Enabling Preserving Bisimulation Equivalence: a tool to prove liveness properties

Rob van Glabbeek

Data61, CSIRO, Sydney, Australia

University of New South Wales, Sydney, Australia

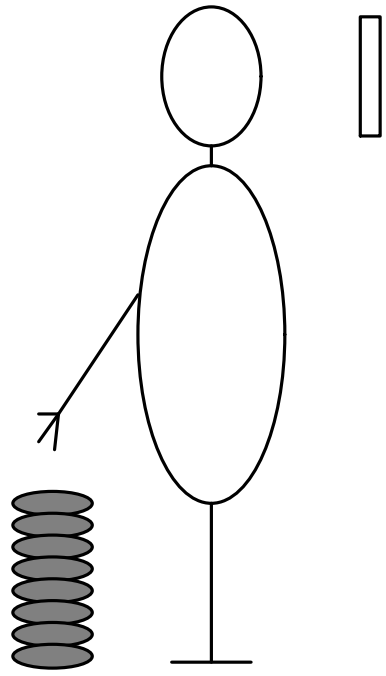
16 August 2021

Joint work with Weiyou Wang and Peter Höfner

Liveness properties – an example

↑

Something good will eventually happen.

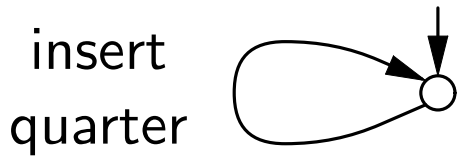


Task: insert an infinite pile of quarters in slot

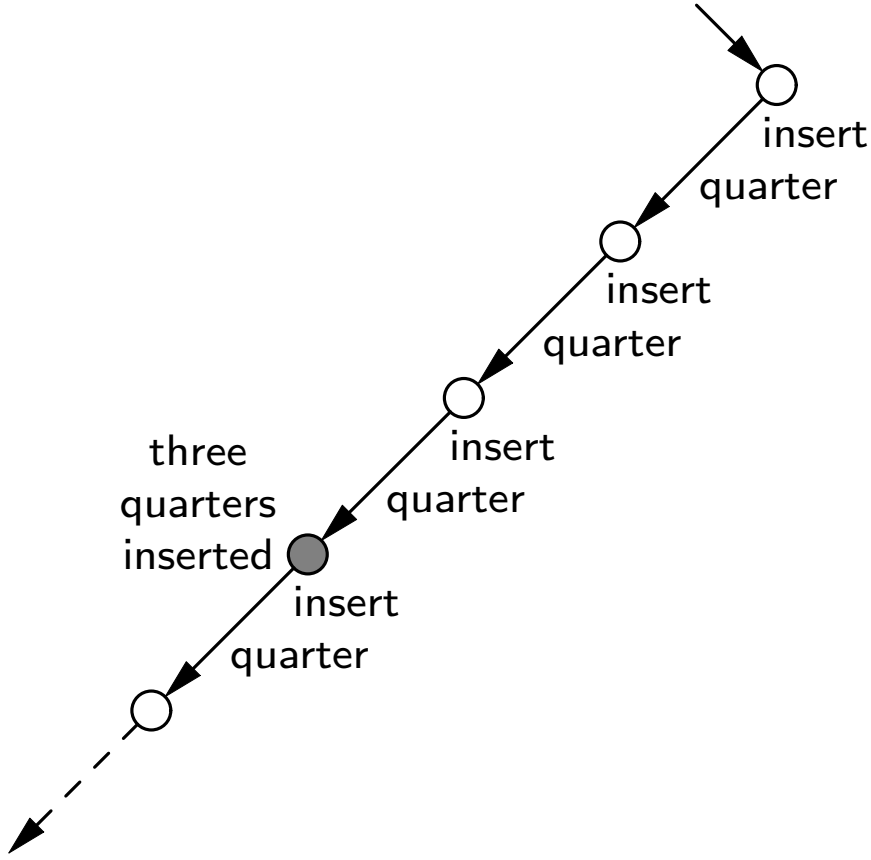
A liveness property: at least 3 quarters will be inserted.

Intuitively, this property holds, when assuming *progress*.

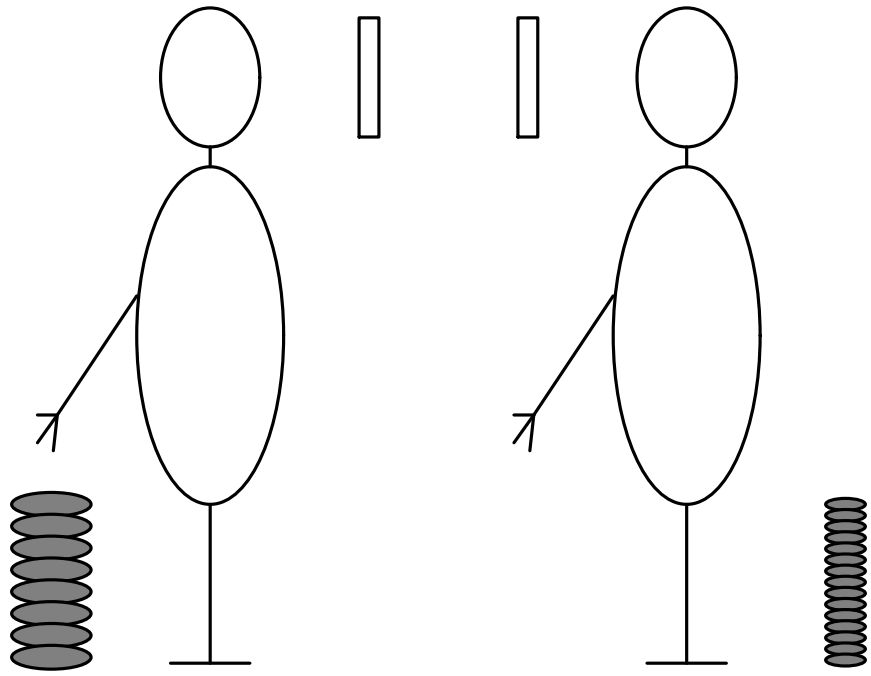
Transition system of example



Transition system with success state



Liveness properties – a more interesting example



Tasks: insert an infinite pile of quarters in left slot

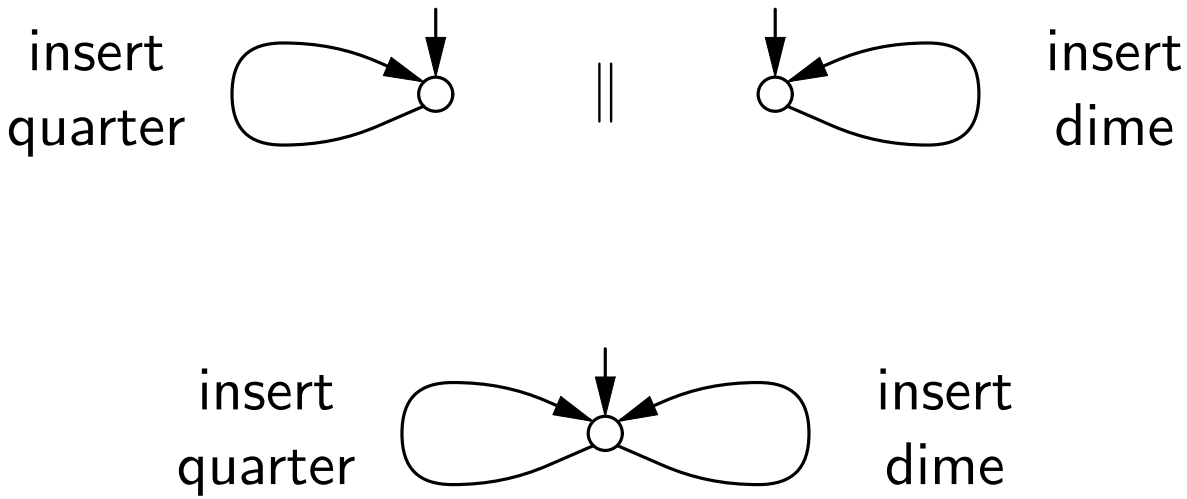
insert an infinite pile of dimes in right slot

A liveness property: at least 3 quarters will be inserted.

