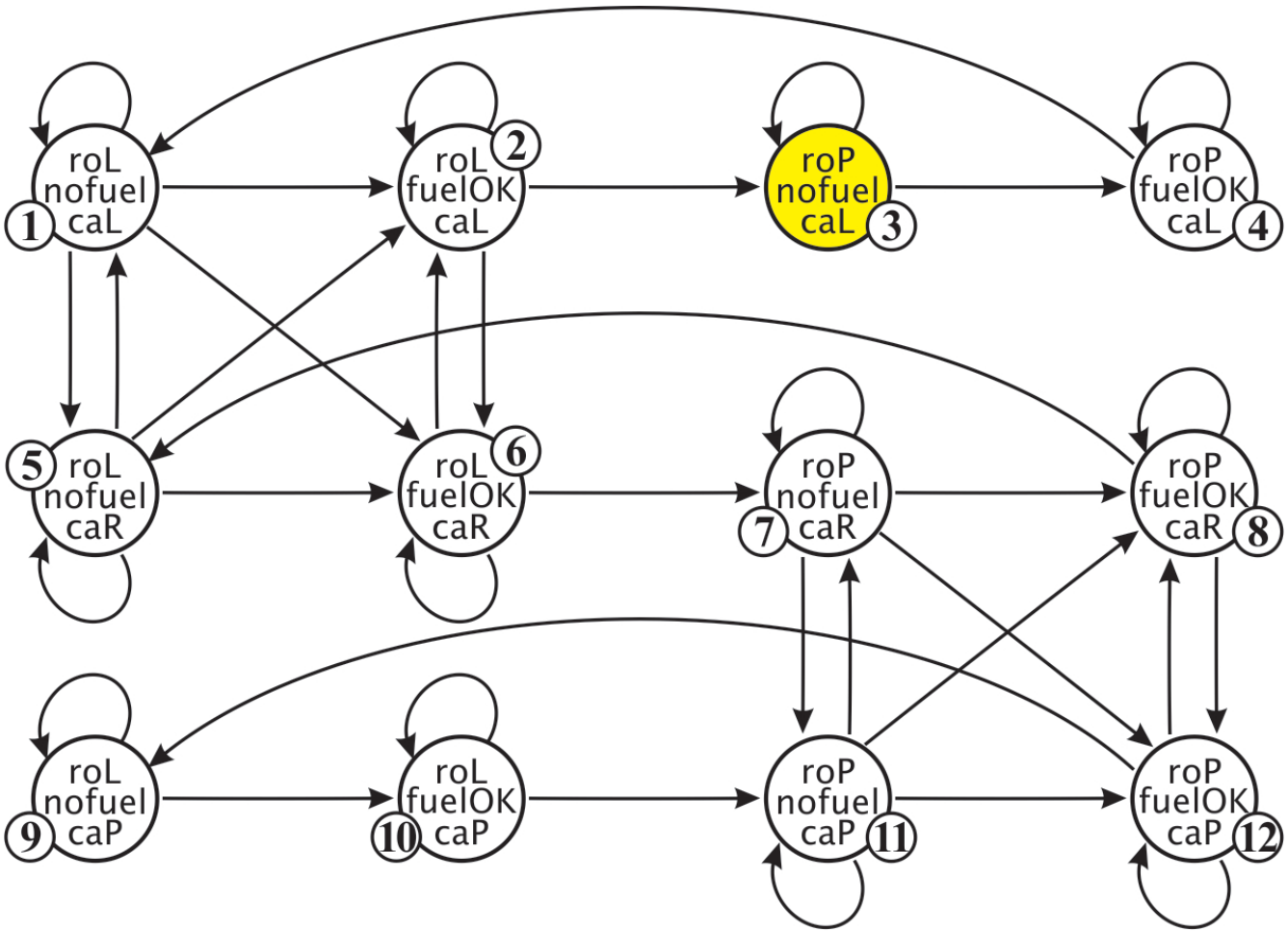
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

