Strategy Composition in Compositional Games

Marcus Gelderie

Lehrstuhl für Informatik 7

RWTH Aachen University

20th September 2013

RWTH AACHEN UNIVERSITY

Synthesis and Game Theory

Synthesis

Algorithmically derive a controller from a specification.

Synthesis and Game Theory

Synthesis

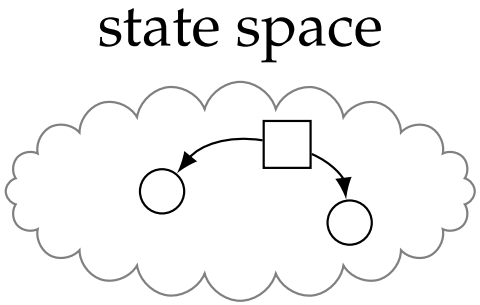
Algorithmically derive a controller from a specification.

Input?

Synthesis

Algorithmically derive a controller from a specification.

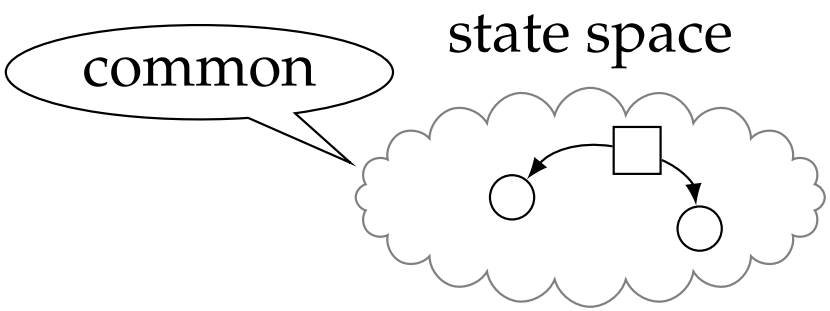
Input?



Synthesis

Algorithmically derive a controller from a specification.

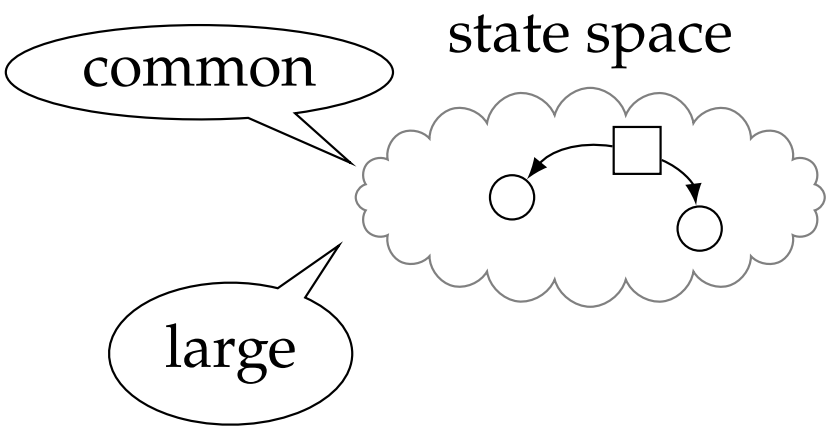
Input?



Synthesis

Algorithmically derive a controller from a specification.

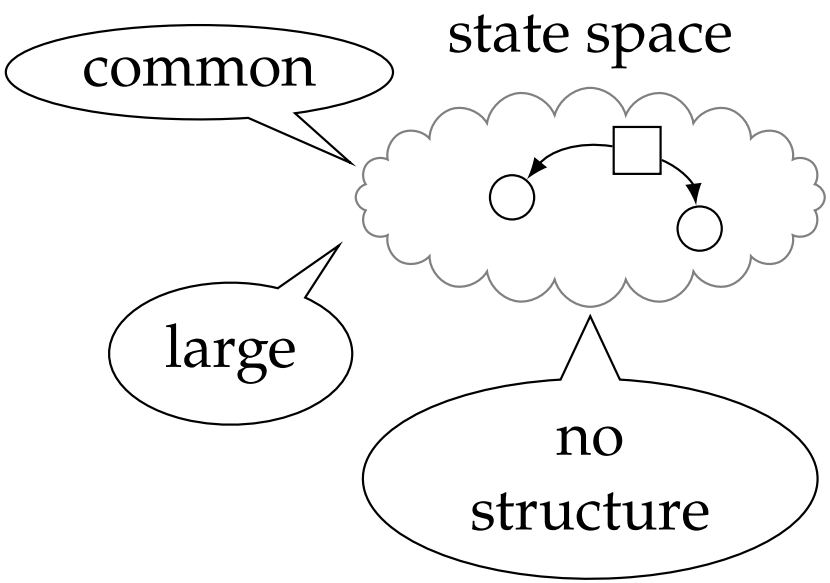
Input?



Synthesis

Algorithmically derive a controller from a specification.

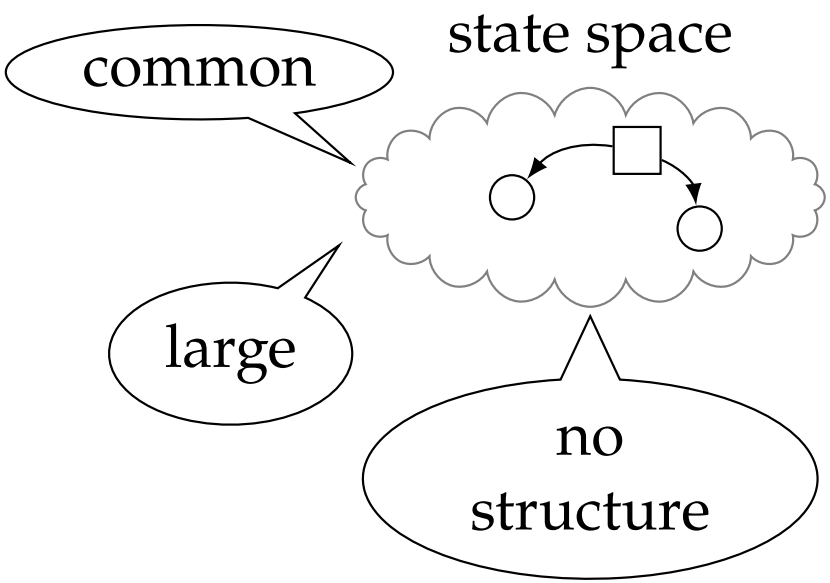
Input?

