

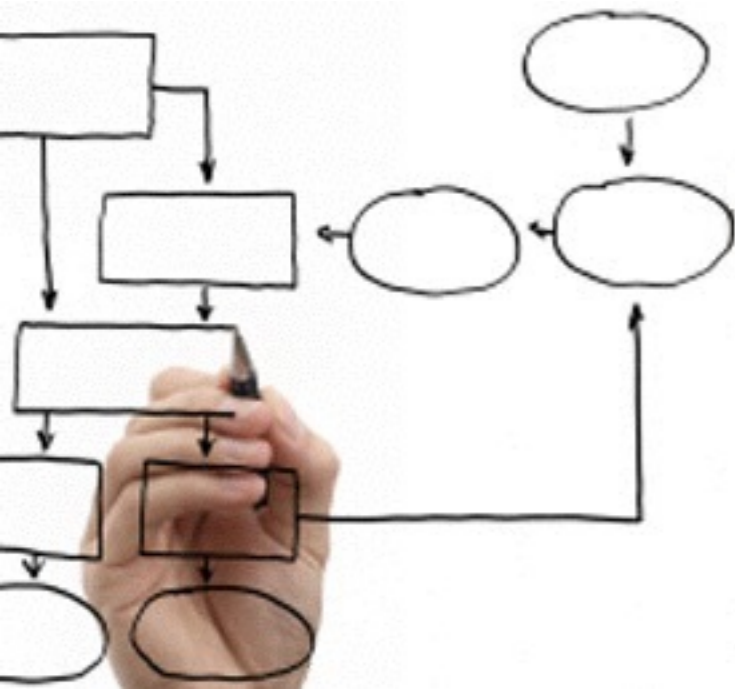
Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni

<http://www.di.unipi.it/~bruni>

07 - Introduction to nets



Object

Overview of the basic concepts of
Petri nets

Free Choice Nets (book, optional reading)

<https://www7.in.tum.de/~esparza/bookfc.html>

Why Petri nets

Business process analysis:

validation: testing correctness

verification: proving correctness

performance: planning and optimization

Use of Petri nets (or alike)

visual + formal

tool supported

Approaching Petri nets

Are you familiar with automata / transition systems?
They are fine for sequential protocols / systems
but do not capture concurrent behaviour directly

A Petri net is a mathematical model
of a parallel and concurrent system,

in the same way that a finite automaton is a
mathematical model of a sequential system

Approaching Petri nets

Petri net theory can be studied
at several level of details

We study some basics aspects, relevant to the
analysis of business processes

Petri nets have a faithful and convenient graphical
representation, that we introduce and motivate next

Finite automata examples

Applications

Finite automata are widely used, e.g., in
protocol analysis,
text parsing,
video game character behavior,
security analysis,
CPU control units,
natural language processing,
speech recognition,
mechanical devices
(like elevators, vending machines, traffic lights)

How to

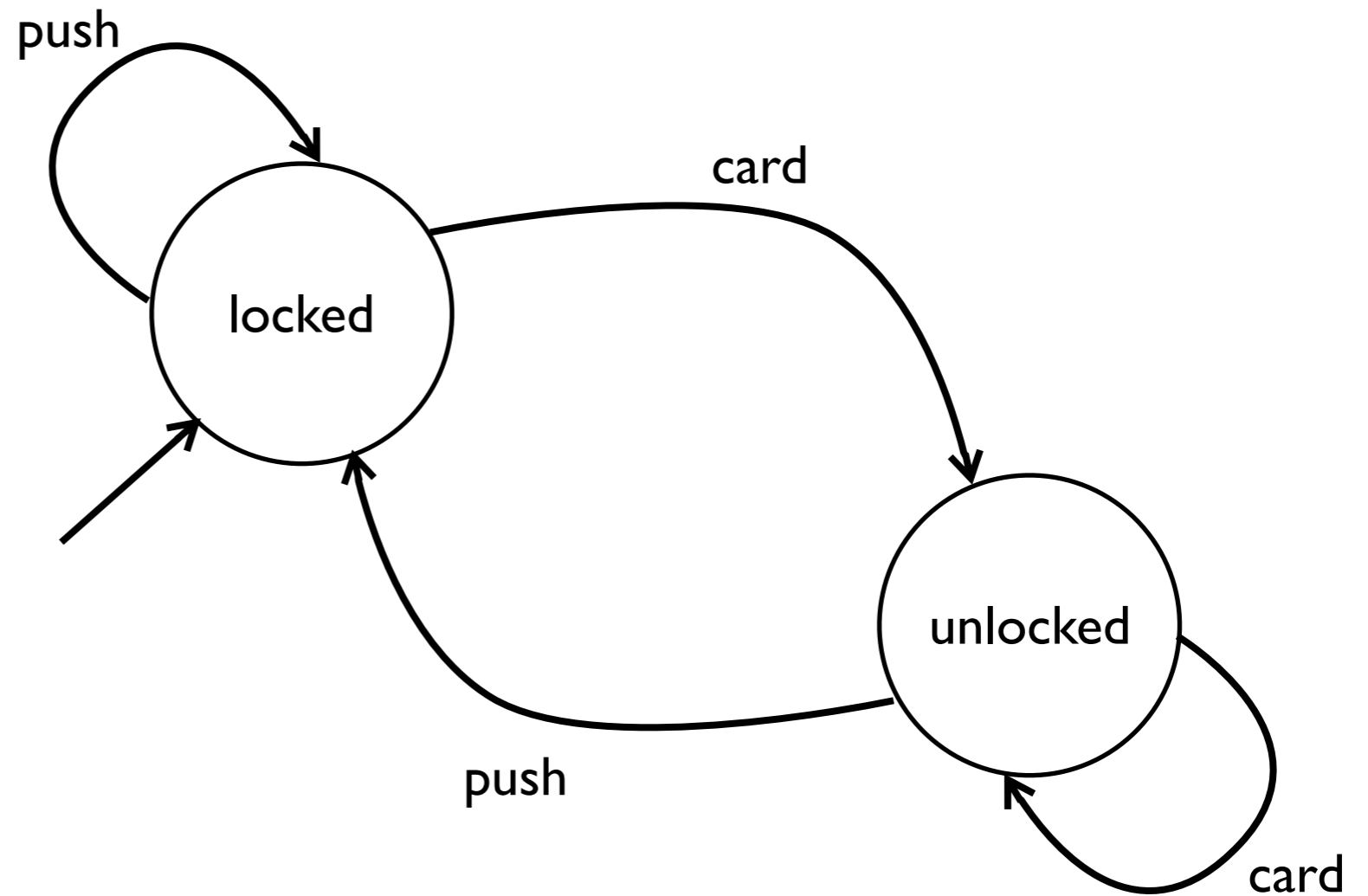
Identify the admissible states of the system
Optional: Mark some states as error states

Add transitions
to move from one state to another
(no transition to recover from error states)

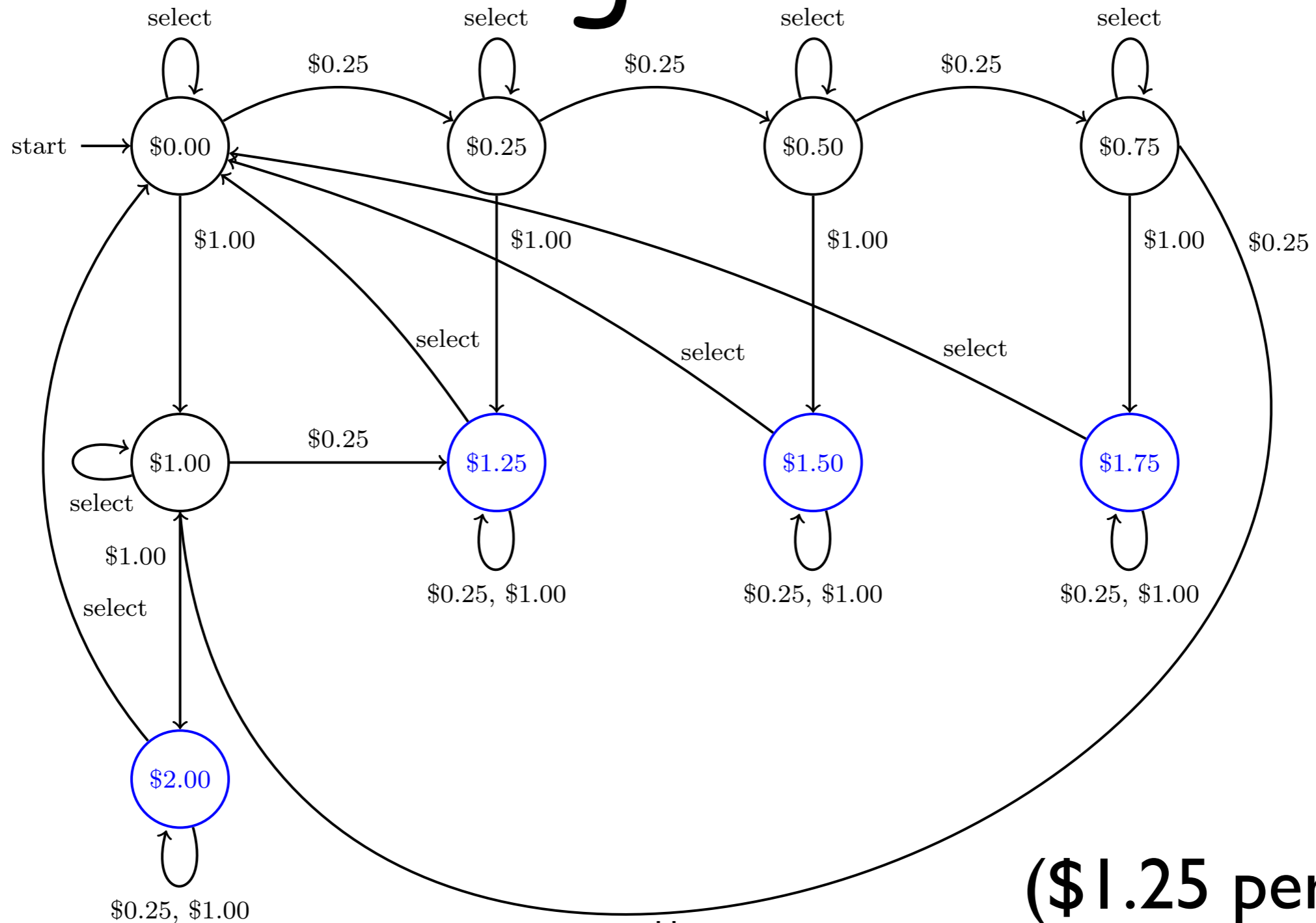
Set the starting state

Optional: Mark some states as final

Example: Turnstile



Example: Vending Machine



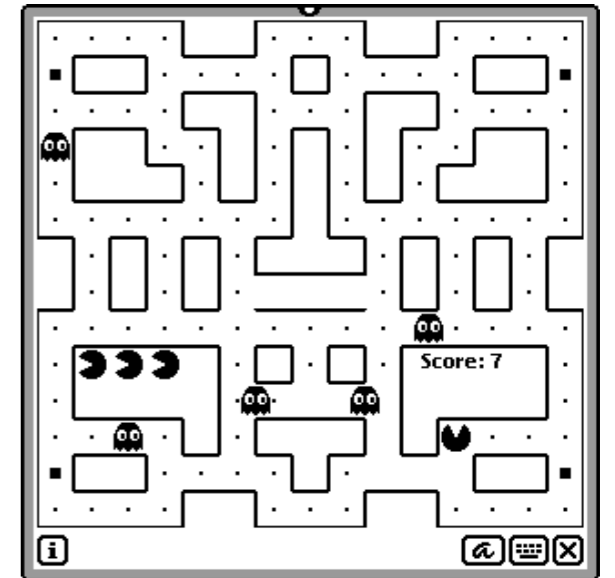
(\$1.25 per soda)