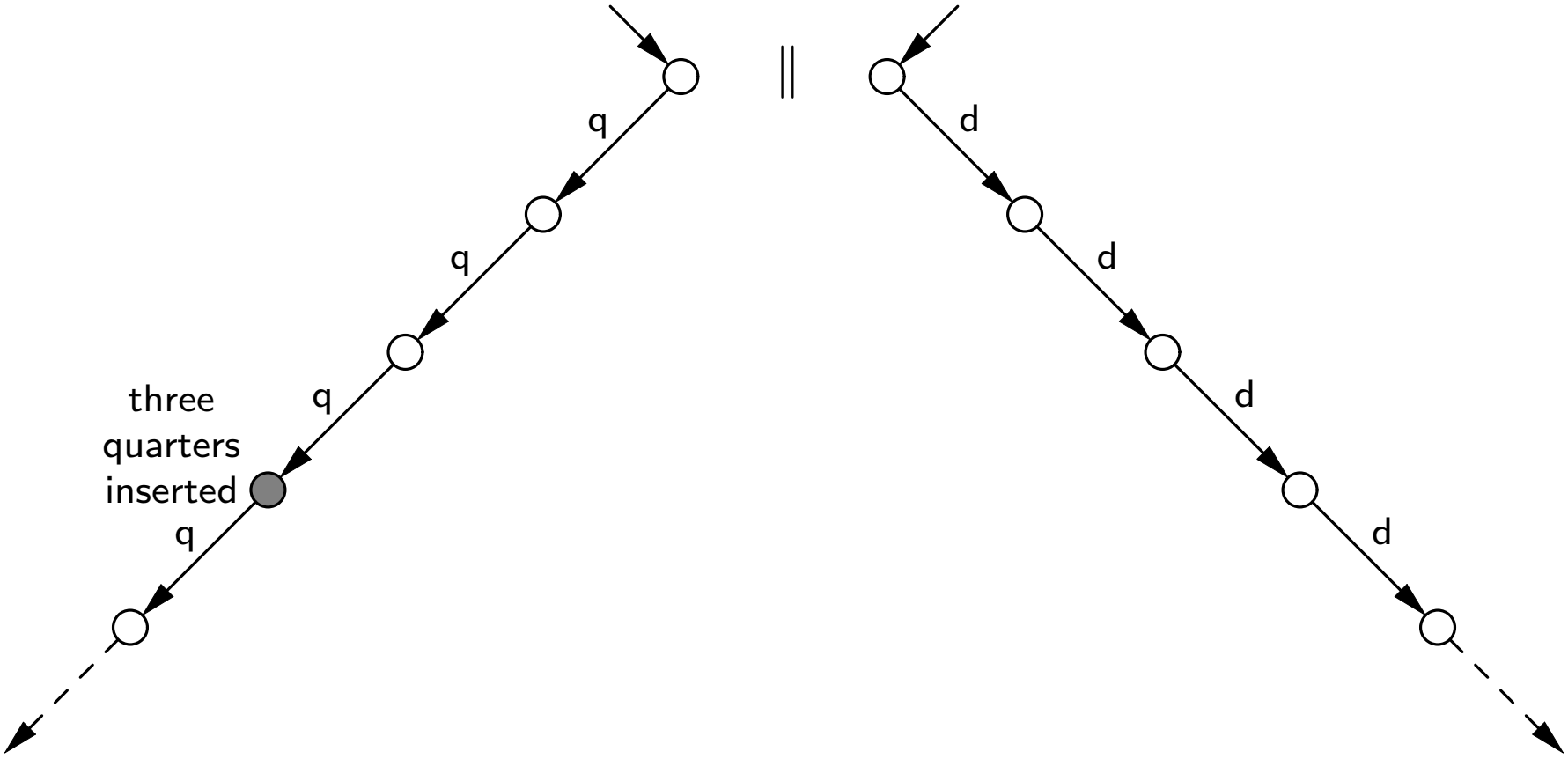
Intuitively, this property holds, when assuming *justness*.