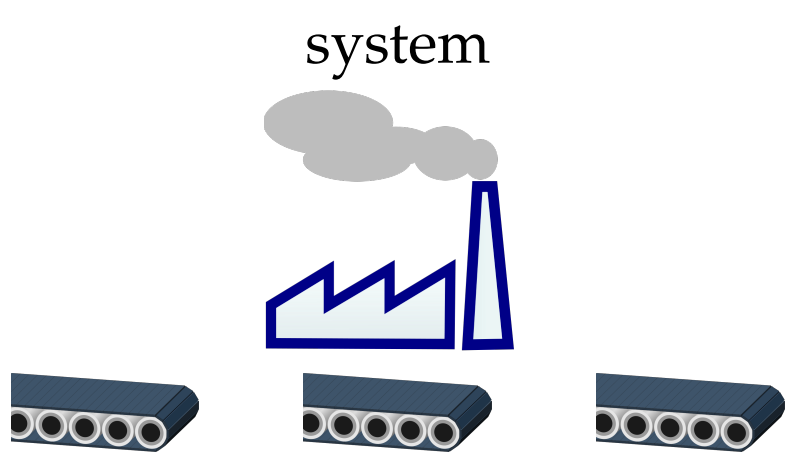


Synthesis

Algorithmically derive a controller from a specification.

Input?

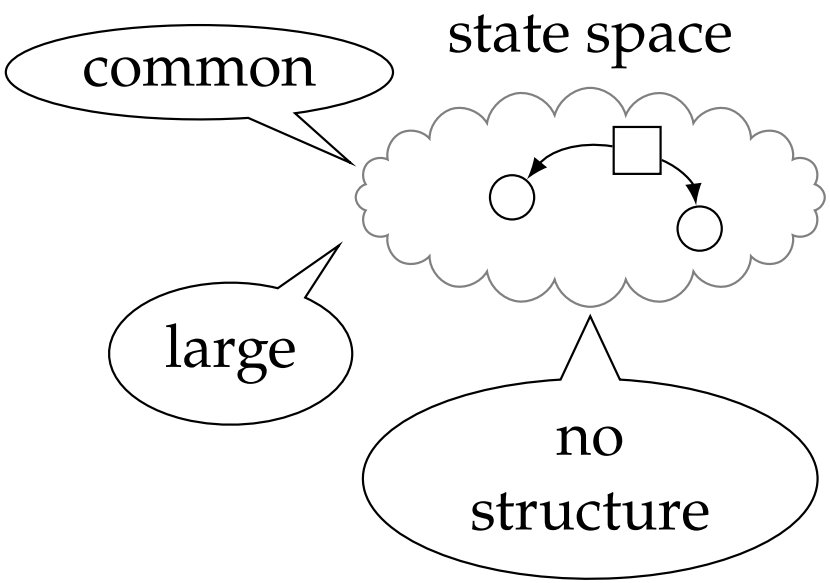


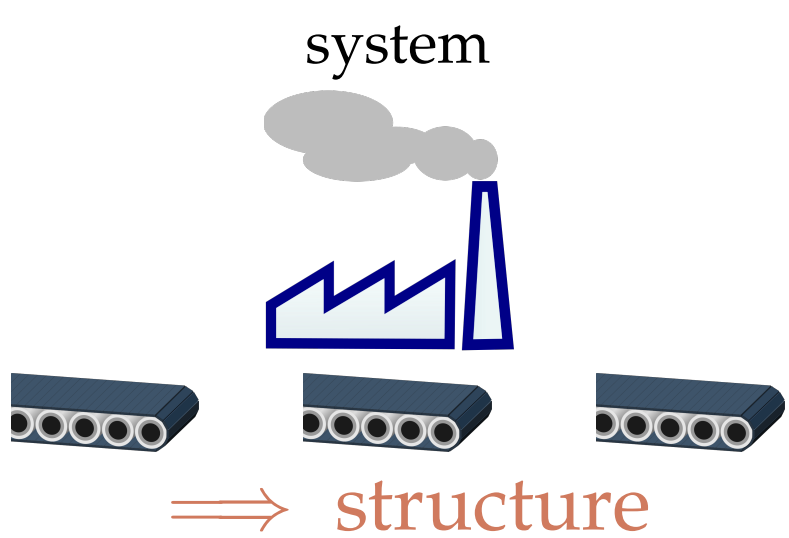


Synthesis

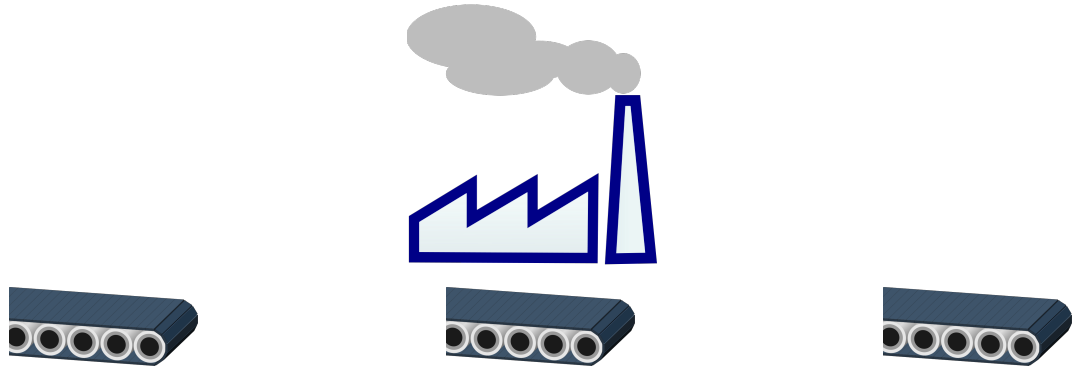
Algorithmically derive a controller from a specification.

Input?