Computer controlled characters for games

States = characters behaviours

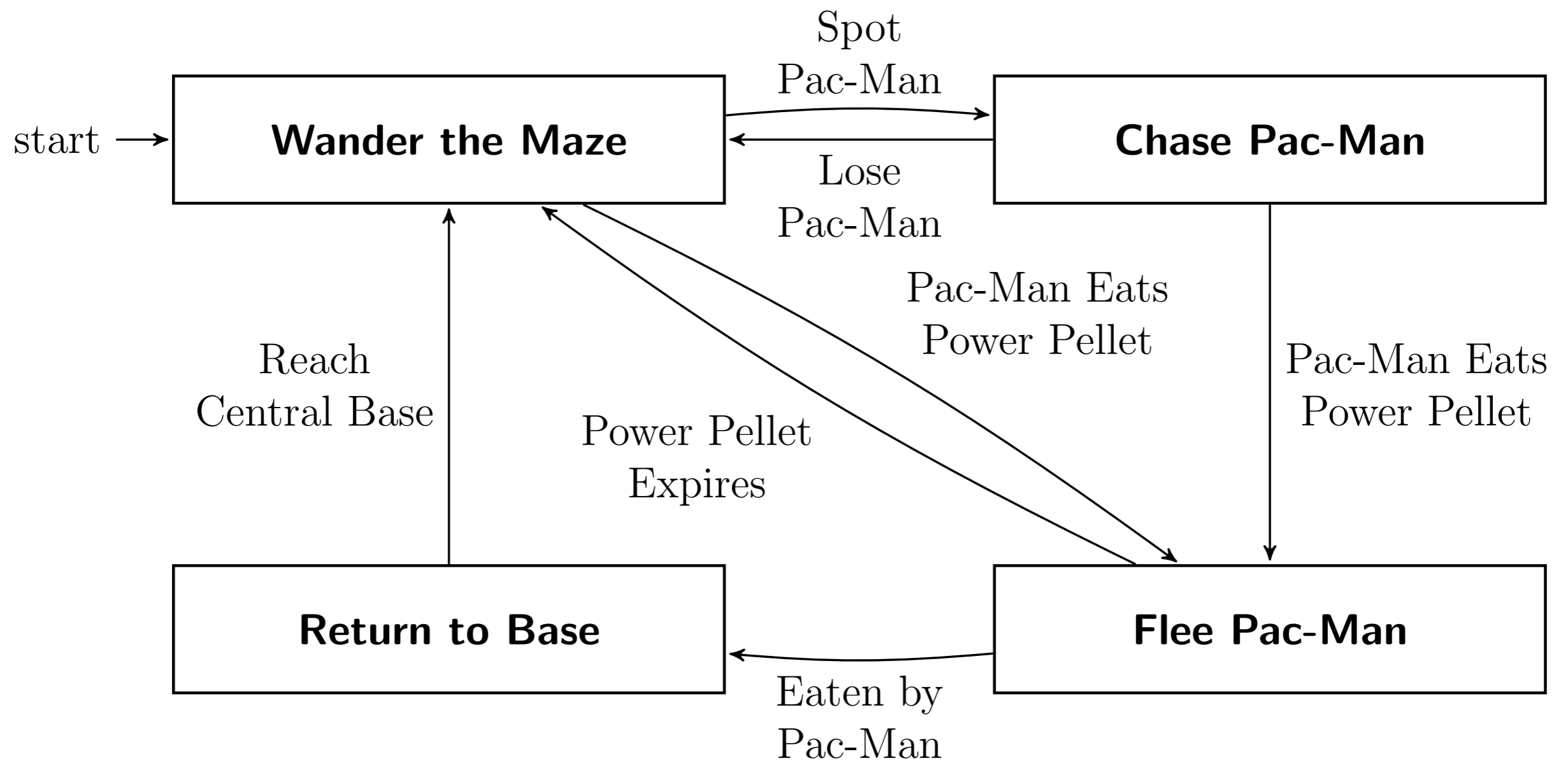
Transitions = labelled by events that cause a change
in behaviour

Example: Pac-man ghosts
pac-man navigates in a maze
wants to eat pills
is chased by ghosts



by eating power pills, pac-man can defeat ghosts

Example: Pac-Man Ghosts



Exercises

Without adding states, draw the automata for a SuperGhost that can't be eaten. It chases Pac-Man when the power pill is eaten, and returns to base if Pac-Man eats a piece of fruit.

Choose a favourite (video) game, and try drawing the state automata for one of the computer controlled characters in that game.

From automata to Petri nets

DFA

A **Deterministic Finite Automaton (DFA)** is a tuple $A = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states;
- Σ is a finite set of input symbols;
- $\delta : Q \times \Sigma \rightarrow Q$ is the transition function;
- $q_0 \in Q$ is the initial state (also called start state);
- $F \subseteq Q$ is the set of final states (also accepting states)

Notation A^*

Given a set A we denote by A^*

the set of finite sequences of elements in A , i.e.:

$$A^* = \{ a_1 \cdots a_n \mid n \geq 0 \wedge a_1, \dots, a_n \in A \}$$

We denote the empty sequence by $\epsilon \in A^*$

For example:

$$A = \{ a, b \} \quad A^* = \{ \epsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots \}$$

Extended transit. func. (destination function)

Given $A = (Q, \Sigma, \delta, q_0, F)$, we define $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ by induction:

base case: For any $q \in Q$ we let

$$\hat{\delta}(q, \epsilon) = q$$

inductive case: For any $q \in Q, a \in \Sigma, w \in \Sigma^*$ we let

$$\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$$

String processing

Given $A = (Q, \Sigma, \delta, q_0, F)$ and $w \in \Sigma^*$ we say that A **accept** w iff

$$\hat{\delta}(q_0, w) \in F$$

The **language** of $A = (Q, \Sigma, \delta, q_0, F)$ is

$$L(A) = \{ w \mid \hat{\delta}(q_0, w) \in F \}$$

Transition diagram

We represent $A = (Q, \Sigma, \delta, q_0, F)$ as a graph s.t.

- Q is the set of nodes;
- $\{ q \xrightarrow{a} q' \mid q' = \delta(q, a) \}$ is the set of arcs.

Plus some graphical conventions:

- there is one special arrow $Start$ with $\xrightarrow{Start} q_0$
- nodes in F are marked by double circles;
- nodes in $Q \setminus F$ are marked by single circles.

String processing as paths

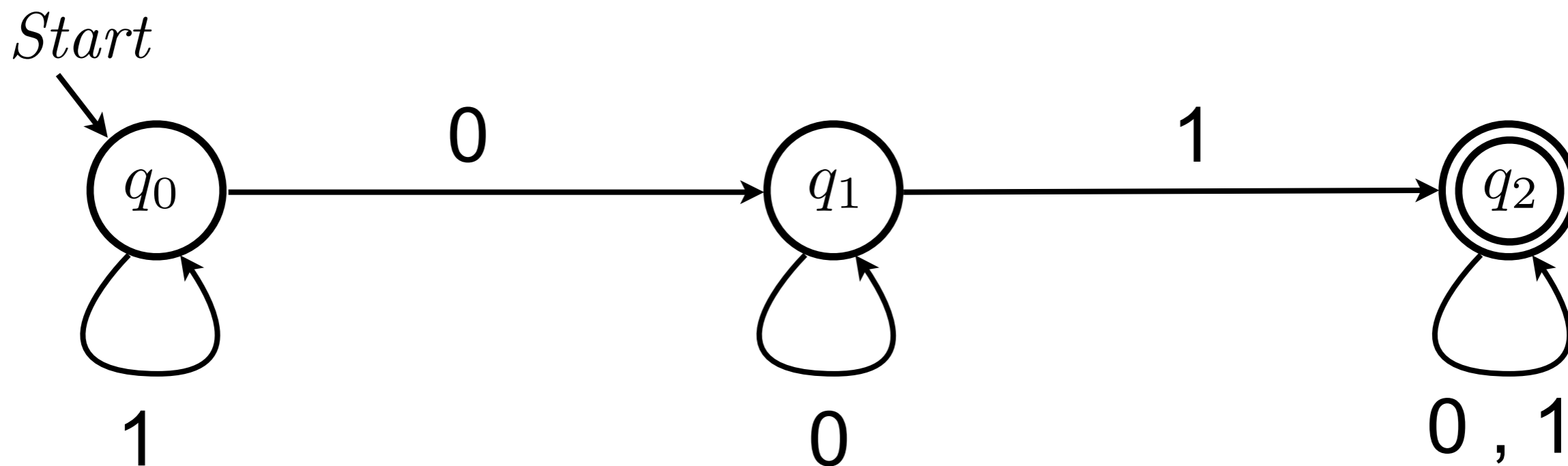
A DFA accepts a string w , if there is a path in its transition diagram such that:

it starts from the initial state

it ends in one final state

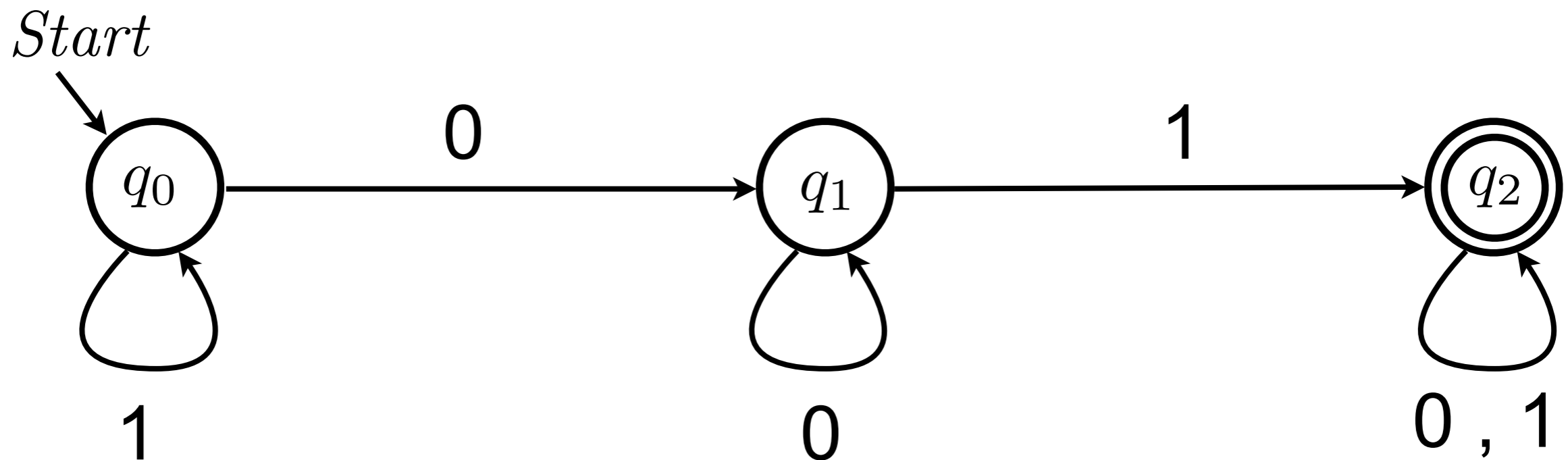
the sequence of labels in the path is exactly w