Transition system of example

Transition system of example

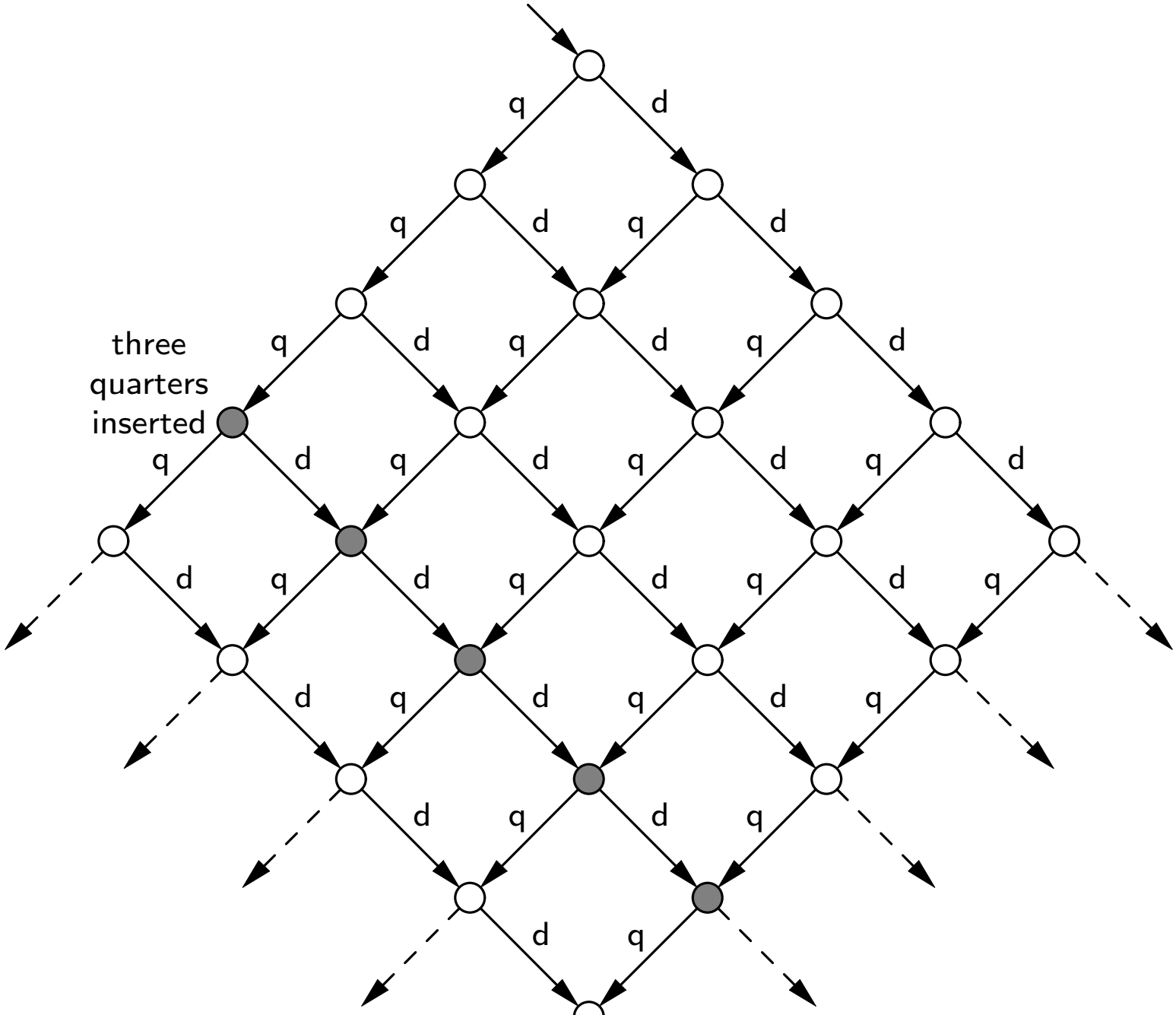


Transition system with success states



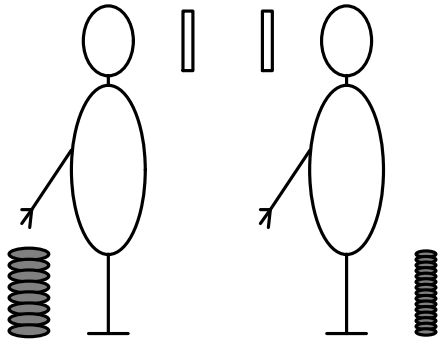
Transition system with success states

$=$

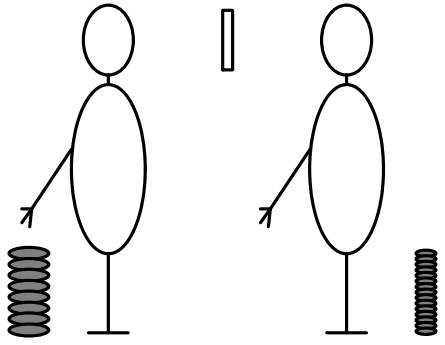


Concurrency versus competition

Concurrency:



Competition:



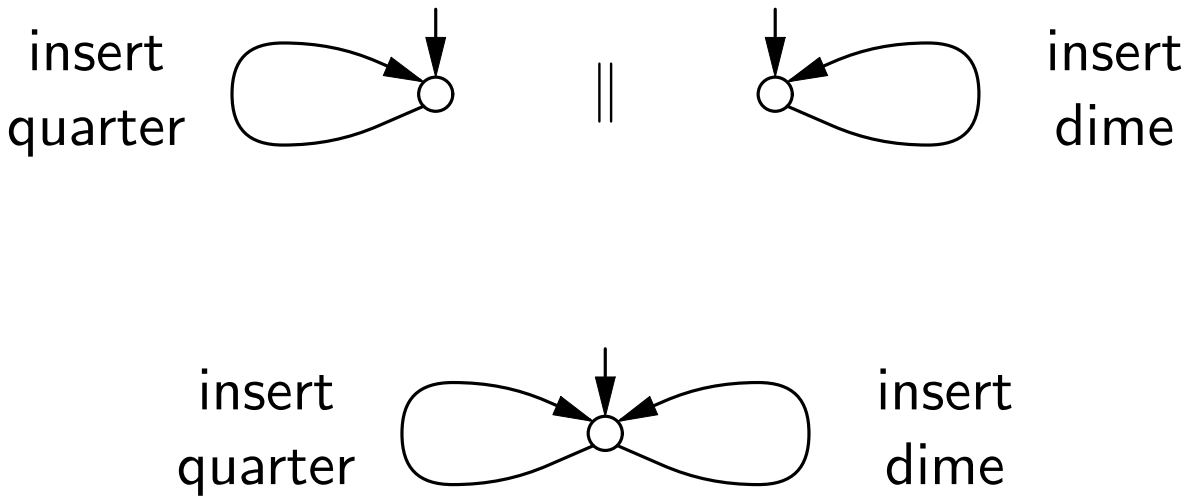
A liveness property: at least 3 quarters will be inserted.

When assuming *justness*
this property holds for the *concurrency* example,

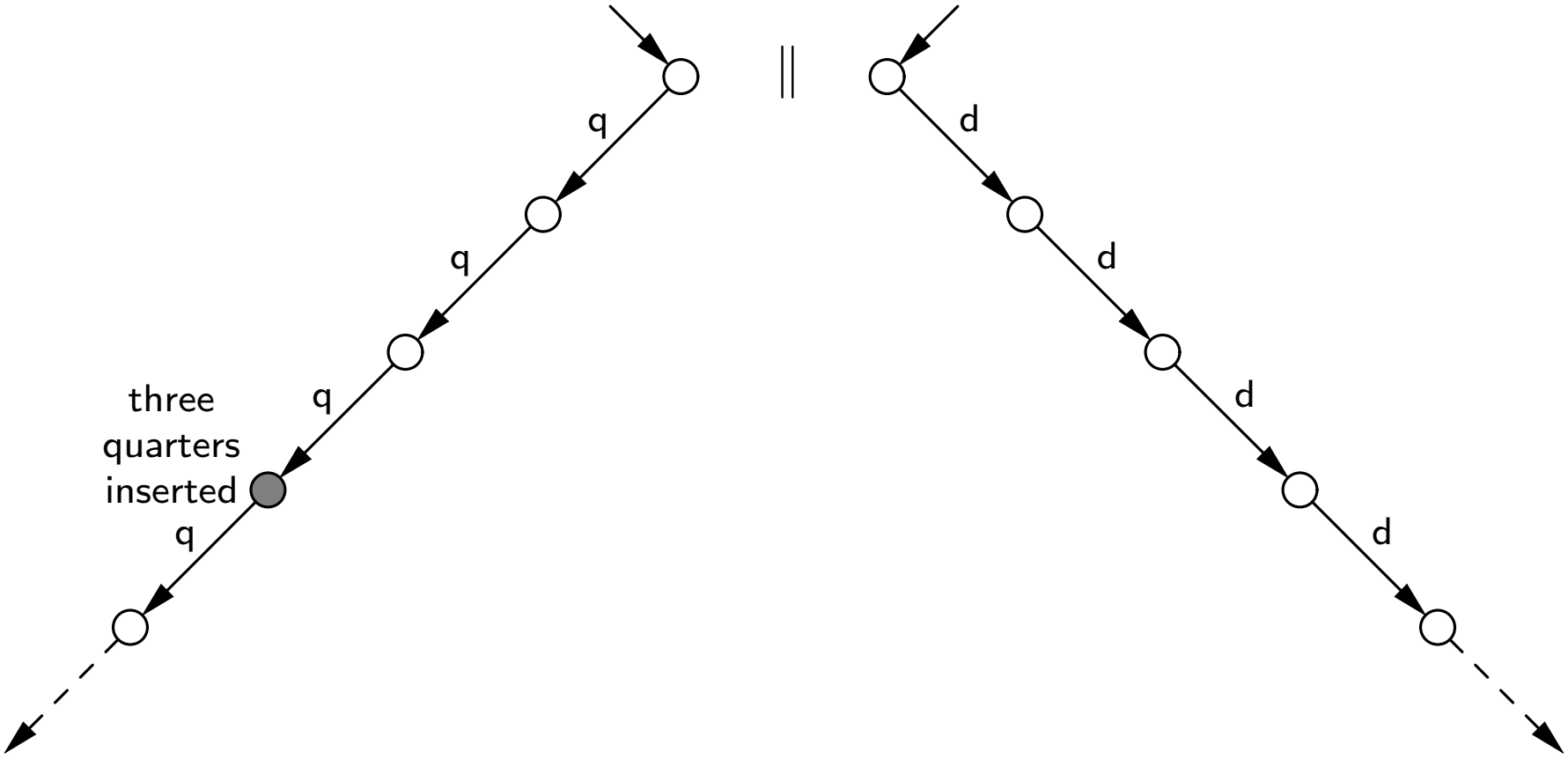
but not for the *competition* example.