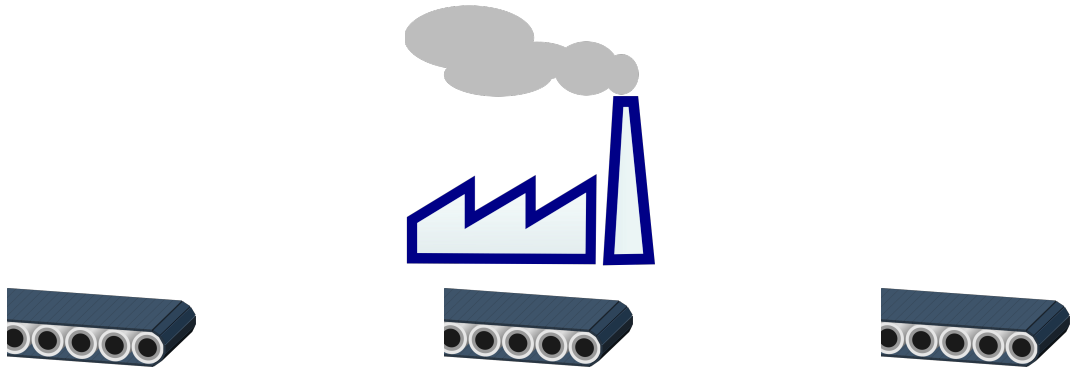


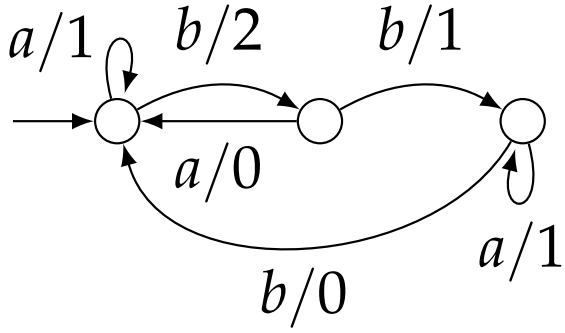


Controller Synthesis

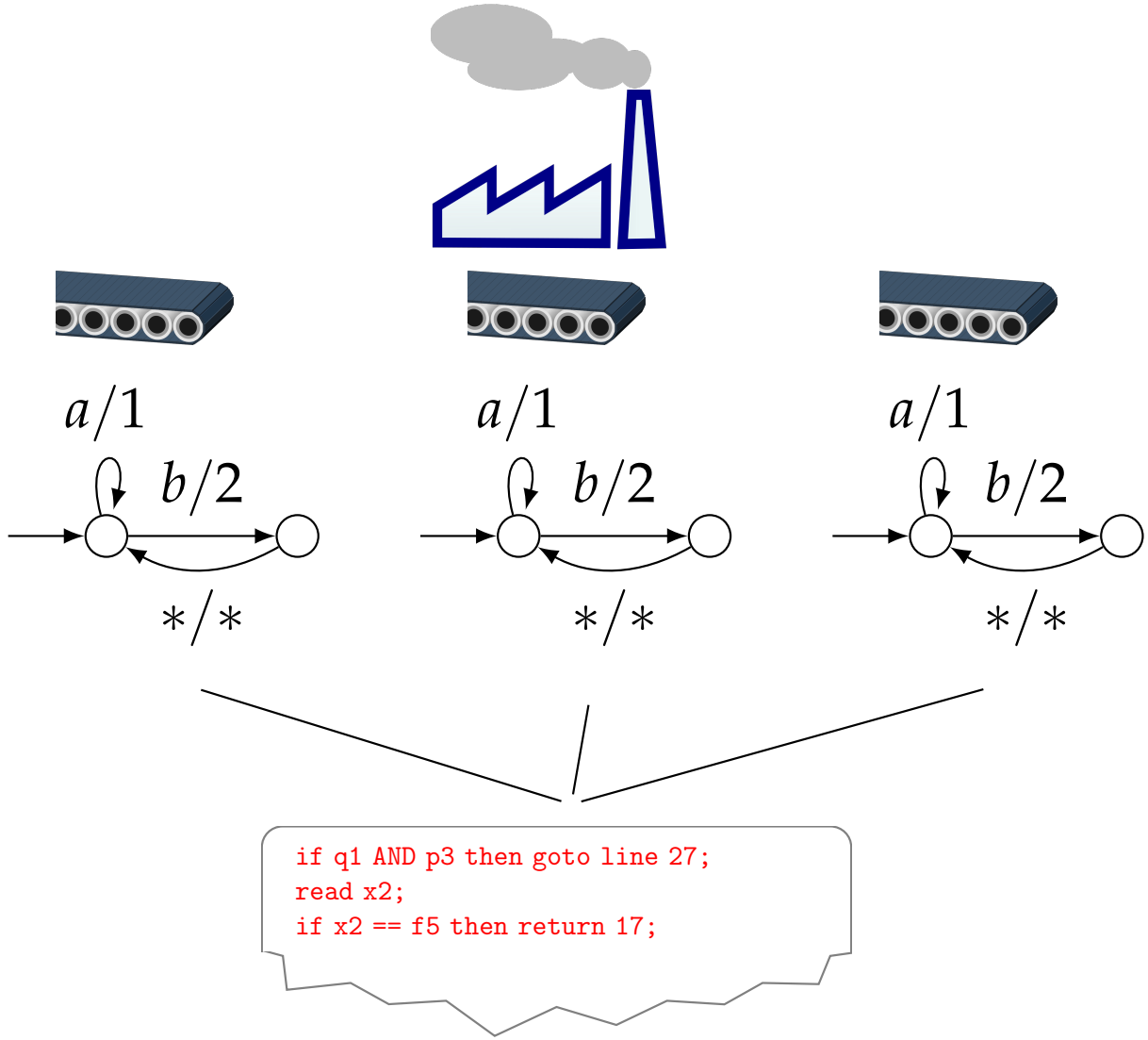


Controller Synthesis

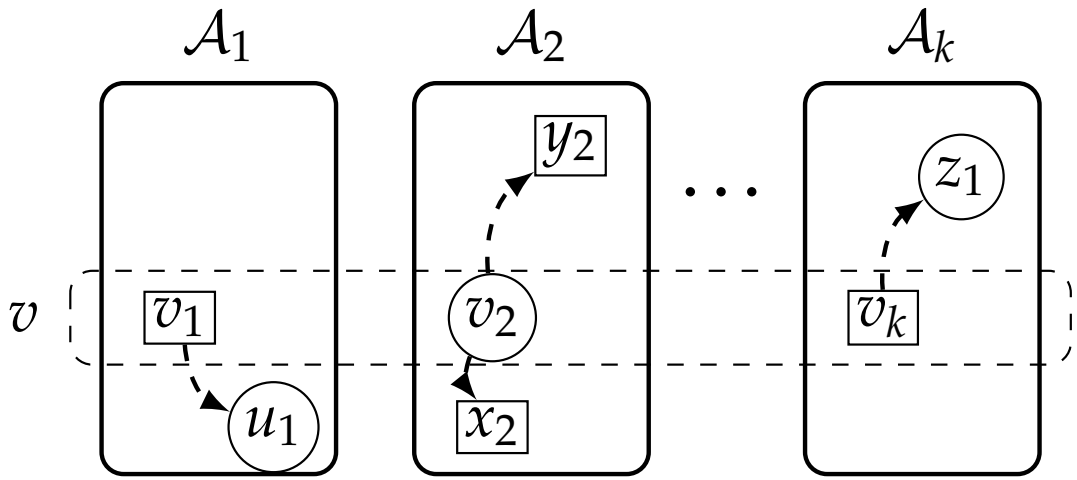




Controller Synthesis

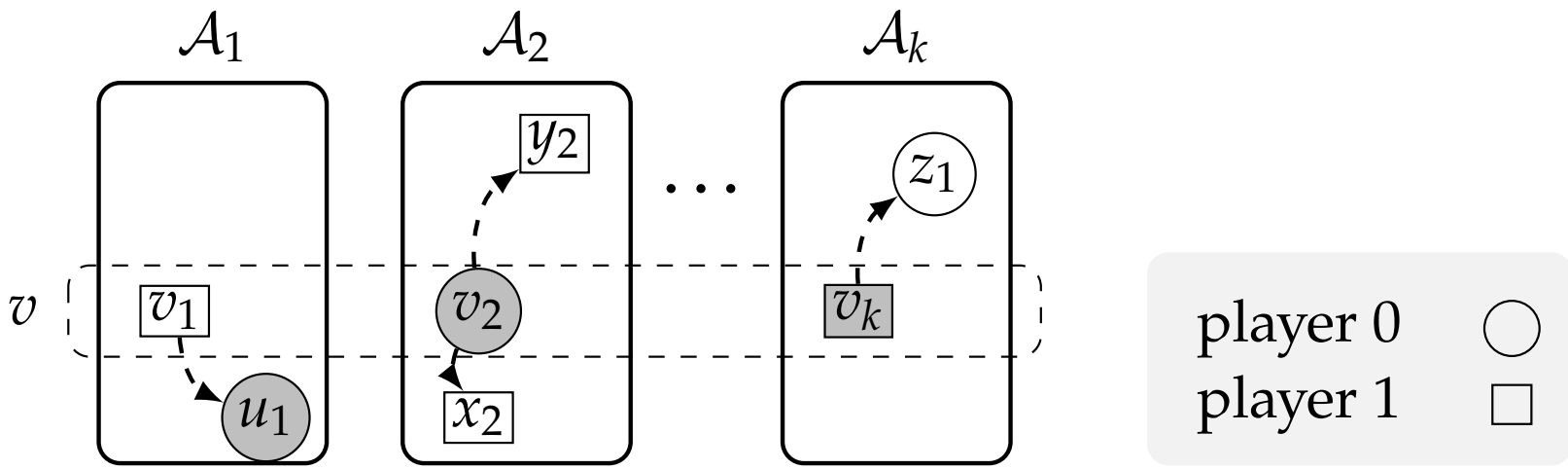