DFA: example



| | | | | | | | | | | | | |
|-------|---|-------|---|-------|---|-------|---|-------|---|-------|---|----------------|
| q_0 | 1 | q_0 | 1 | q_0 | 1 | q_0 | 0 | q_1 | 0 | q_1 | 0 | $q_1 \notin F$ |
| q_0 | 1 | q_0 | 0 | q_1 | 0 | q_1 | 1 | q_2 | 1 | q_2 | 0 | $q_2 \in F$ |

DFA: question time



Does it accept 100 ?

Does it accept 011 ?

Does it accept 1010010 ?

What is $L(A)$?

Transition table

Conventional tabular representation

its rows are in correspondence with states

its columns are in correspondence with input symbols

its entries are the states reached after the transition

Plus some decoration

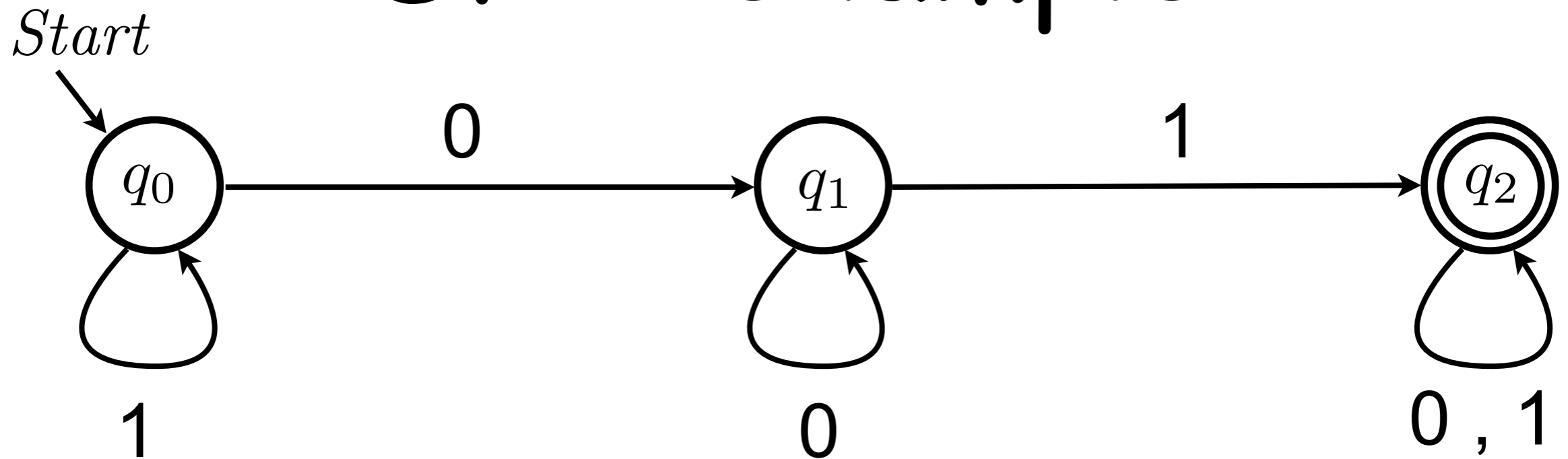
start state decorated with an arrow

all final states decorated with *

Transition table

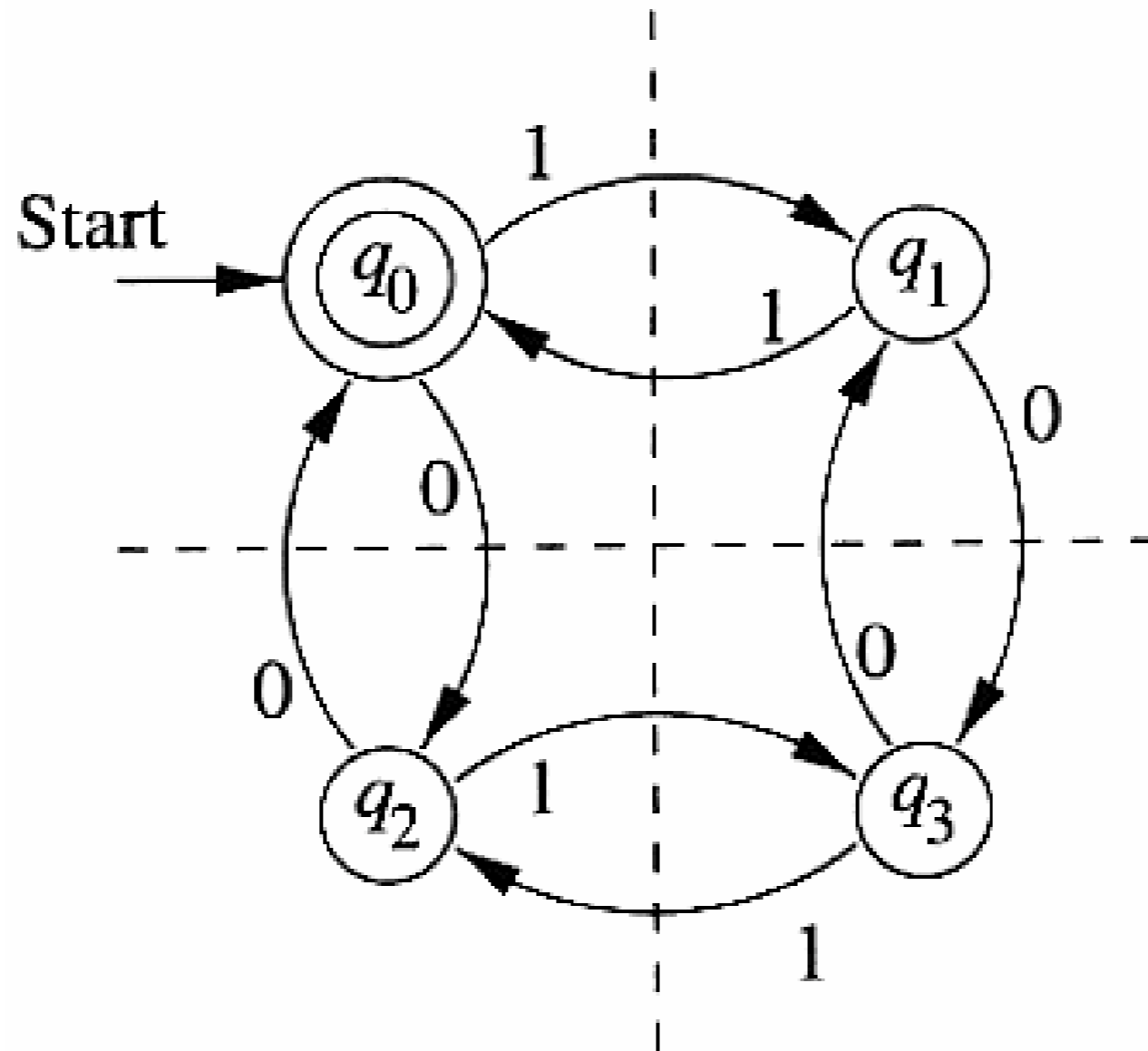
| | | | | a | | | |
|---|--|--|--|----------------|--|--|--|
| → | | | | | | | |
| | | | | | | | |
| q | | | | $\delta(q, a)$ | | | |
| * | | | | | | | |
| * | | | | | | | |
| | | | | | | | |

DFA: example



| | 0 | 1 |
|-------------------|---|---|
| $\rightarrow q_0$ | | |
| q_1 | | |
| $\star q_2$ | | |

DFA: exercise



Does it accept 100 ?
Write its transition table.

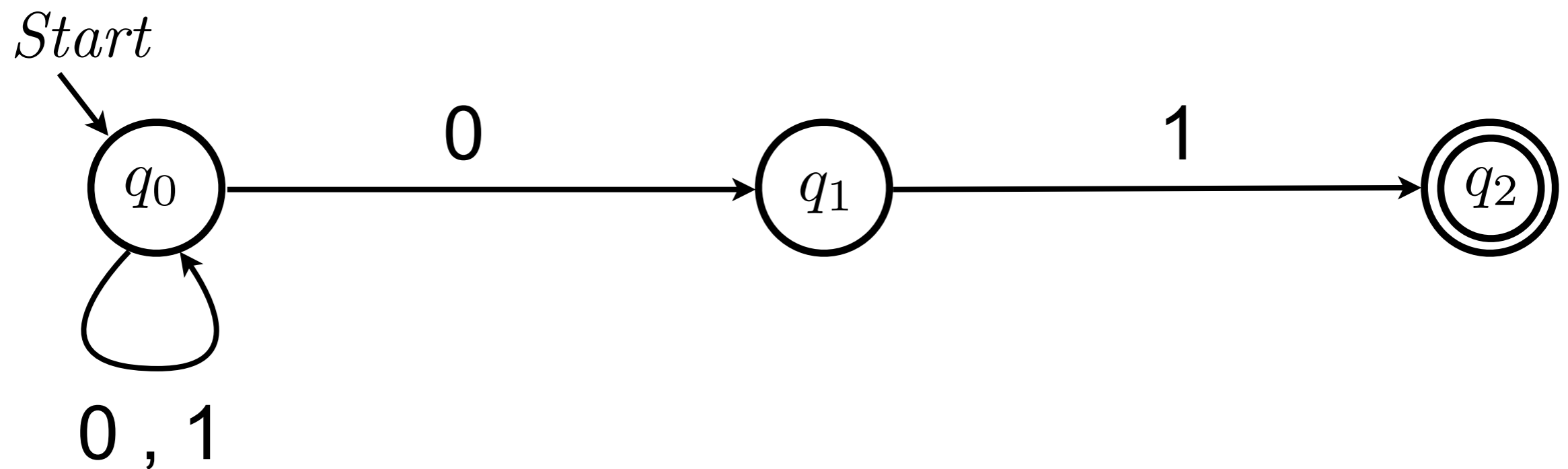
Does it accept 1010 ?
What is $L(A)$?