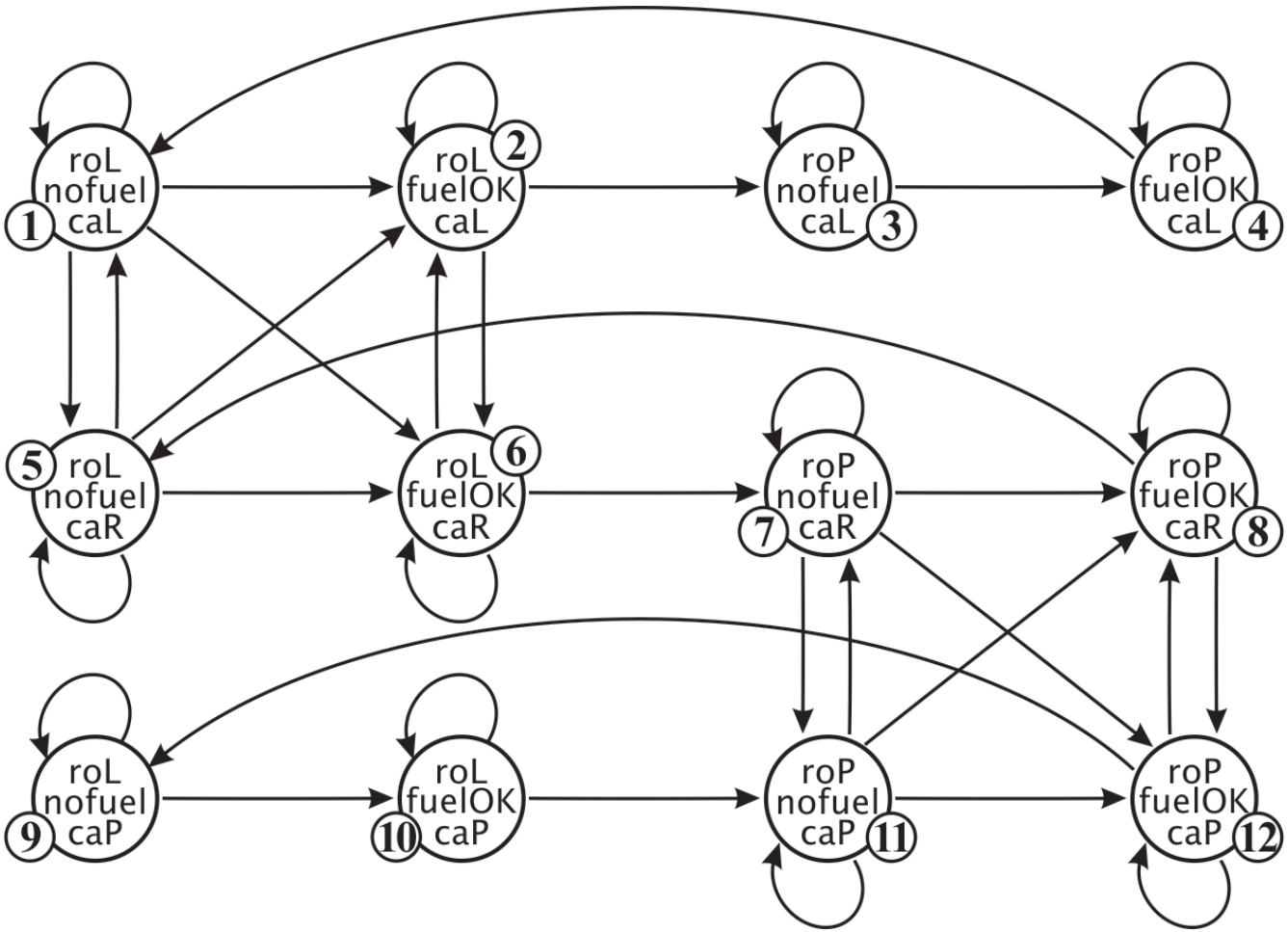
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