Parallel Product

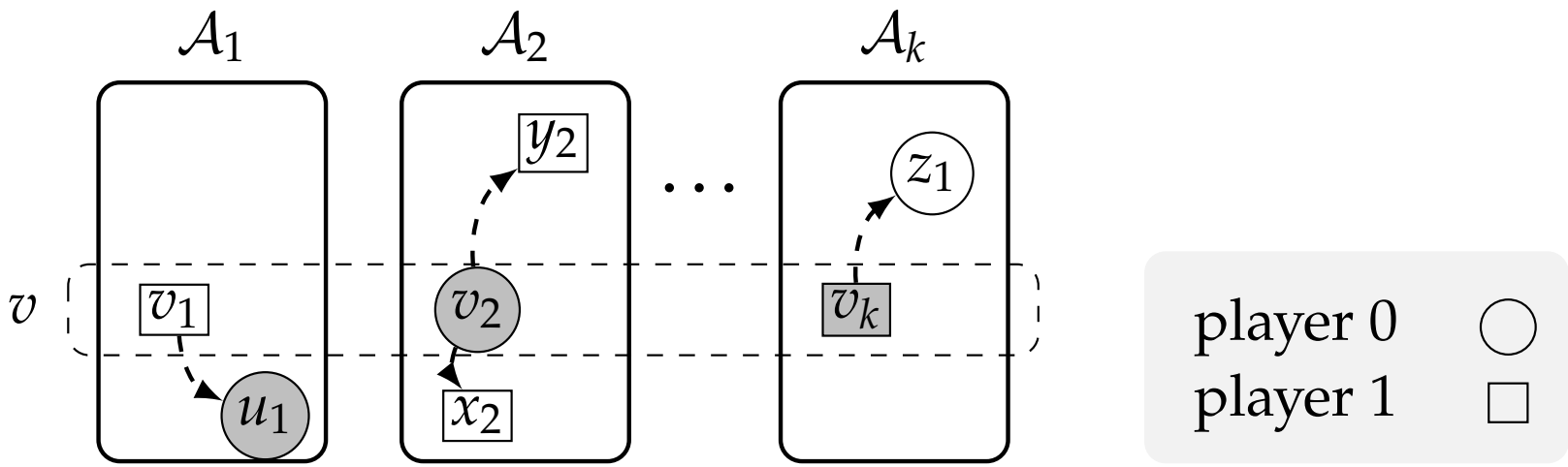


player 0 ○
player 1 □

Parallel Product

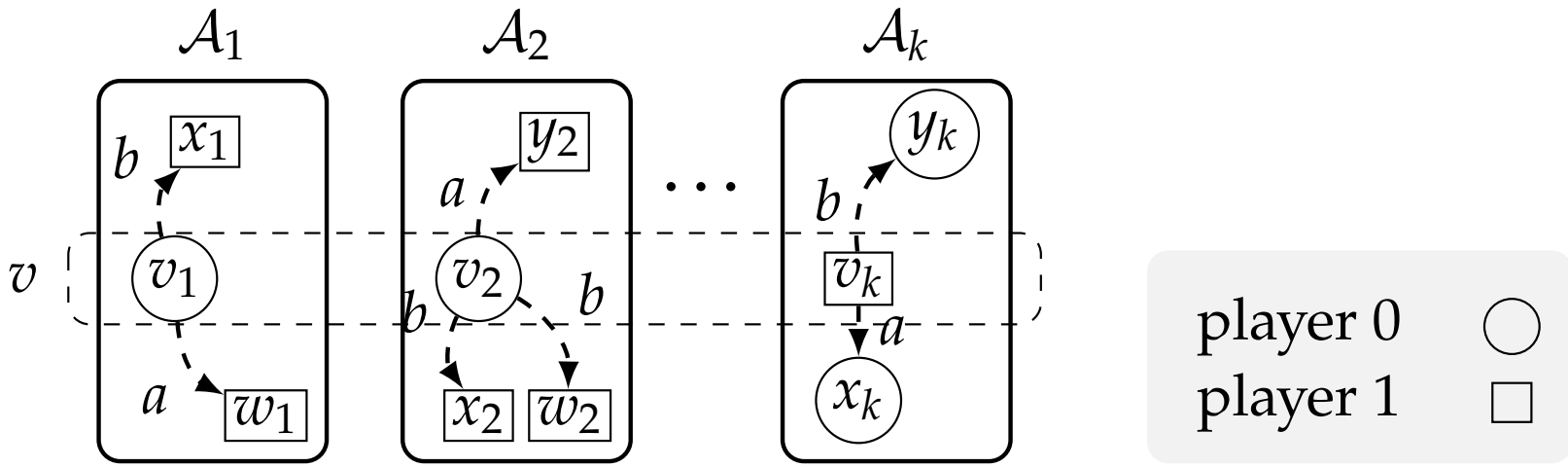


Parallel Product

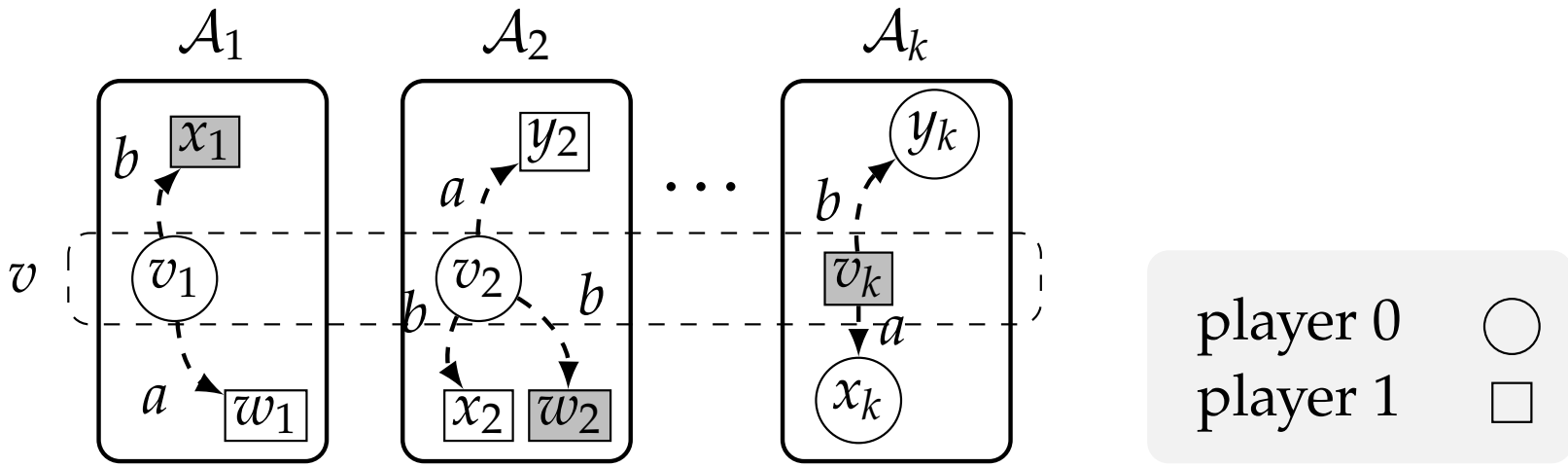


- player p picks one component i in a p-vertex
- player p picks a neighbour of v_i in $\mathcal{A}_i$