NFA

A **Non-deterministic Finite Automaton (NFA)** is a tuple $A = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states;
- Σ is a finite set of input symbols;
- $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$ is the transition function;
 set of sets over Q
- $q_0 \in Q$ is the initial state (also called start state);
- $F \subseteq Q$ is the set of final states (also accepting states)

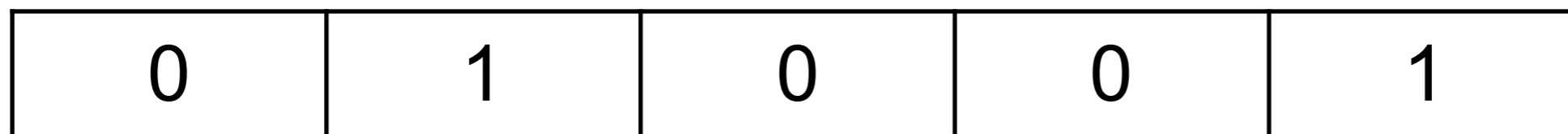
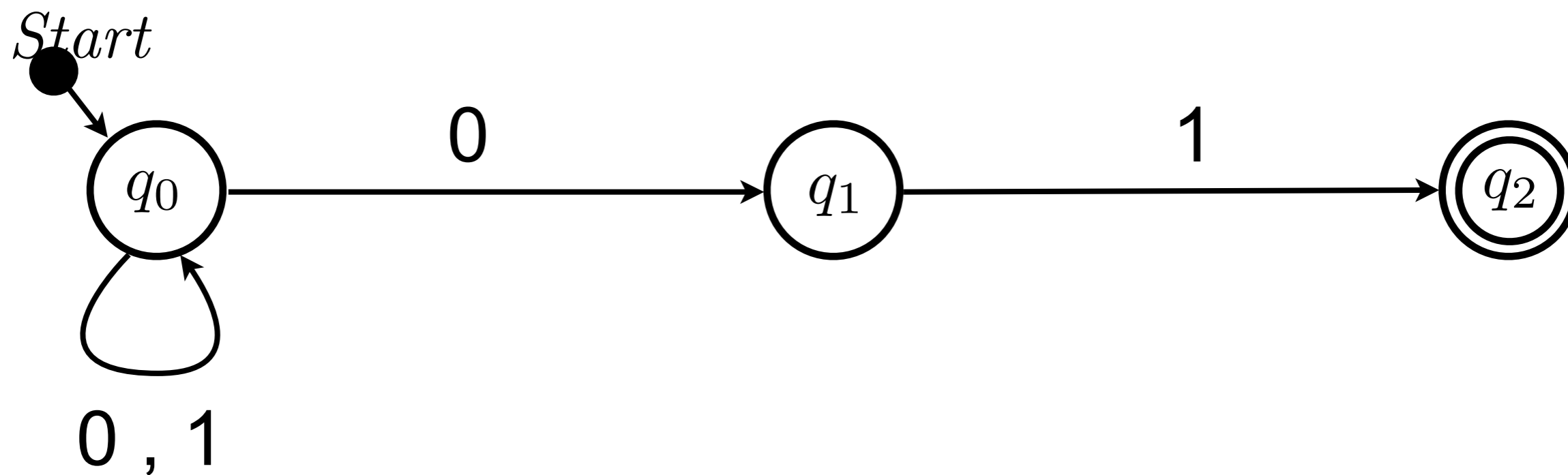
NFA: example



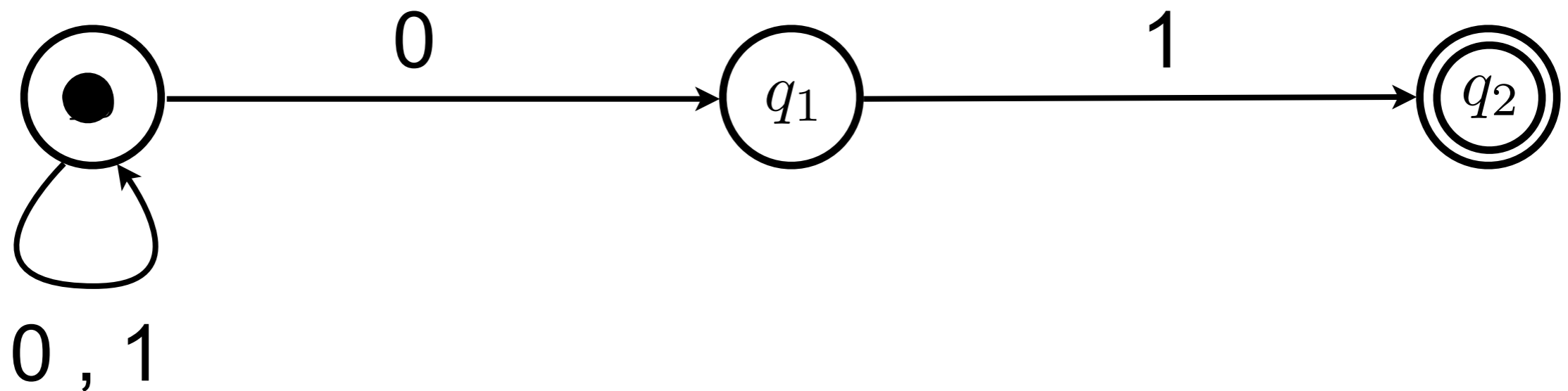
Can you explain why it is not a DFA?