Transition system of example

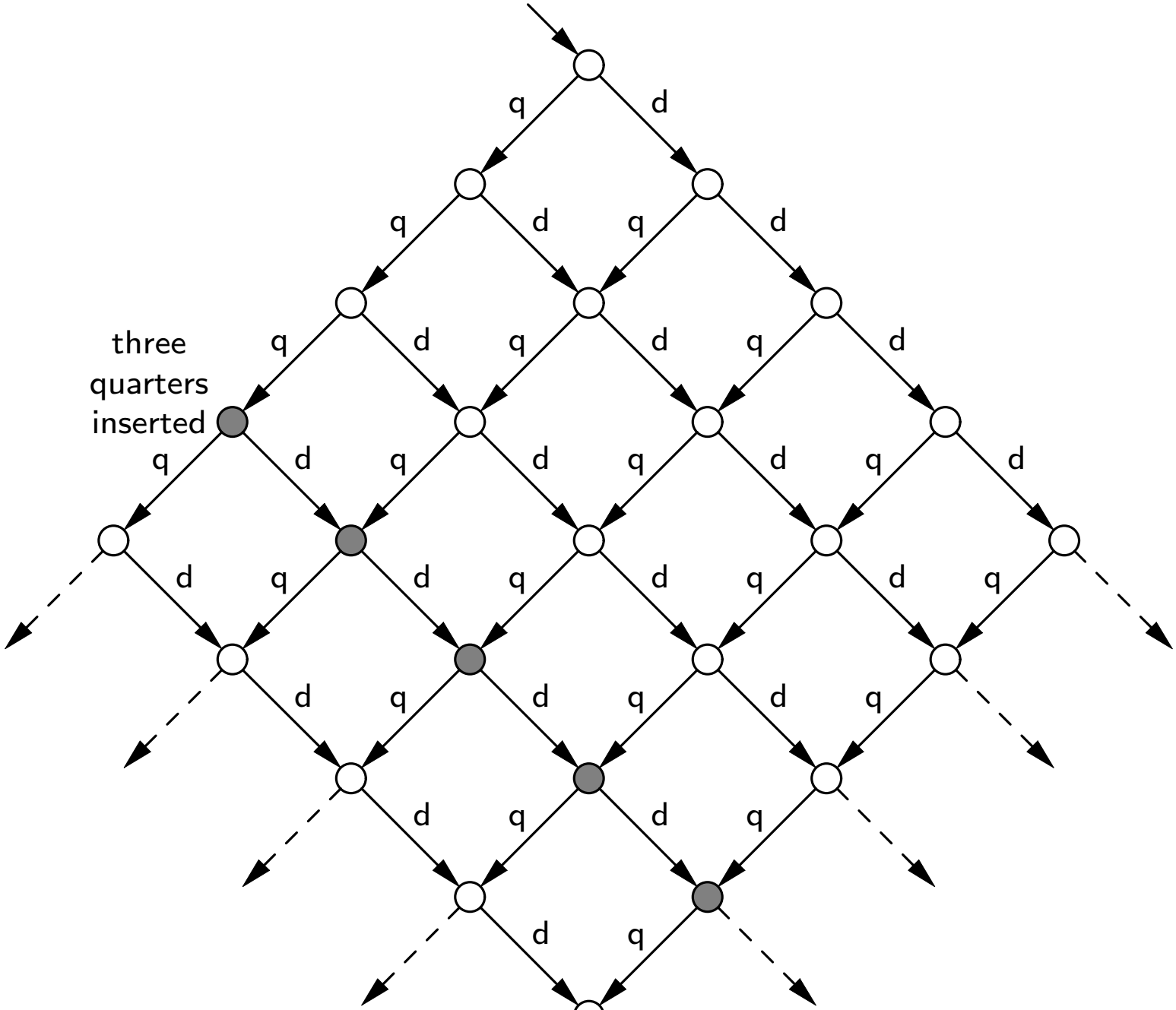
Transition system of example



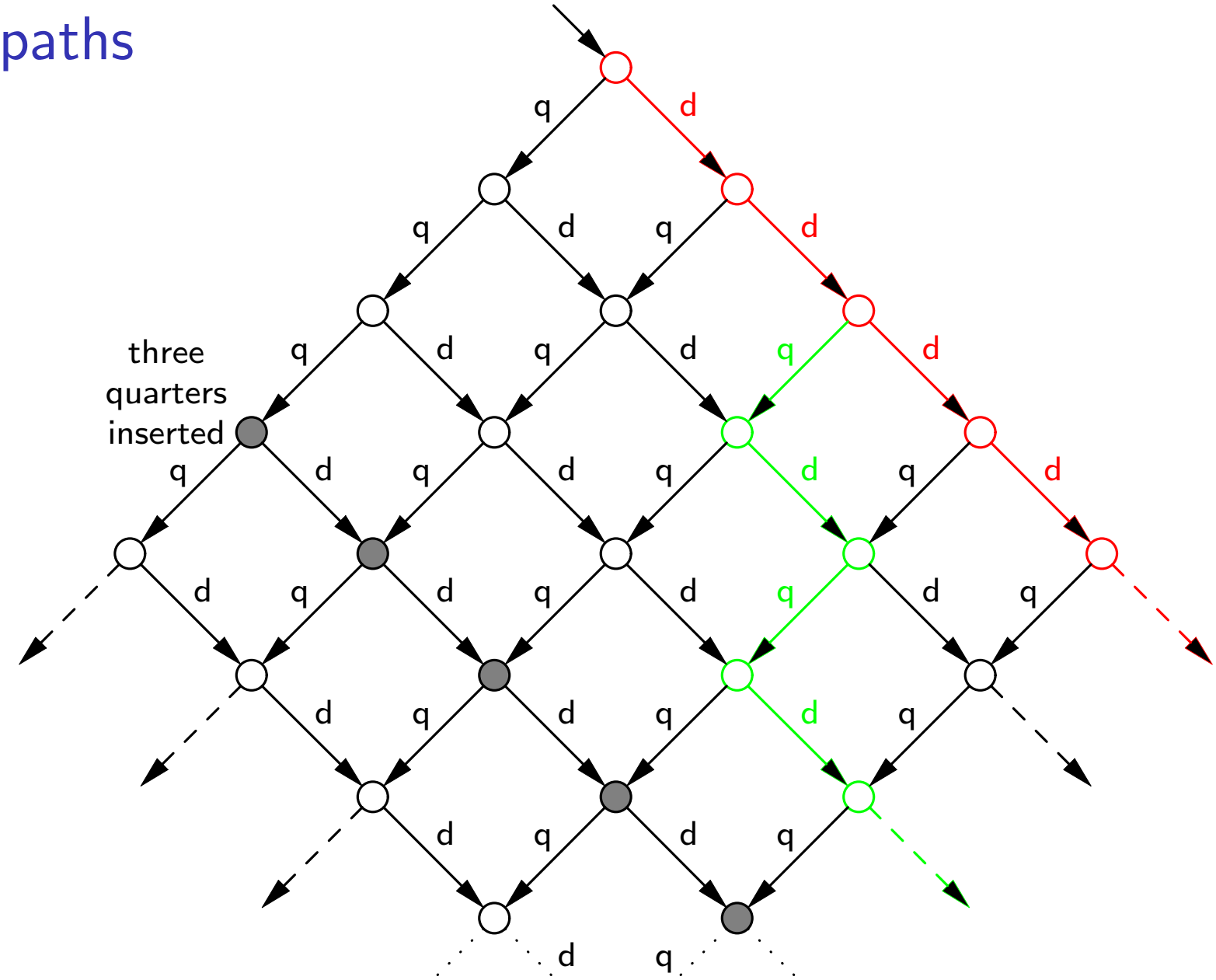
Transition system with success states