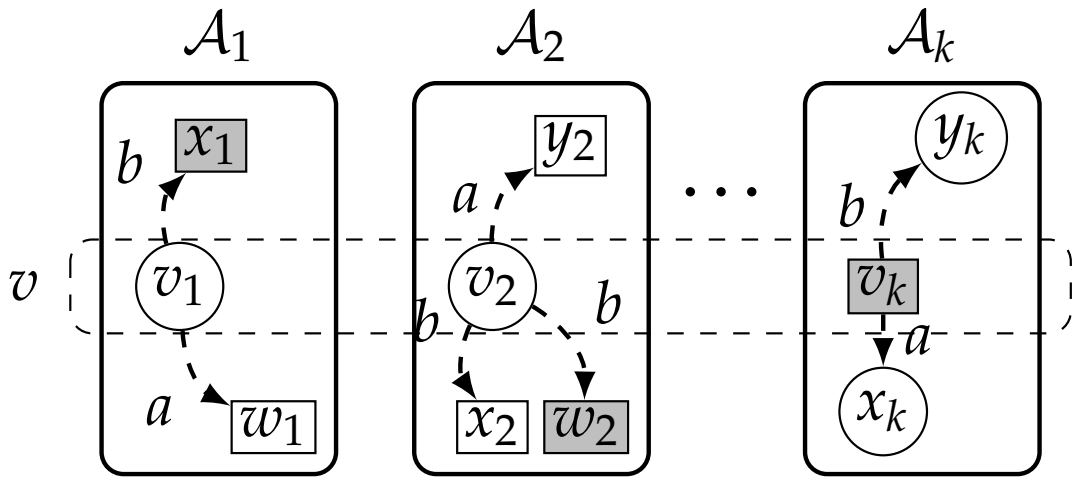
Synchronized Product



Synchronized Product



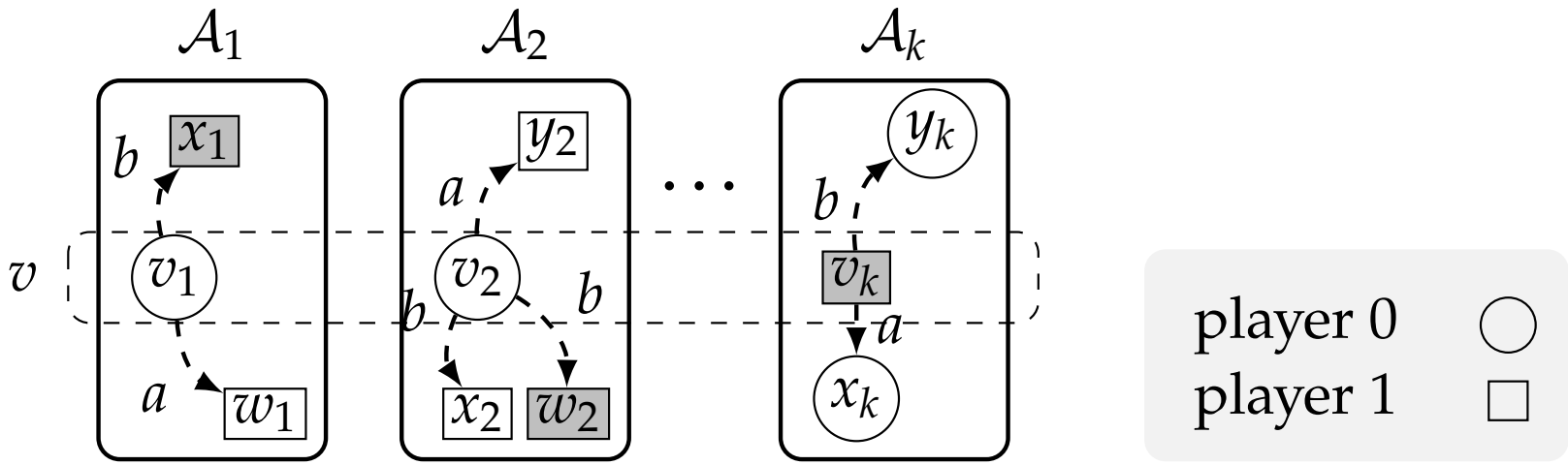
Synchronized Product



player 0 ◯
player 1 □

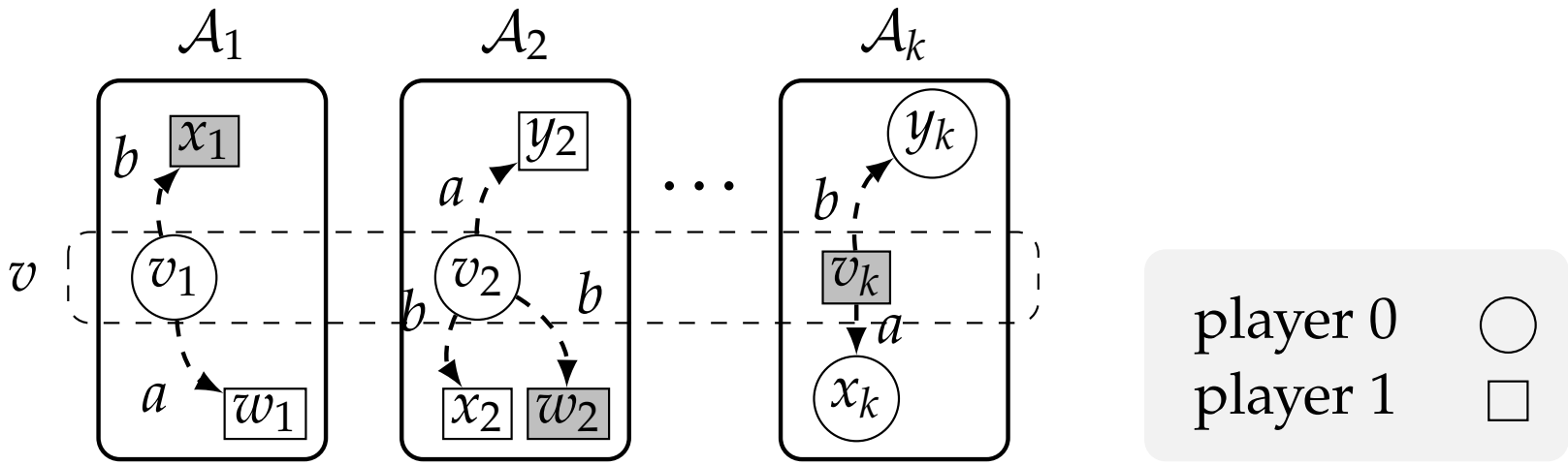
- labelled arenas $\mathcal{A}_i = (V_i, \Delta_i, \Sigma_i)$