Reshaping

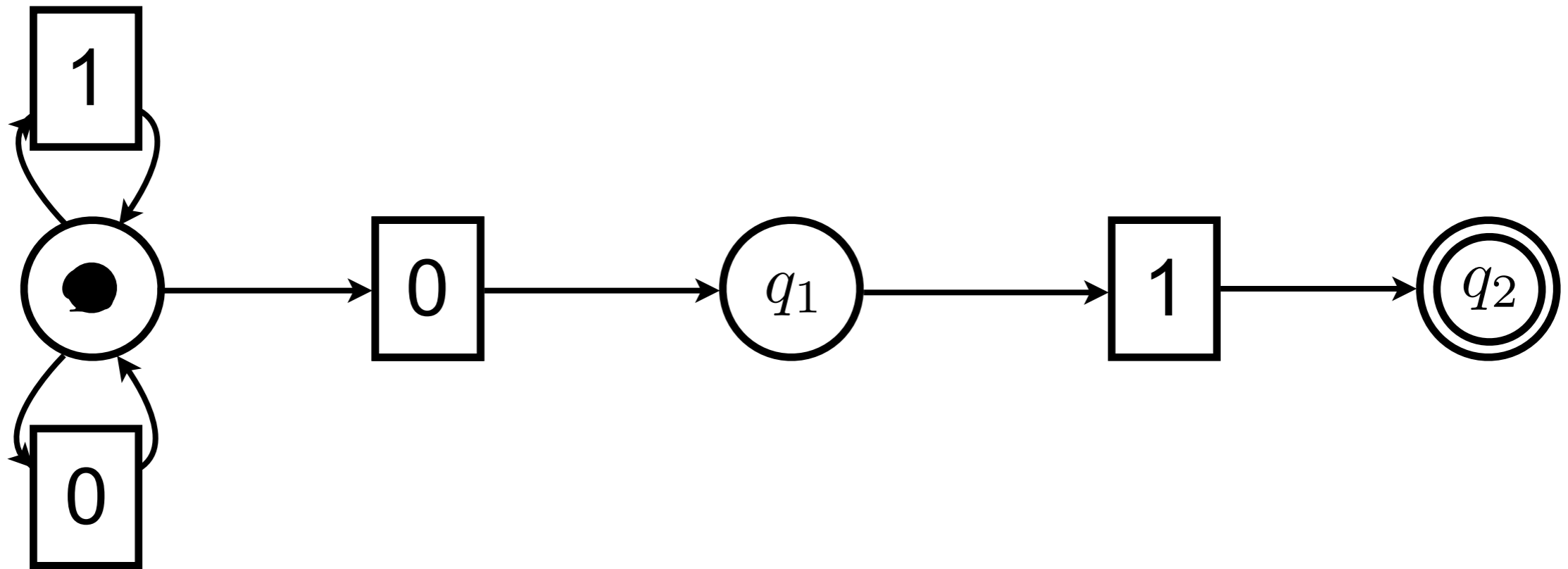
Step 1: get a token



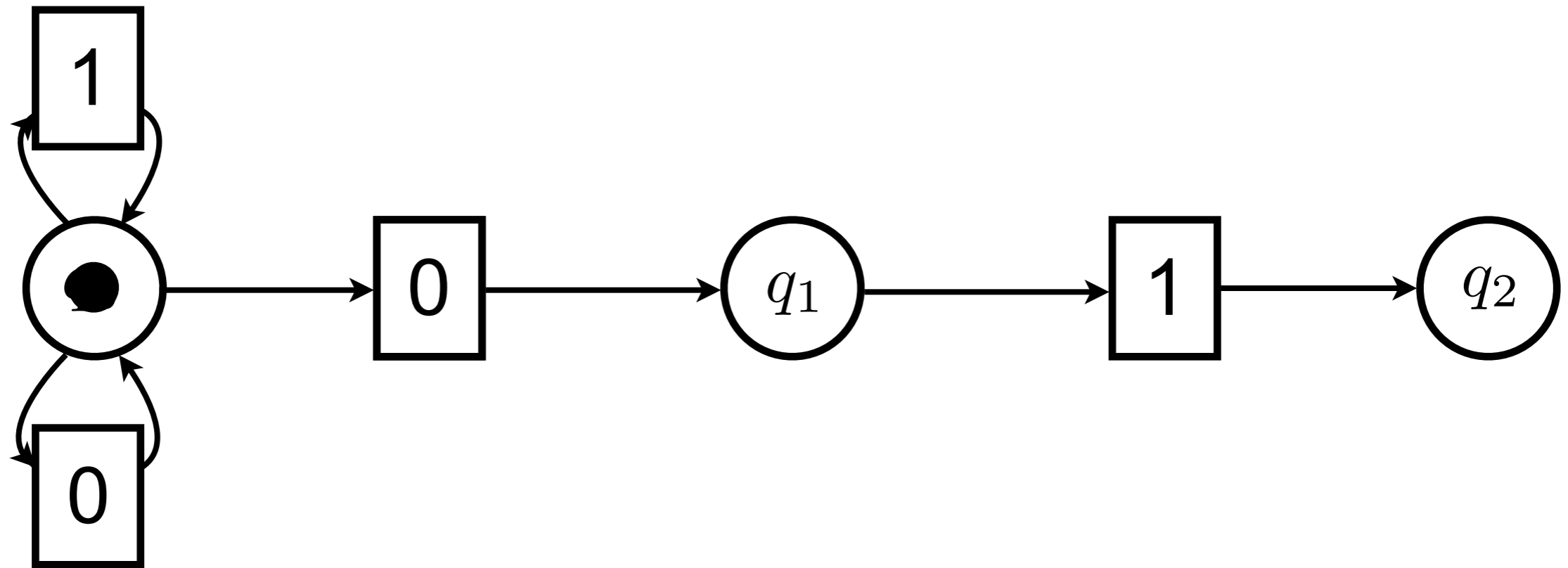
Step 2: forget initial state decoration



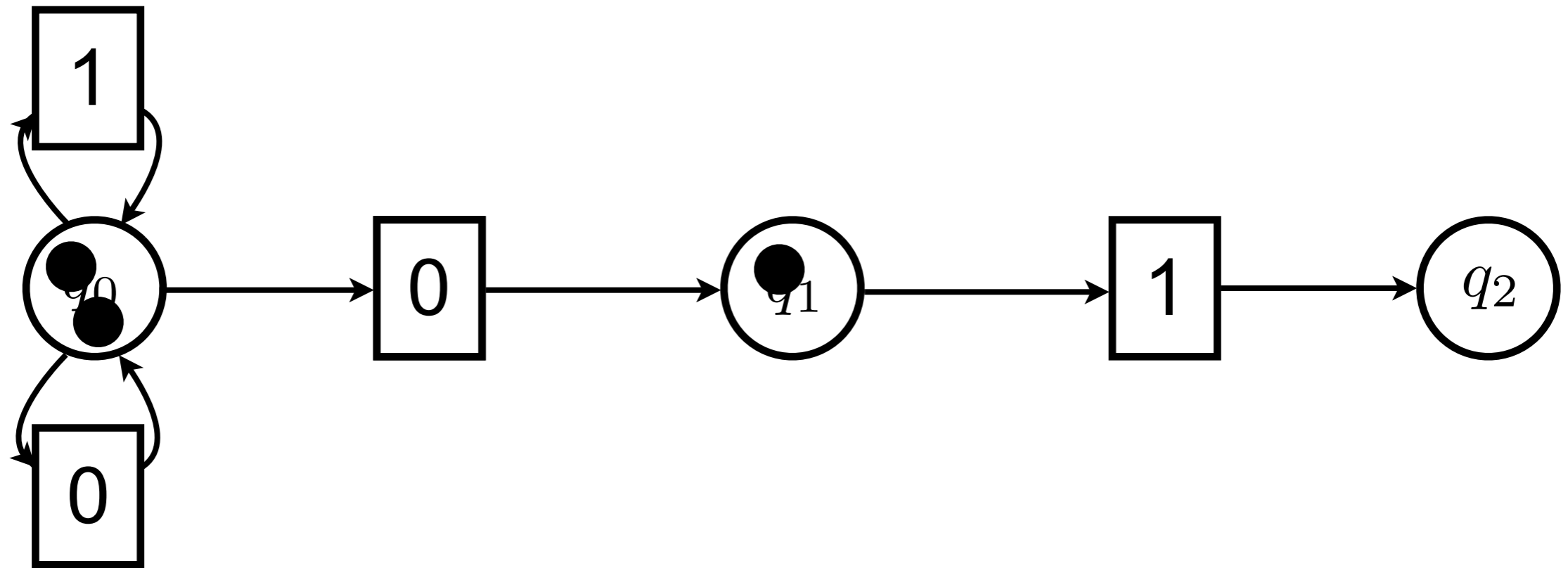
Step 3: transitions as boxes



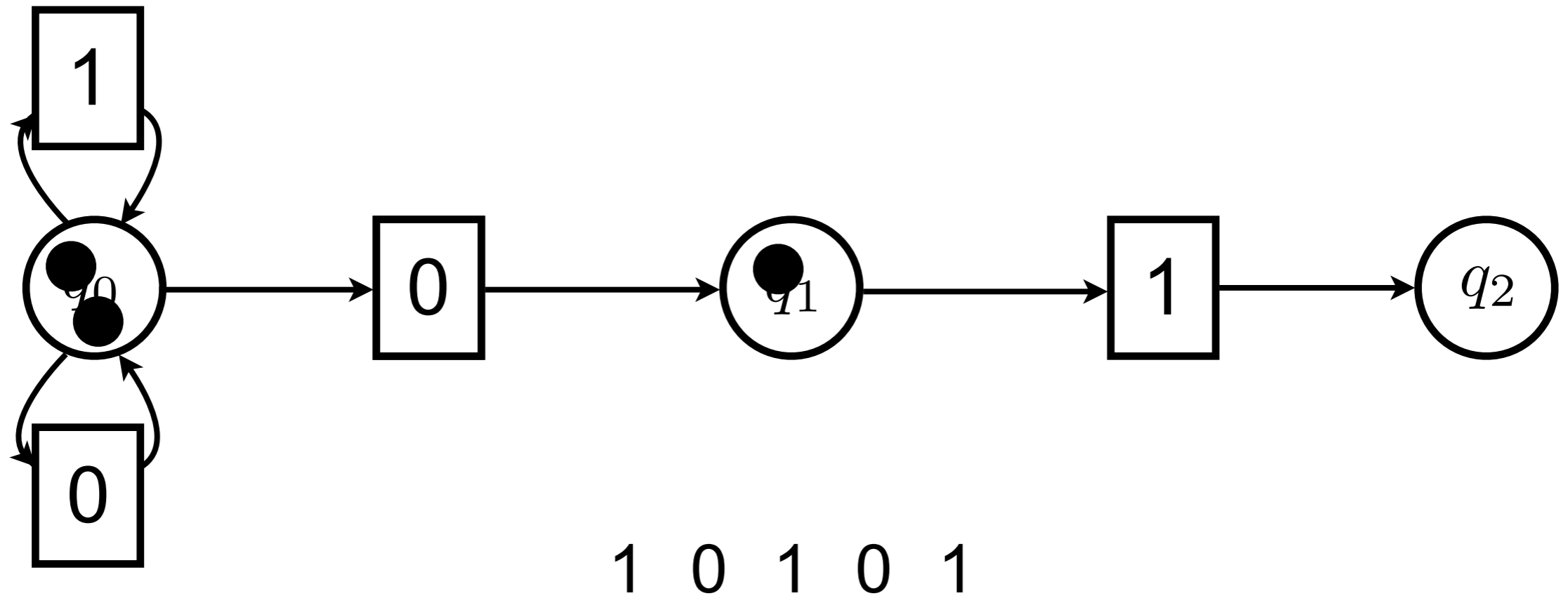
Step 4: forget final states



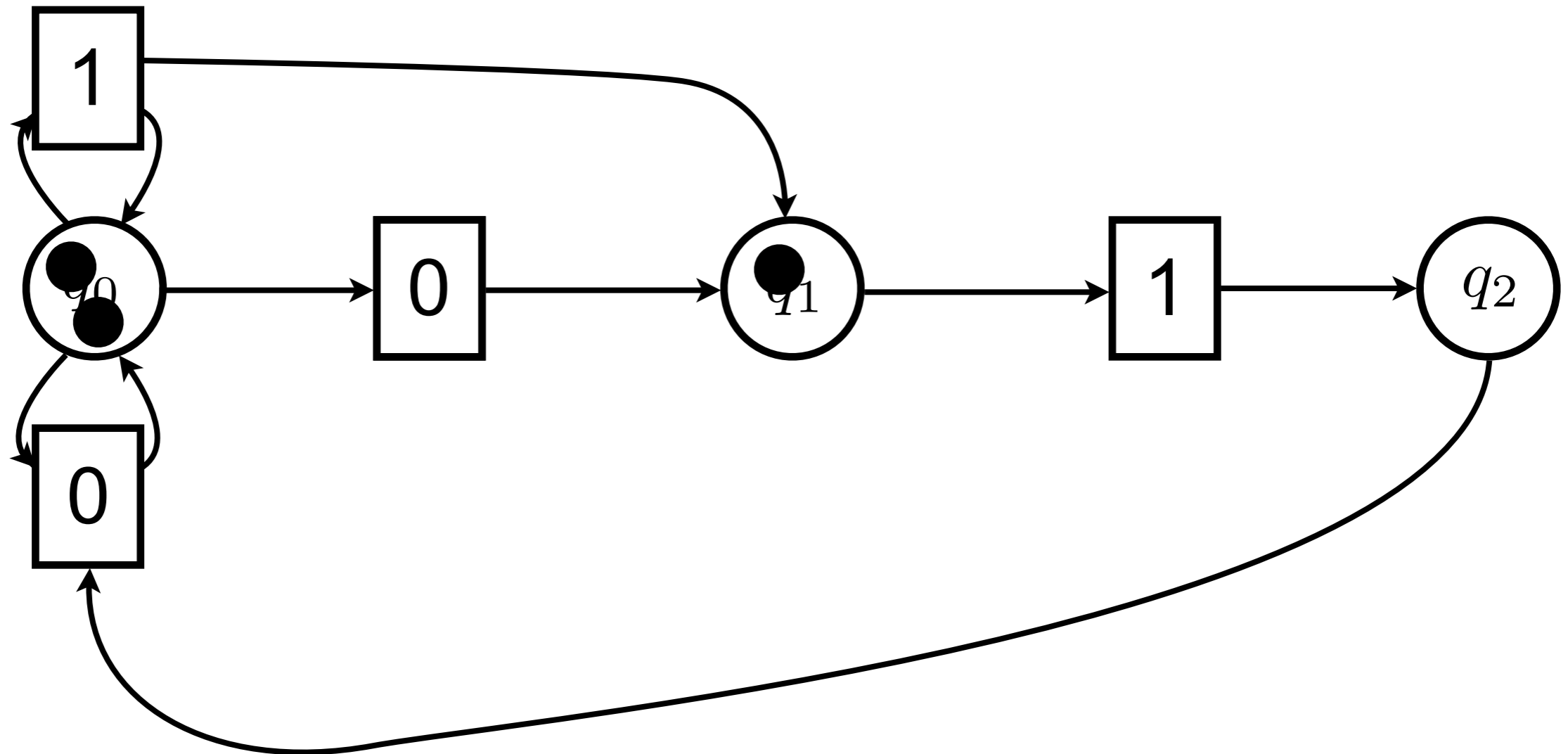
Step 5: allow for more tokens



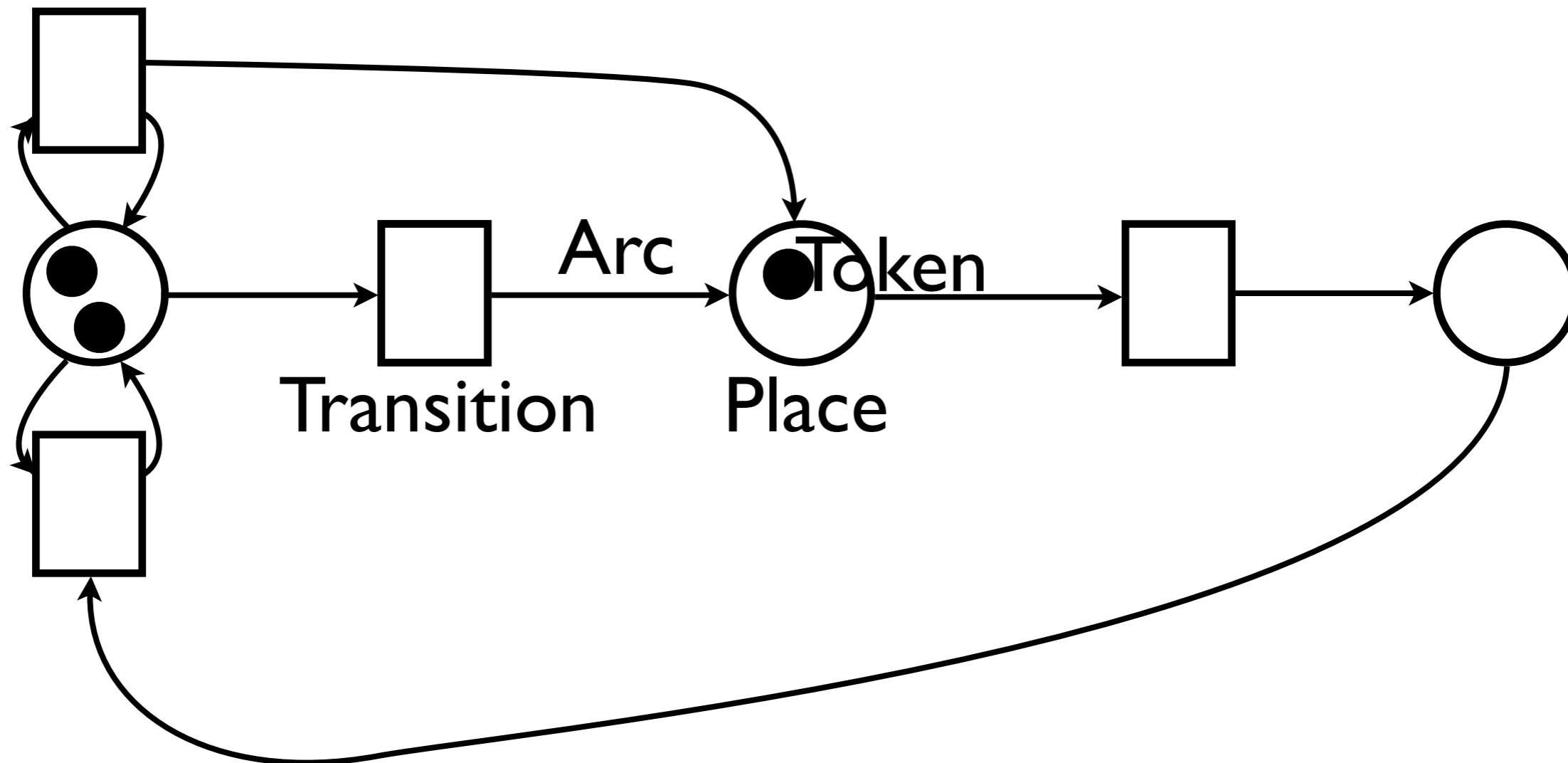
Example: token game



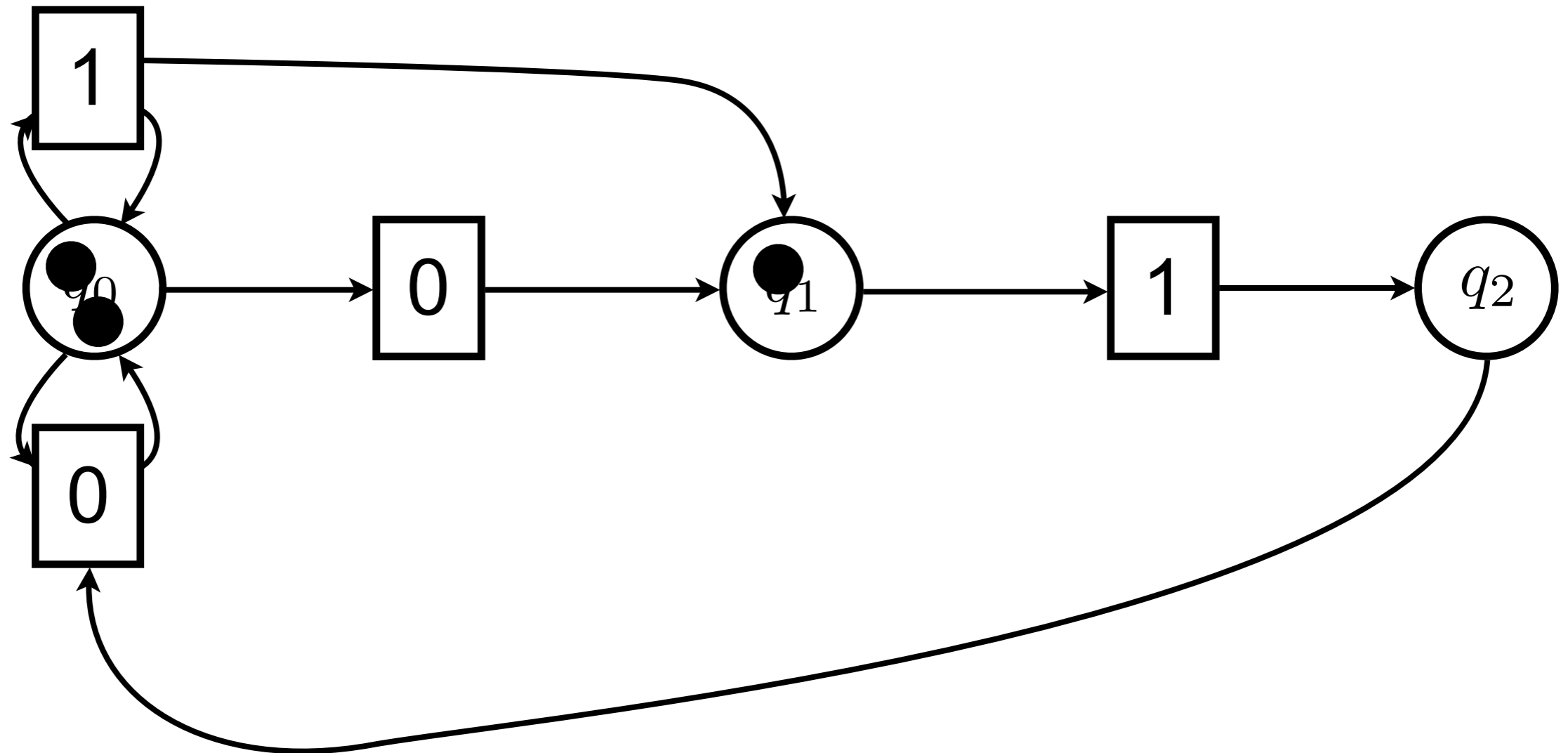
Step 6: allow for more arcs



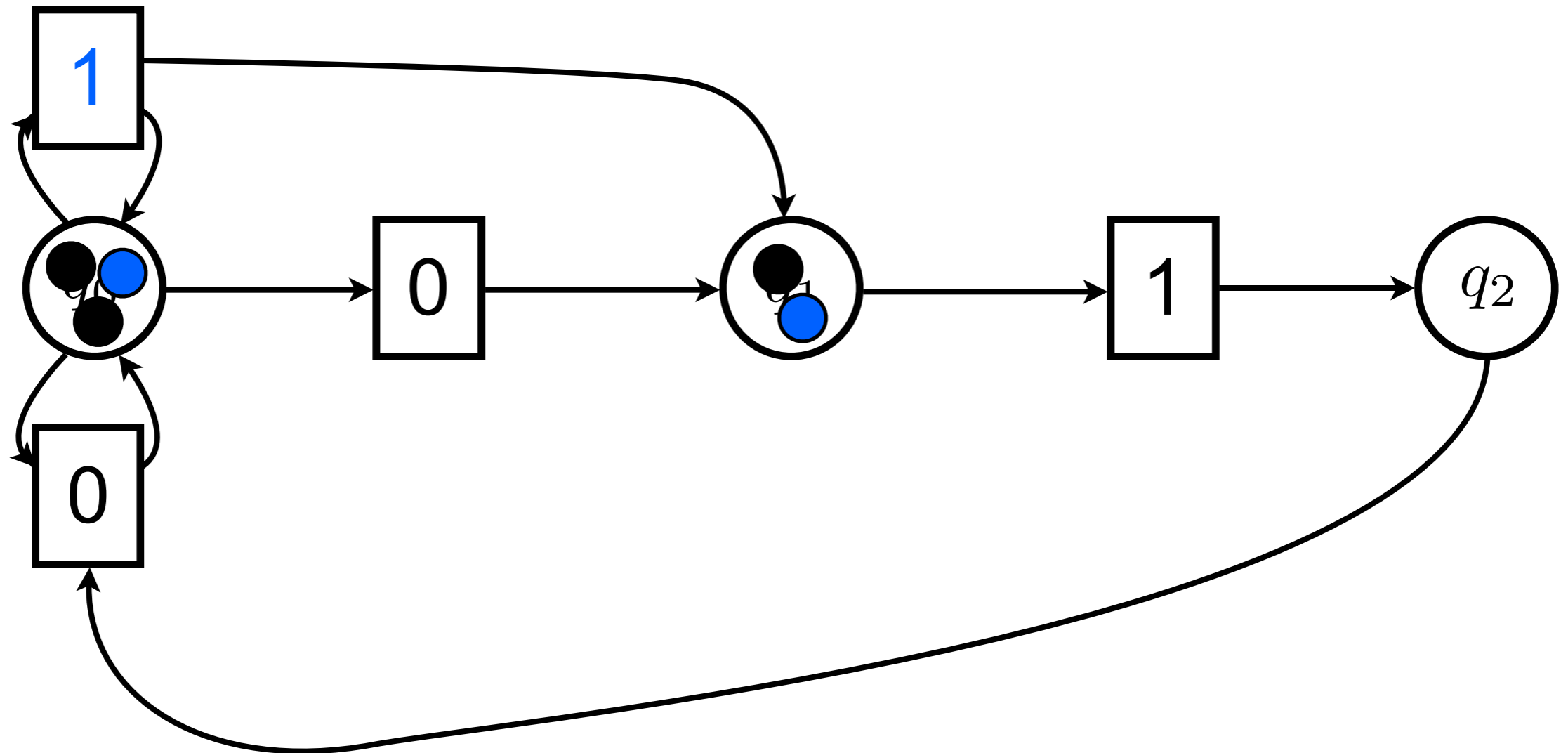
Terminology



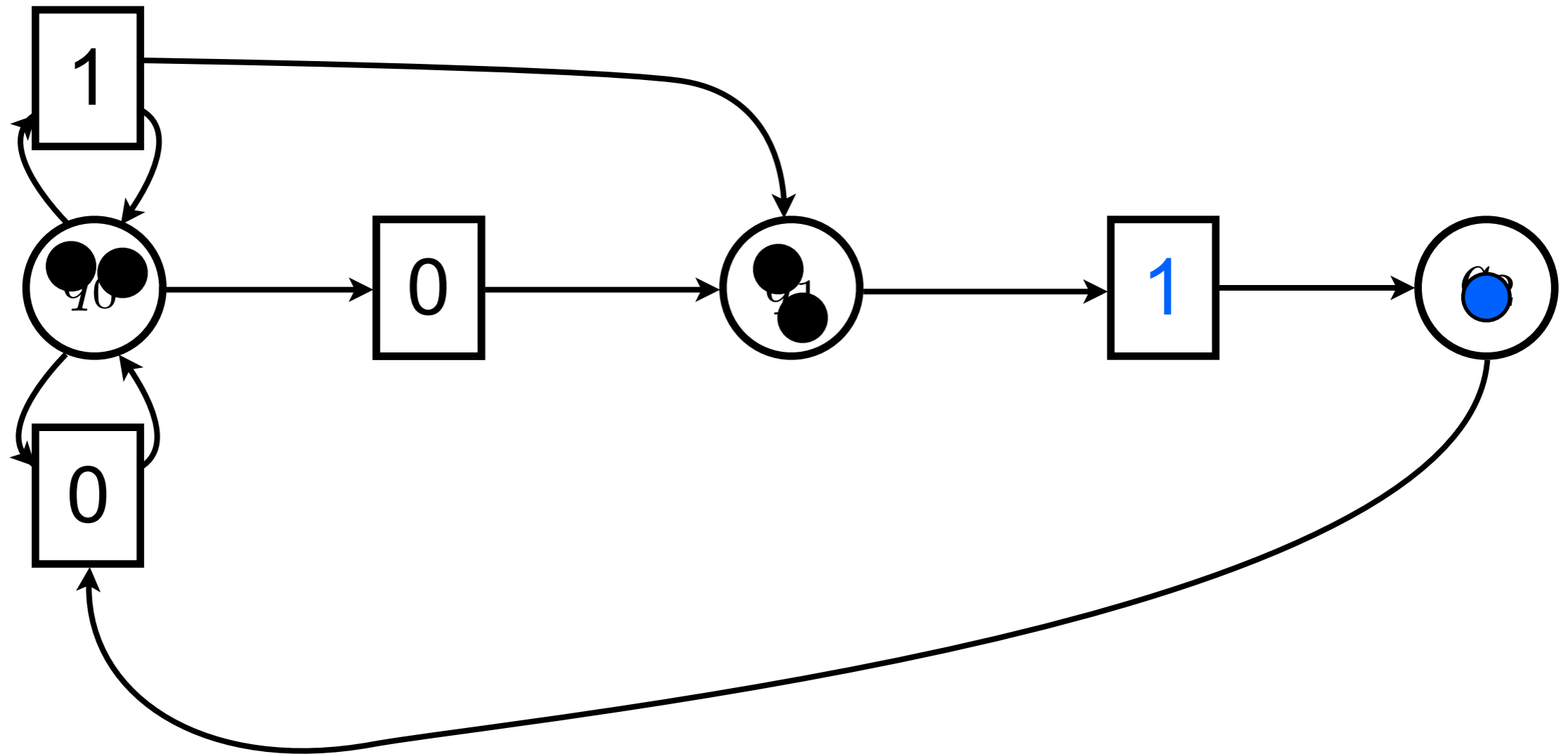