Transition system with success states



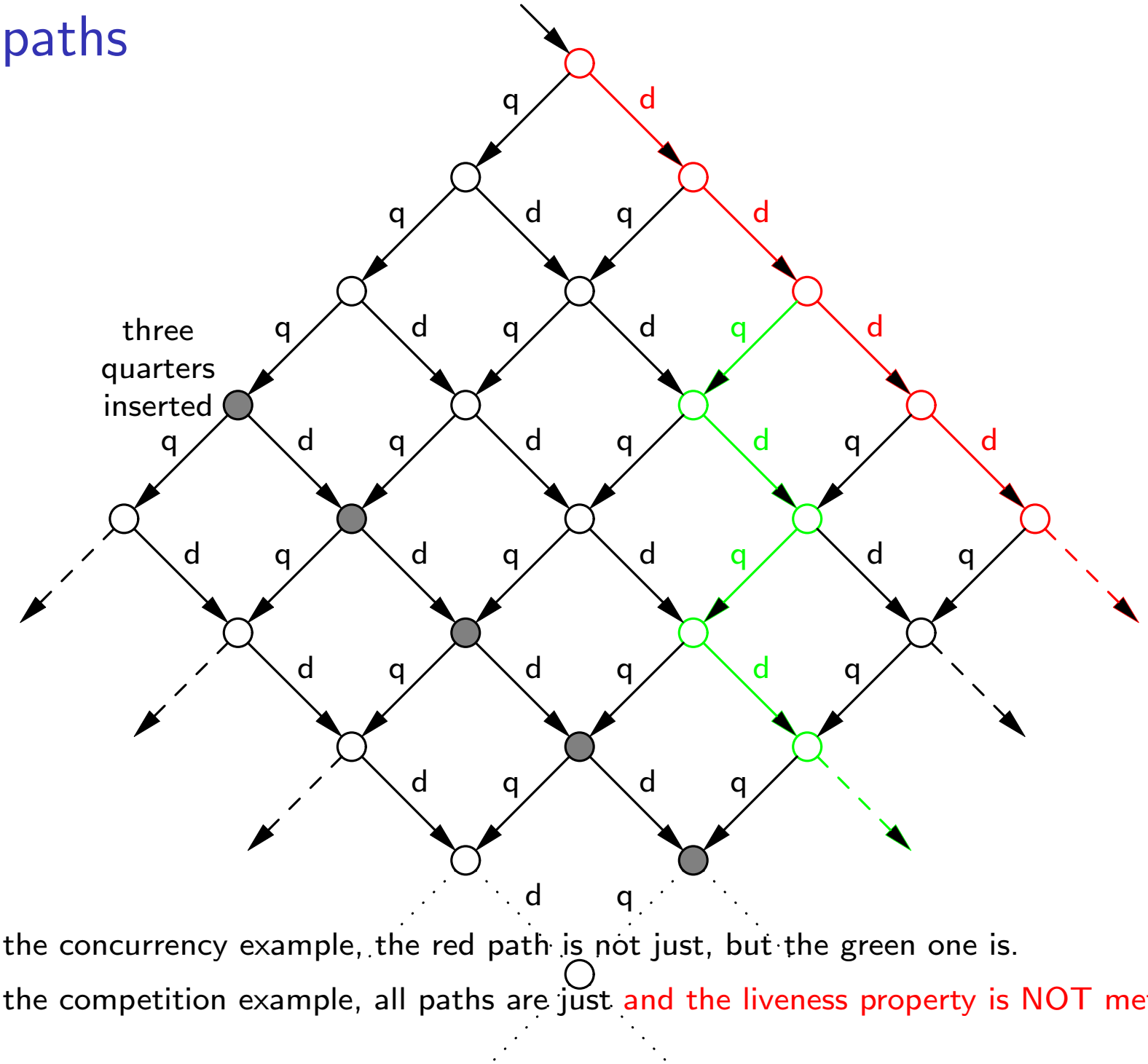
Just paths



In the concurrency example, the red path is not just, but the green one is.