Synchronized Product



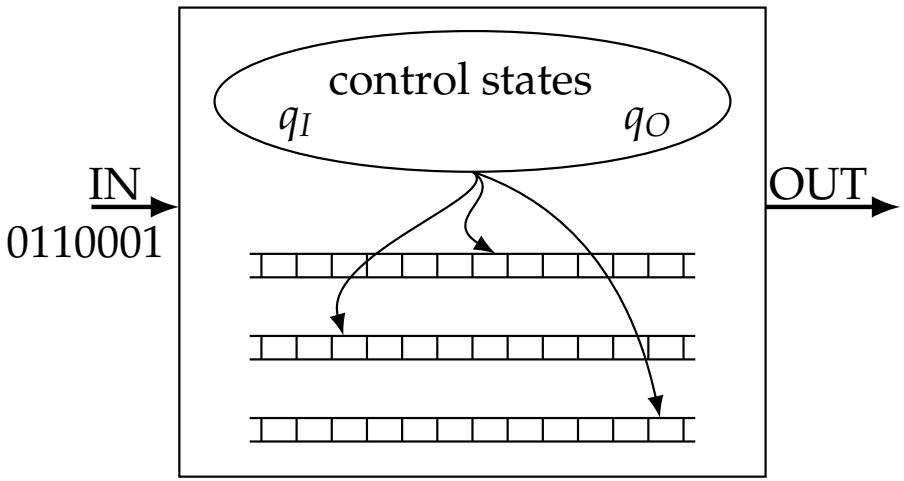
- labelled arenas $\mathcal{A}_i = (V_i, \Delta_i, \Sigma_i)$
- player p picks letter b

Synchronized Product



- labelled arenas $\mathcal{A}_i = (V_i, \Delta_i, \Sigma_i)$
- player p picks letter b
- player p picks b-transition in all components i where both
  - $v_i \in V_i^{(p)}$
  - and $b \in \Sigma_i$

Strategy Machines

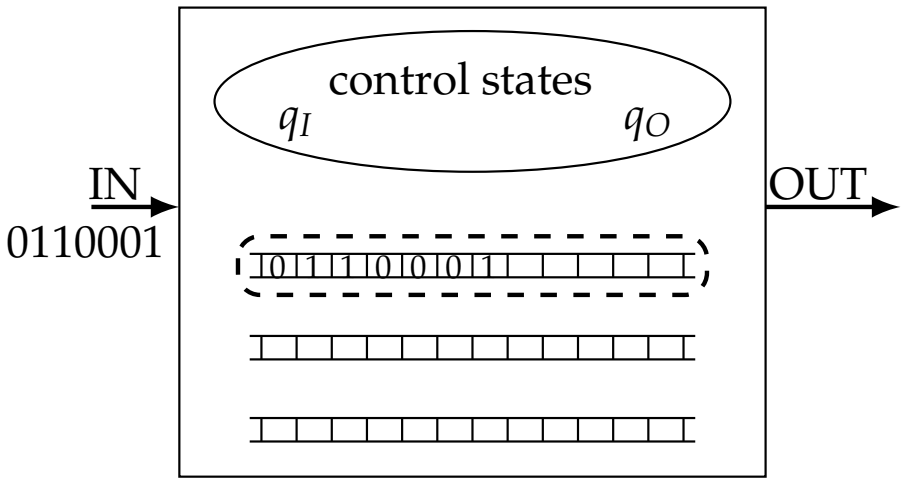


Definition (Strategy Machine)

3-tape Turing machine with

- input state q_I
- output state q_O

Strategy Machines

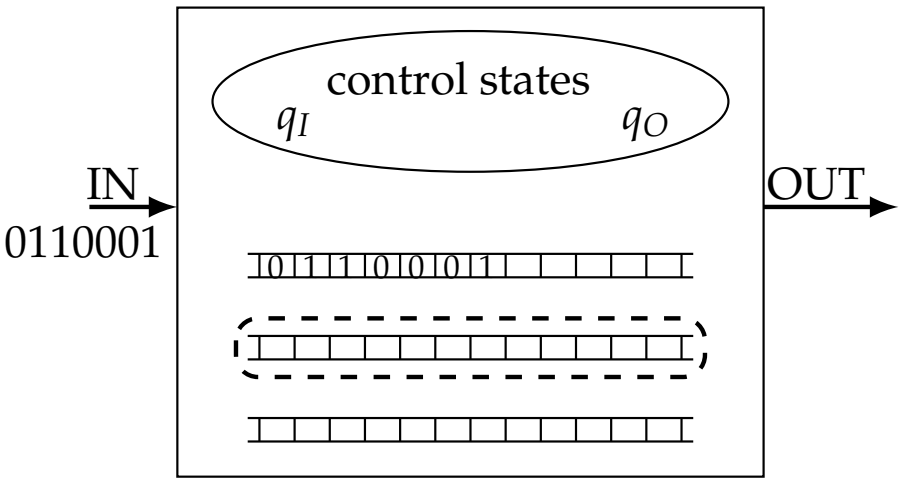


Definition (Strategy Machine)

3-tape Turing machine with

- input state q_I
- output state q_O
- IO-tape

Strategy Machines

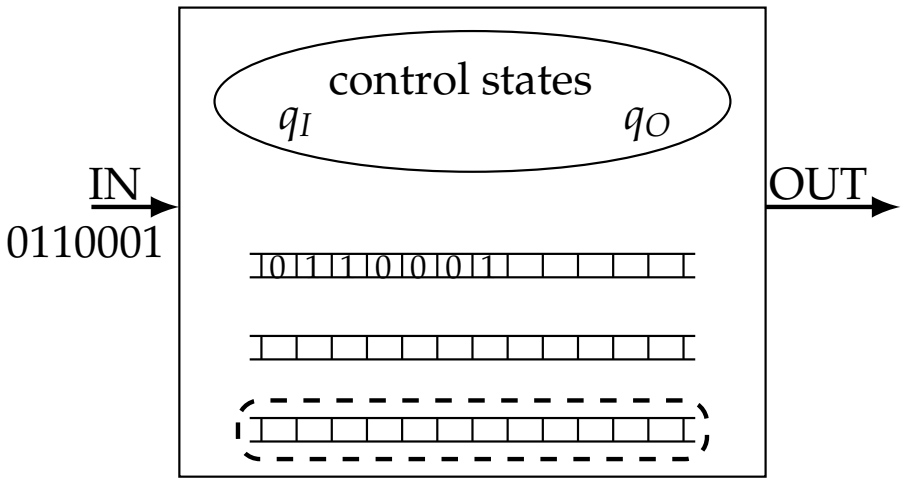


Definition (Strategy Machine)