Example: token game



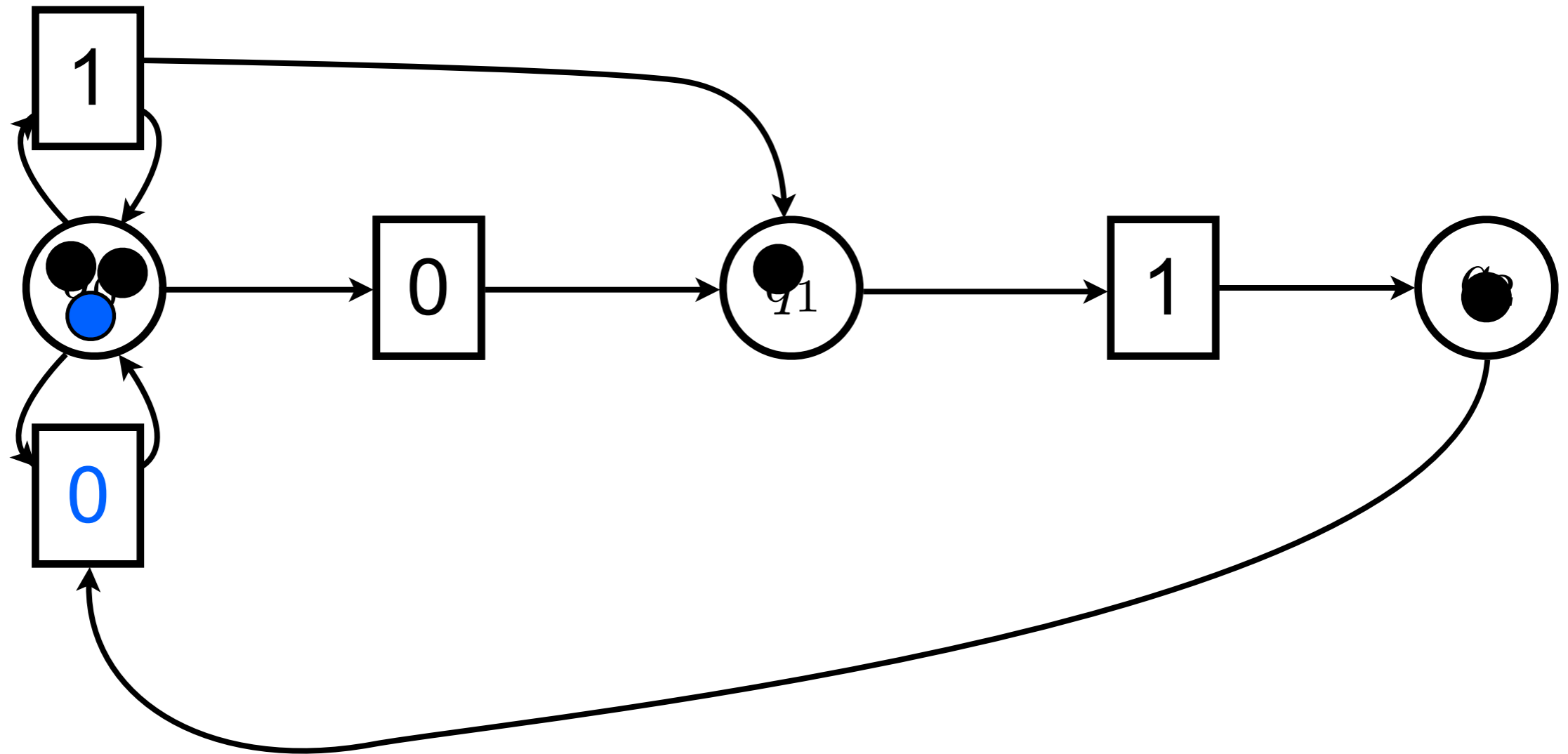
Example: token game



Example: token game



Example: token game



Some hints

Nets are **bipartite graphs**:
arcs never connect two places
arcs never connect two transitions

Static structure for dynamic systems:
places, transitions, arcs do not change
tokens move around places

Places are passive components
Transitions are active components:
tokens do not flow!
(they are removed or freshly created)

Petri nets: basic definition



Carl Adam Petri

July 12, 1926 - July 2, 2010

http://www.informatik.uni-hamburg.de/TGI/mitarbeiter/profs/petri_eng.html