In the competition example, all paths are just

Just paths



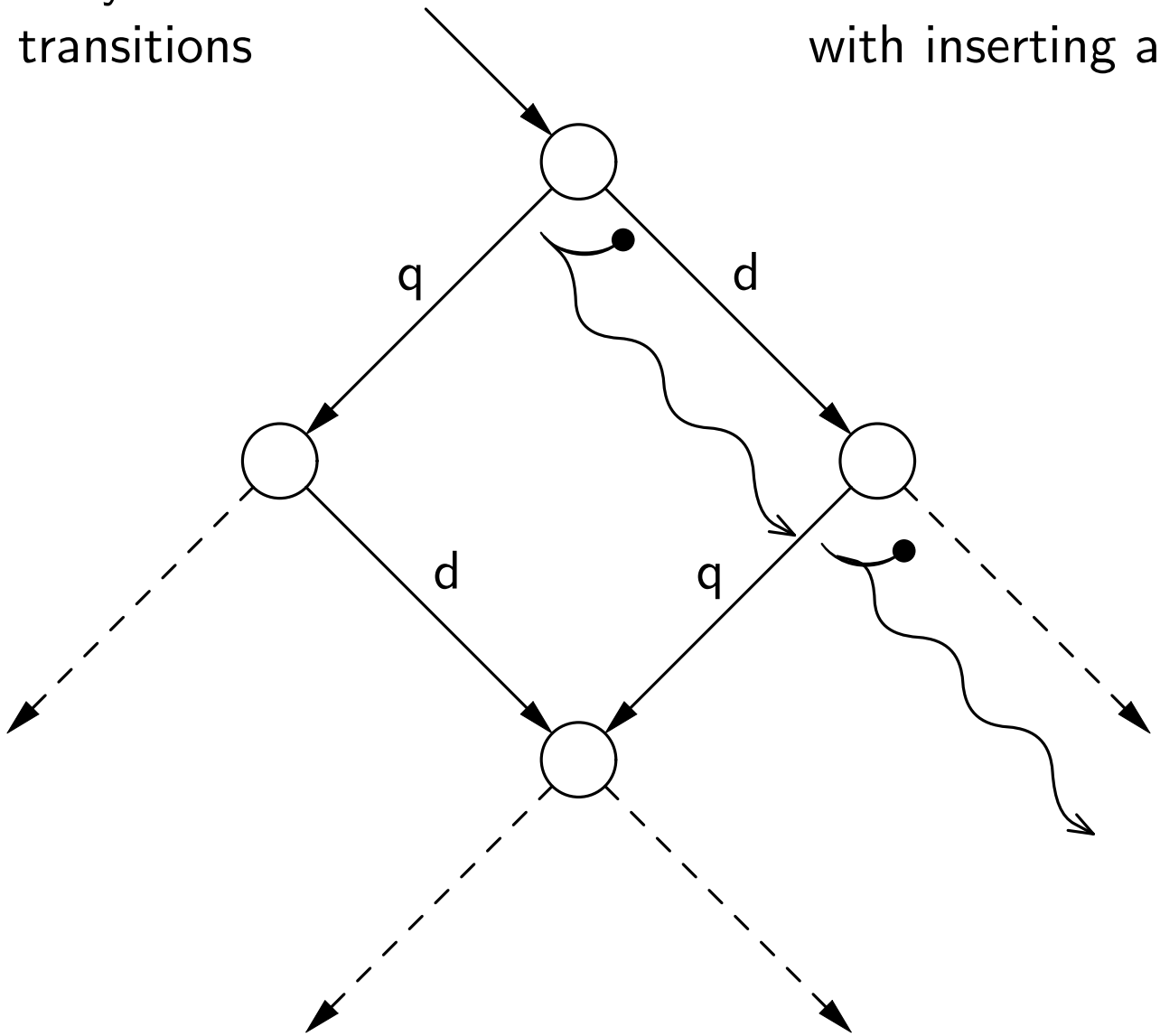
In the concurrency example, the red path is not just, but the green one is.

In the competition example, all paths are just and the liveness property is NOT met.

Transition systems with successors

Transition systems
plus a ternary relation
between transitions

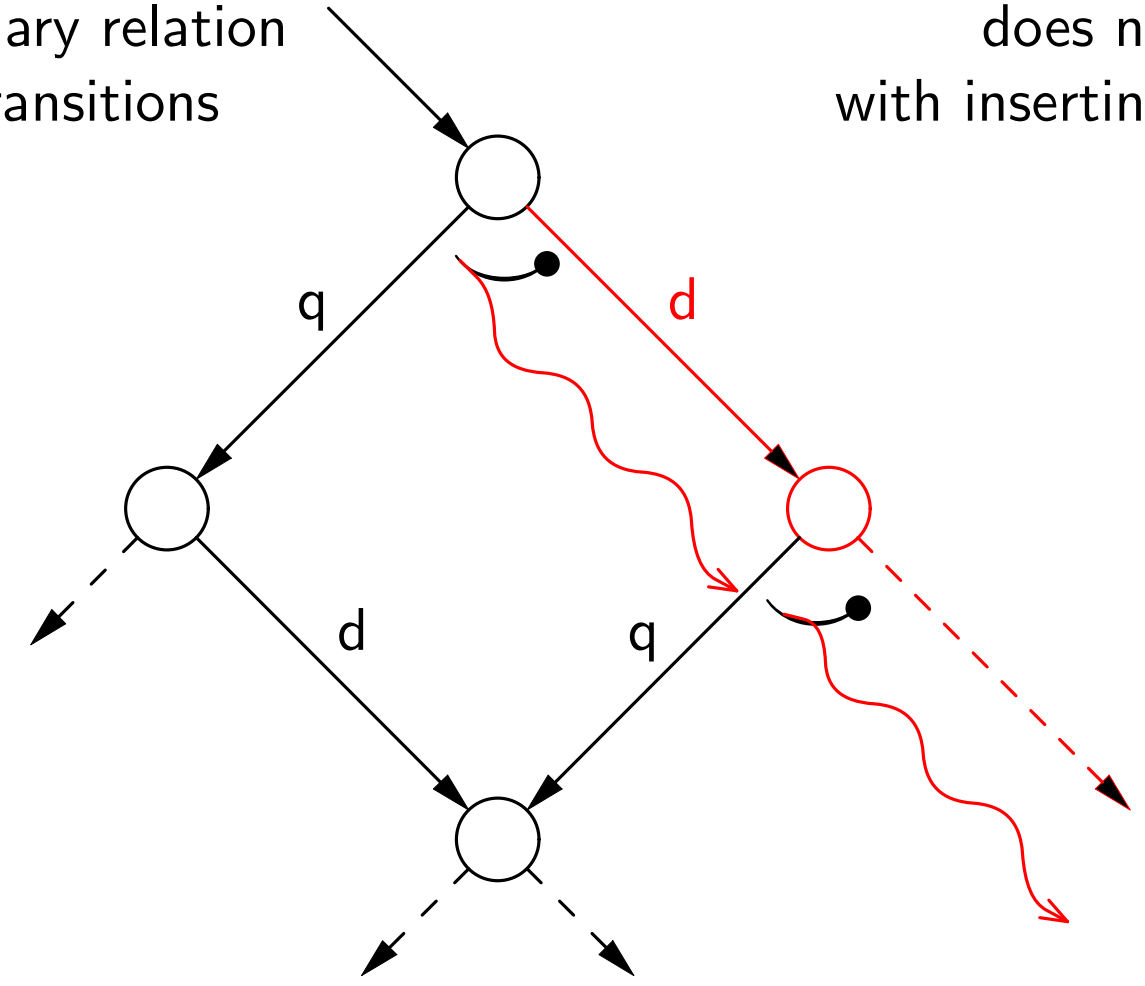
inserting a dime
does not interfere
with inserting a quarter



Formalising Justness

Transition systems
plus a ternary relation
between transitions

inserting a dime
does not interfere
with inserting a quarter



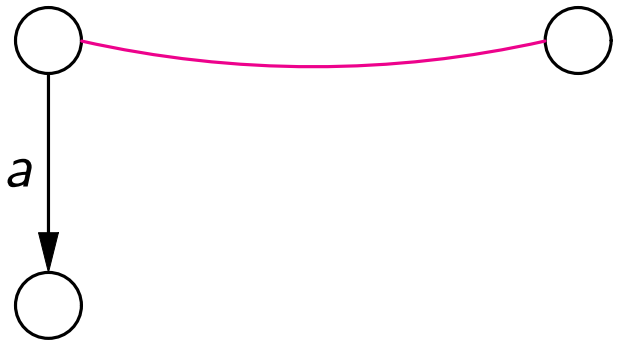
Justness: The system never follows a $\rightarrow$-path
that induces an infinite $\rightsquigarrow$-sequence.