3-tape Turing machine with

- input state q_I
- output state q_O
- IO-tape
- memory tape

Strategy Machines

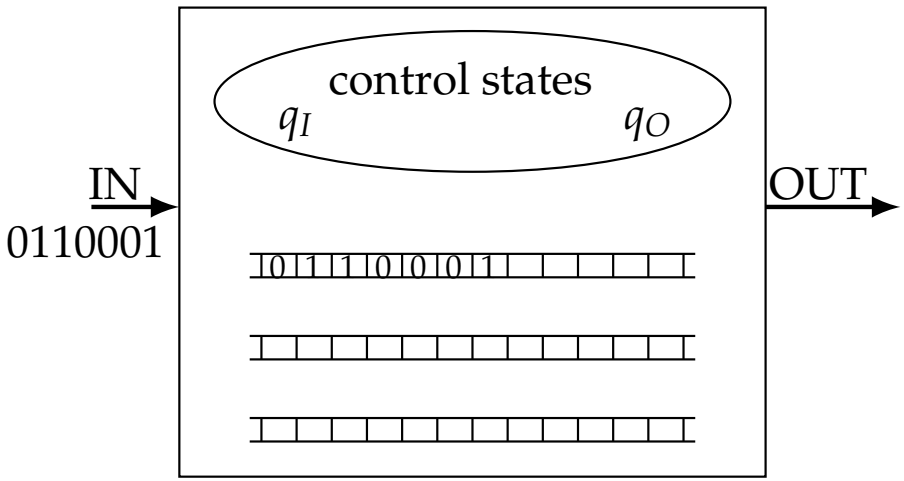


Definition (Strategy Machine)

3-tape Turing machine with

- input state q_I
- output state q_O
- IO-tape
- memory tape
- computation tape

Strategy Machines



Definition (Strategy Machine)