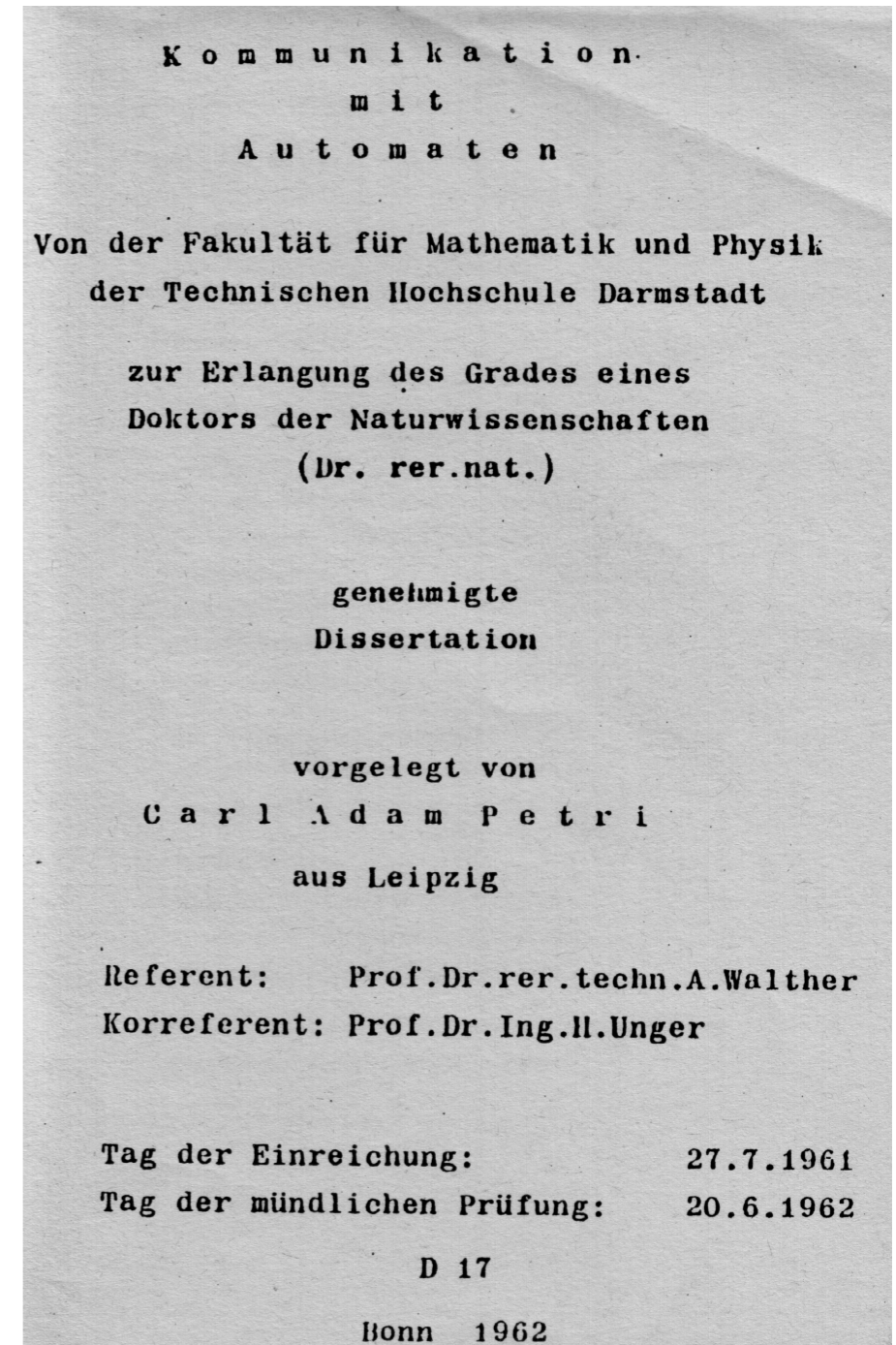
Introduced in 1962 (Petri's PhD thesis)

60's and 70's main focus on theory

80's focus on tools and applications

Now applied in several fields

Success due to simple and clean
graphical and conceptual
representation



Petri nets for us

Formal and abstract business process specification

Formal: the semantics of process instances becomes well defined and not ambiguous

Abstract: execution environment is disregarded

(Remind about separation of concerns)

Places

A place can stand for
a state
a medium
a buffer
a condition
a repository of resources
a type

...