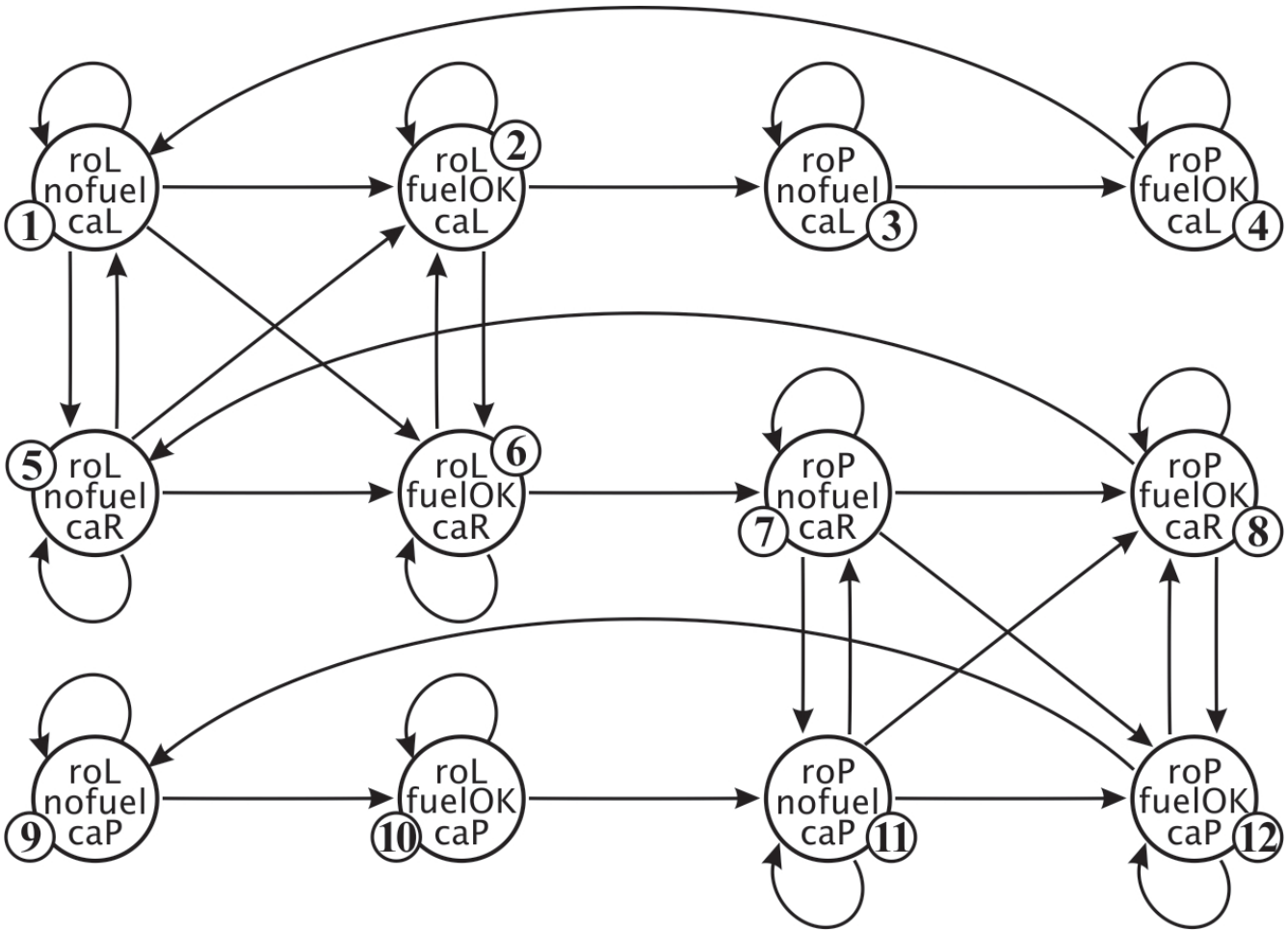
Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Example: Simple Rocket

Compute $mcheck(M, AG(\text{roL} \vee \text{caL}))$



$AG(\text{roL} \vee \text{caL})$

Done!

Complexity of CTL Model Checking

Theorem (Complexity of CTL Model Checking)

Model checking of CTL is P-complete, and can be done in time $O(m \cdot l)$ where m is the number of transitions in the model and the l is the number of subformulae in the formula.