Bisimulation equivalence

To show that two systems have the same properties, one traditionally constructs a *bisimulation* between them. This is a relation $\mathcal{R}$ between their states, such that

- The initial states are related:

- The *transfer property* holds:

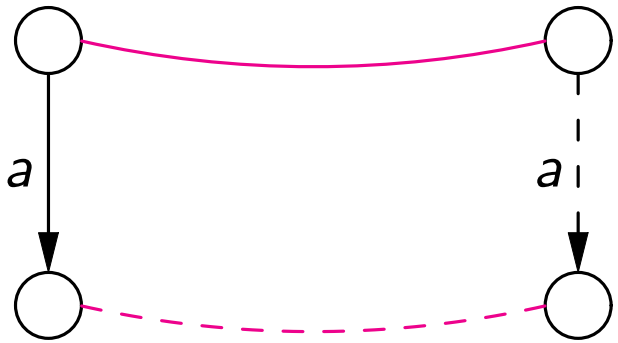


Bisimulation equivalence

To show that two systems have the same properties, one traditionally constructs a *bisimulation* between them. This is a relation $\mathcal{R}$ between their states, such that

- The initial states are related:

- The *transfer property* holds:

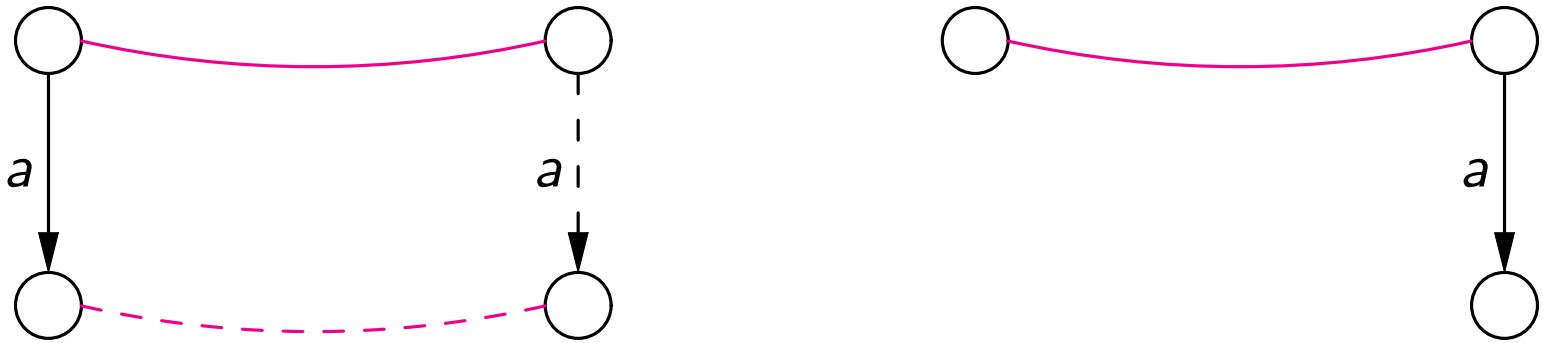


Bisimulation equivalence

To show that two systems have the same properties, one traditionally constructs a *bisimulation* between them. This is a relation $\mathcal{R}$ between their states, such that

- The initial states are related:

- The *transfer property* holds:

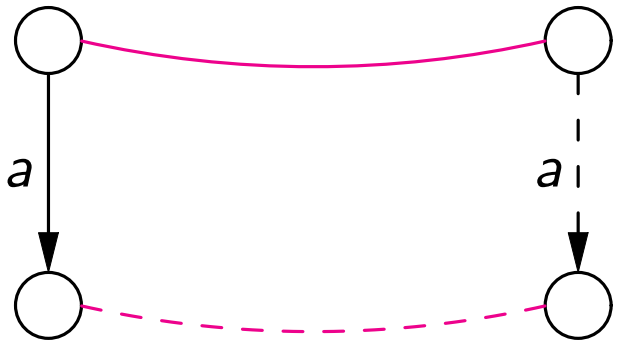


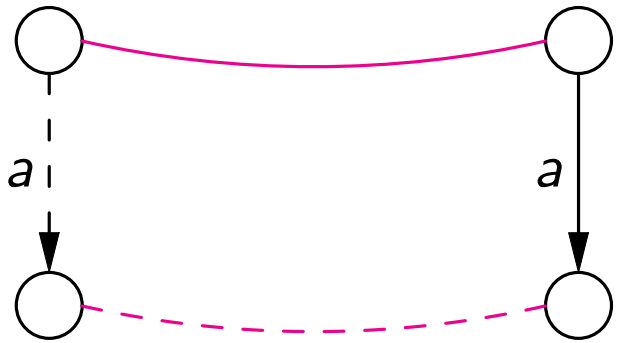
Bisimulation equivalence

To show that two systems have the same properties, one traditionally constructs a *bisimulation* between them. This is a relation $\mathcal{R}$ between their states, such that

- The initial states are related:

- The *transfer property* holds:



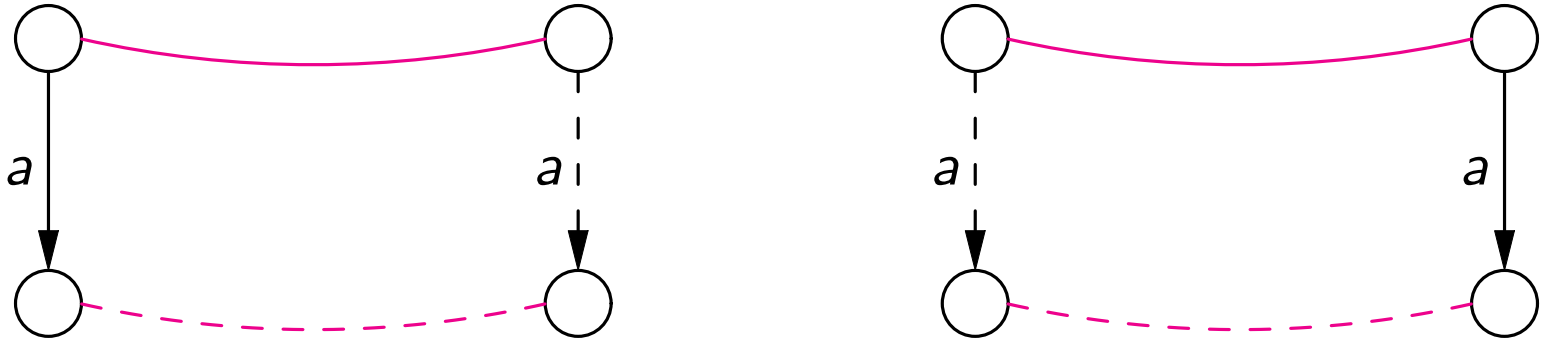


Bisimulation equivalence

To show that two systems have the same properties, one traditionally constructs a *bisimulation* between them. This is a relation $\mathcal{R}$ between their states, such that

- The initial states are related:

- The *transfer property* holds:

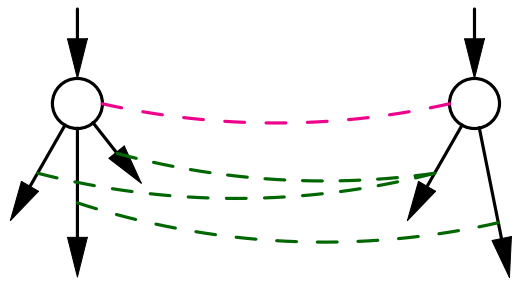


To preserve justness we need a form of bisimulation that also preserves $\rightsquigarrow$.