Tokens

A token can stand for
a physical object
a piece of data
a resource
an activation mark
a message
a document
a case

...

Transitions

A transition can stand for
an event
an operation
a transformation
a transportation
a task
an activity

...

Notation: from sets...

Let S be a set.

Let $\wp(S)$ denote the set of sets over S .

Elements $A \in \wp(S)$ (i.e., $A \subseteq S$)
are in bijective correspondence with
functions $f : S \rightarrow \{0, 1\}$

$$x \in A \text{ iff } f_A(x) = 1$$

Notation: ... to multisets

Let $\mu(S)$ (or S^\oplus) denote the set of multisets over S .

Elements $B \in \mu(S)$ are in bijective correspondence with functions $M : S \rightarrow \mathbb{N}$

$M_B(x)$ is the number of instances of x in B

$x \in B$ iff $M_B(x) > 0$

Notation: multisets

Empty multiset:

\emptyset is such that $\emptyset(x) = 0$ for all $x \in S$

Multiset containment:

we write $M \subseteq M'$ if $M(x) \leq M'(x)$ for all $x \in S$

Multiset strict containment:

we write $M \subset M'$ if $M \subseteq M'$ and $M \neq M'$

Multiset union:

$M + M'$ is the multiset s.t. $(M + M')(x) = M(x) + M'(x)$ for all $x \in S$

Multiset difference (defined only if $M \supseteq M'$):

$M - M'$ is the multiset s.t. $(M - M')(x) = M(x) - M'(x)$ for all $x \in S$

Notation: multisets

Multiset $M = \{ k_1x_1, k_2x_2, \dots, k_nx_n \}$ as formal sum:

$$k_1x_1 + k_2x_2 + \dots + k_nx_n$$

$$\sum_{i=1}^n k_i x_i$$

Marking

A **marking** $M : P \rightarrow \mathbb{N}$ denotes the number of tokens in each place

The marking of a Petri net represents its state

$M(a) = 0$ denotes the absence of tokens in place a

Question time

$$3a + 2b \stackrel{?}{\subseteq} 2a + 3b + c$$

$$3a + 2b \stackrel{?}{\supseteq} 2a + 3b + c$$

$$a + 2b \stackrel{?}{\subset} 2a + 3b$$

$$(a + 2b) + (2a + c) = ?$$

$$(2a + 3b) - (2a + b) = ?$$

$$(2a + 2b) - (a + c) = ?$$

Petri nets

A **Petri net** is a tuple (P, T, F, M_0) where

- P is a finite set of **places**;
- T is a finite set of **transitions**;
- $F \subseteq (P \times T) \cup (T \times P)$ is a **flow relation**;
- $M_0 : P \rightarrow \mathbb{N}$ is the **initial marking**.
(i.e. $M_0 \in \mu(P)$)

Pre-set and post-set

A place p is an input place for transition t iff

$$(p, t) \in F$$

We let $\bullet t$ denote the set of input places of t .

(pre-set of t)

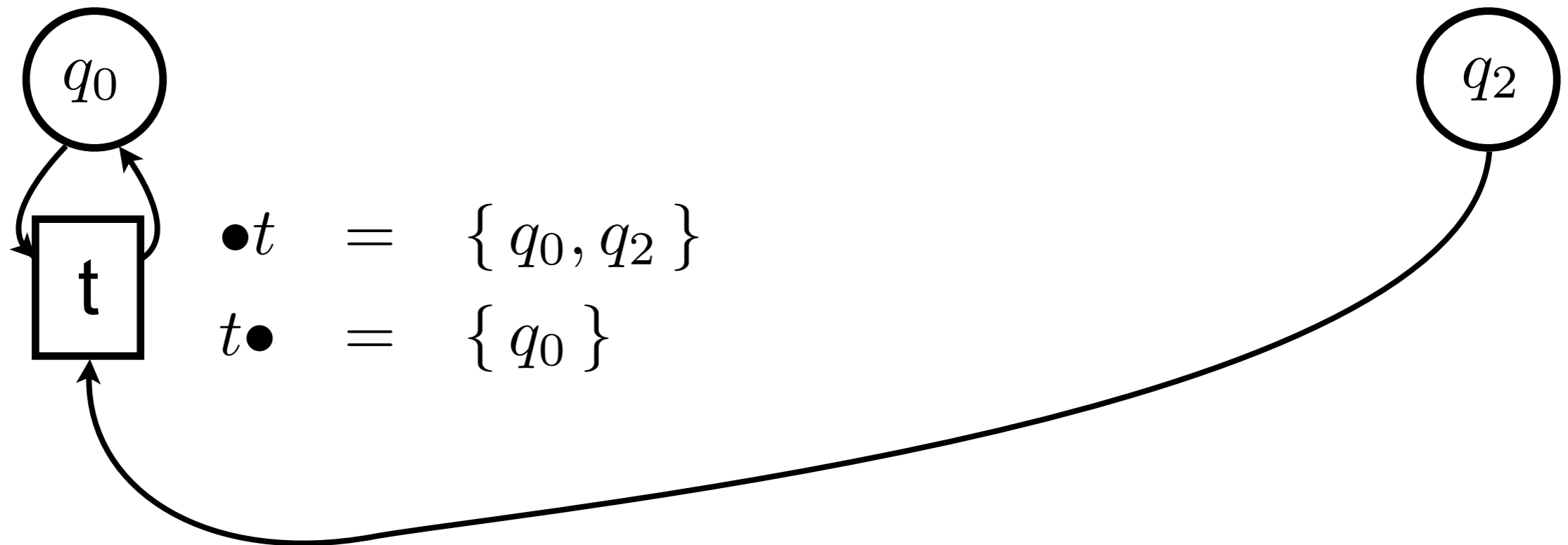
A place p is an output place for transition t iff

$$(t, p) \in F$$

We let $t\bullet$ denote the set of output places of t .

(post-set of t)

Example: pre and post



Pre-set and post-set

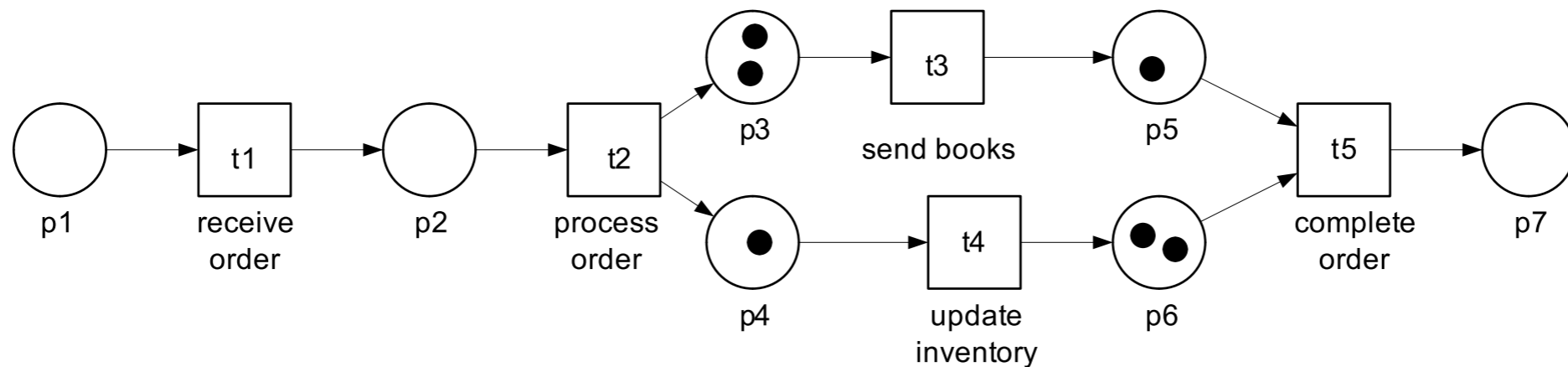
Analogously, we let

- p denote the set of transitions that share p as output place
- p • denote the set of transitions that share p as input place

Formally:

$$\begin{aligned} \bullet x &= \{ y \mid (y, x) \in F \} \\ x \bullet &= \{ y \mid (x, y) \in F \} \end{aligned}$$

Exercises



M. Weske: Business Process Management, © Springer-Verlag Berlin Heidelberg 2007

$$P = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7\}$$

$$T = \{t_1, t_2, t_3, t_4, t_5\}$$

$$F = \{(p_1, t_1), (t_1, p_2), \dots ?\}$$

$$M_0 = 2p_3 + \dots ?$$

$$\bullet t_1 = ?$$

$$t_1 \bullet = ?$$

$$\bullet t_2 = ?$$

$$t_2 \bullet = ?$$

$$\bullet t_3 = ?$$

$$t_3 \bullet = ?$$

$$\bullet t_4 = ?$$

$$t_4 \bullet = ?$$

$$\bullet t_5 = ?$$

$$t_5 \bullet = ?$$

$$\bullet p_1 = ?$$

$$p_1 \bullet = ?$$

$$\bullet p_2 = ?$$

$$p_2 \bullet = ?$$

$$\bullet p_3 = ?$$

$$p_3 \bullet = ?$$

$$\bullet p_4 = ?$$

$$p_4 \bullet = ?$$

$$\bullet p_5 = ?$$

$$p_5 \bullet = ?$$

$$\bullet p_6 = ?$$

$$p_6 \bullet = ?$$

$$\bullet p_7 = ?$$

$$p_7 \bullet = ?$$