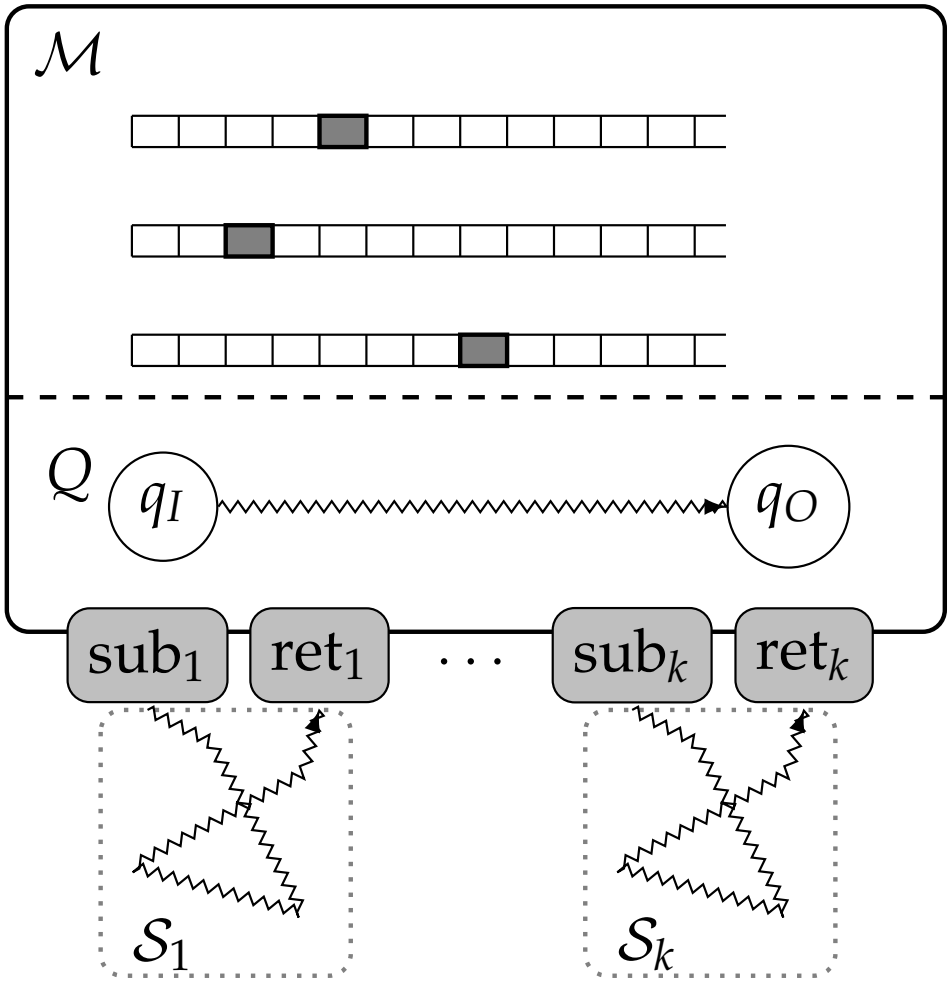
3-tape Turing machine with

- input state q_I
- output state q_O
- IO-tape
- memory tape
- computation tape

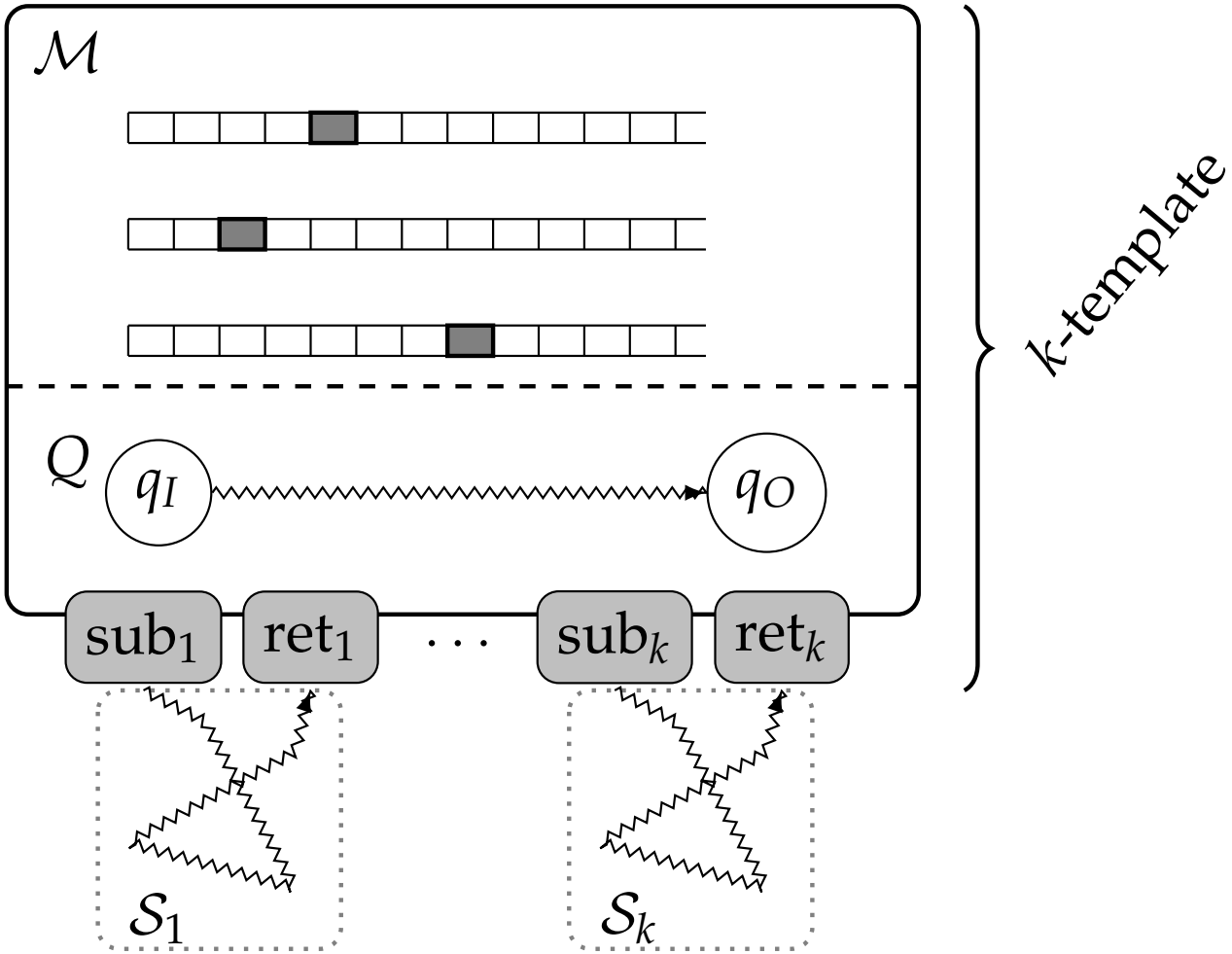
Definition

- size = $|Q|$ number of control states
- latency = length of longest iteration $q_I \rightsquigarrow q_O$
- space requirement = $\max_{\text{iterations}} \sum_{t \in \text{tapes}} \#$ cells used on t

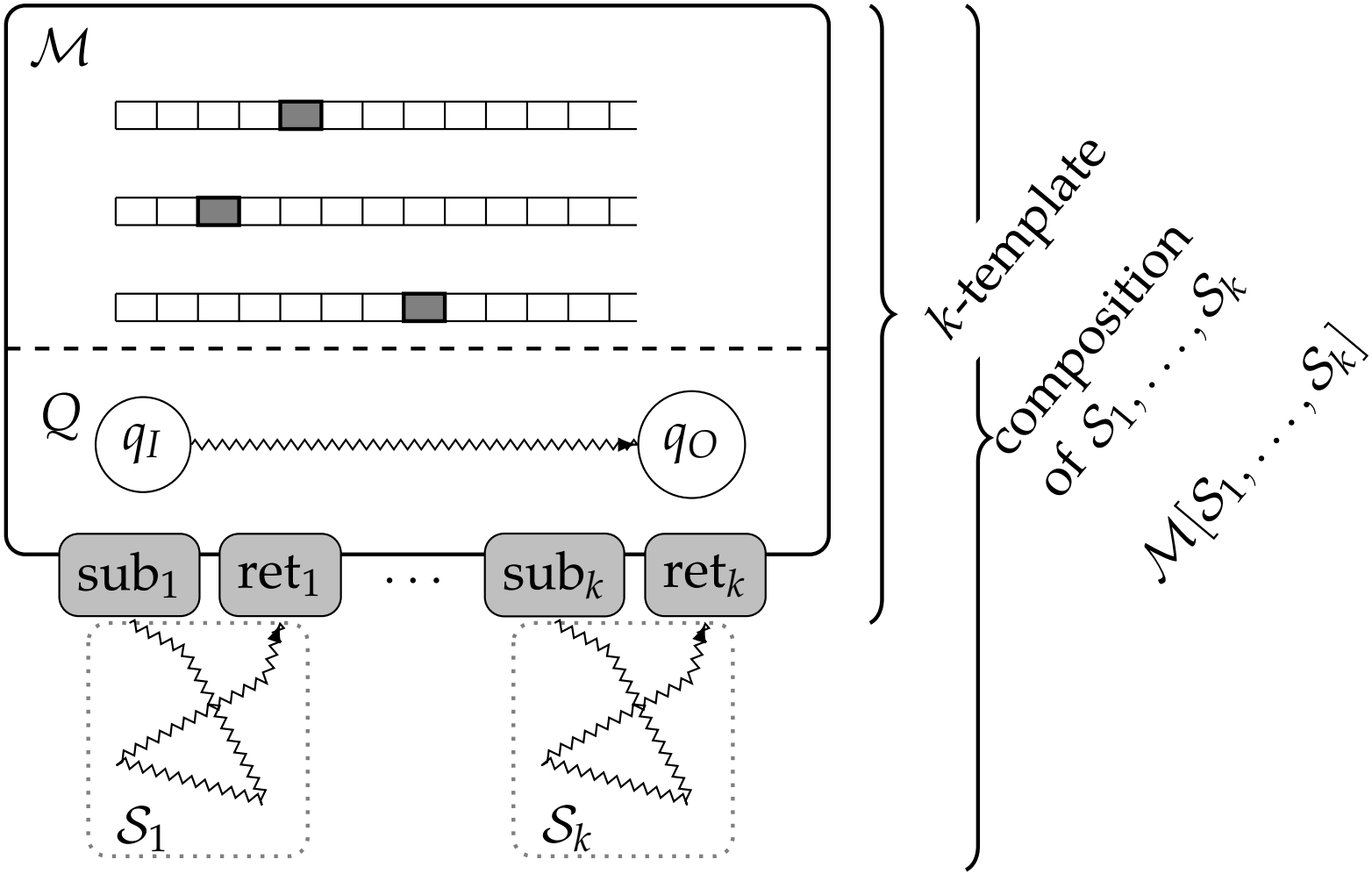
Compositions



Compositions



Compositions



Reachability Games

Theorem

Local and synchronized reachability games on parallel products admit polynomial time computable polynomial compositions.
The component games of the composition are positionally determined.

Theorem

Local or synchronized reachability games on synchronized products admit poly-space compositions if and only if Pspace = Exptime.