Enabling preserving bisimulation equivalence

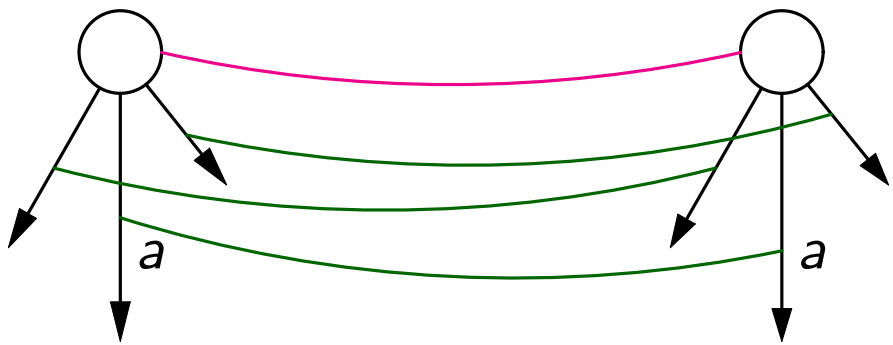
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

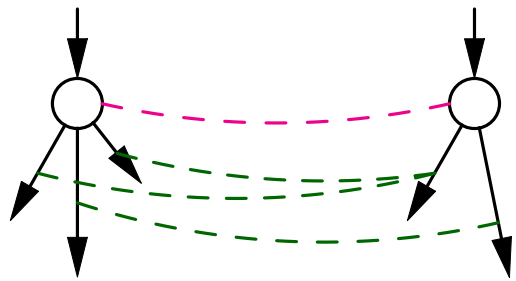
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

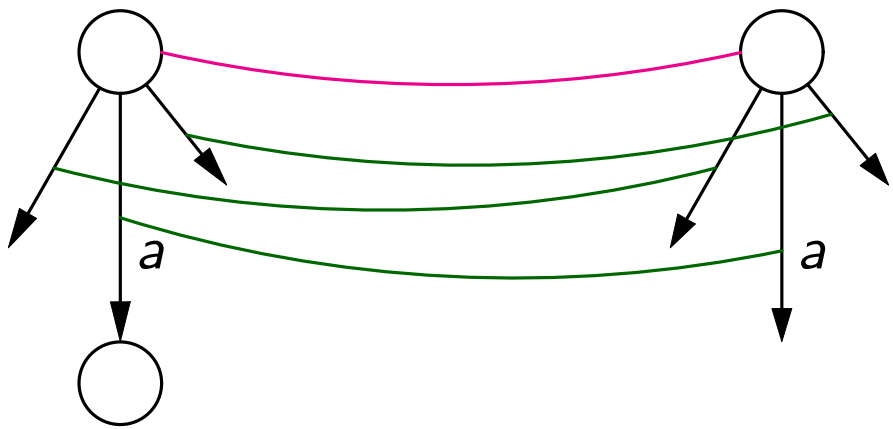
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

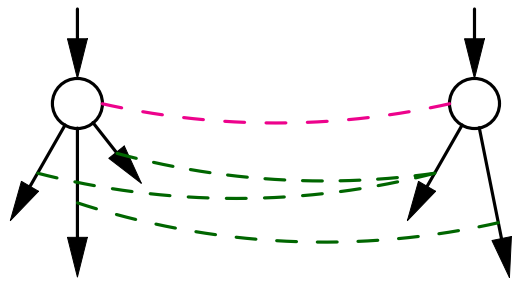
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

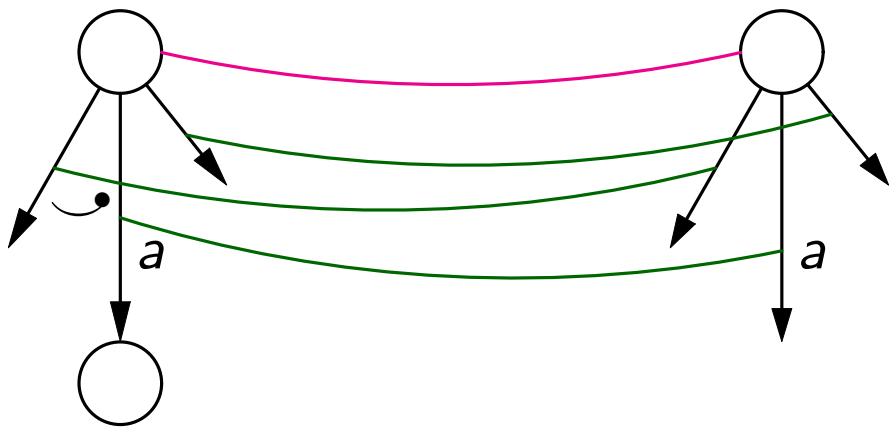
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

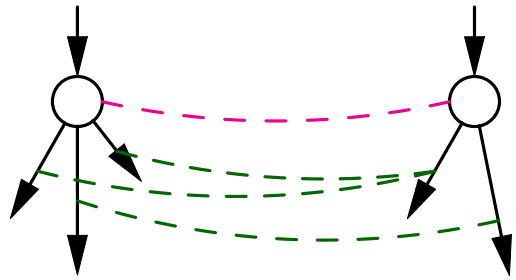
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

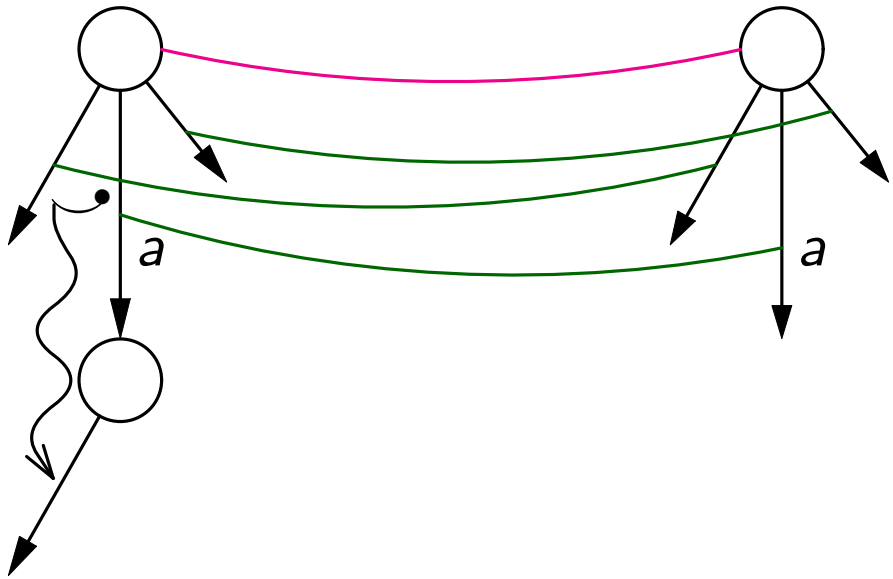
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

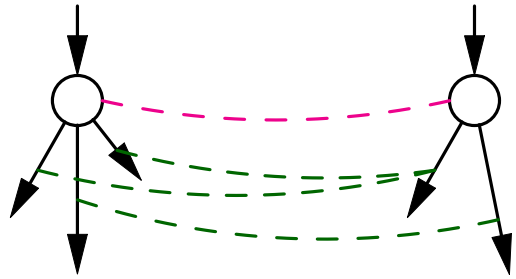
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

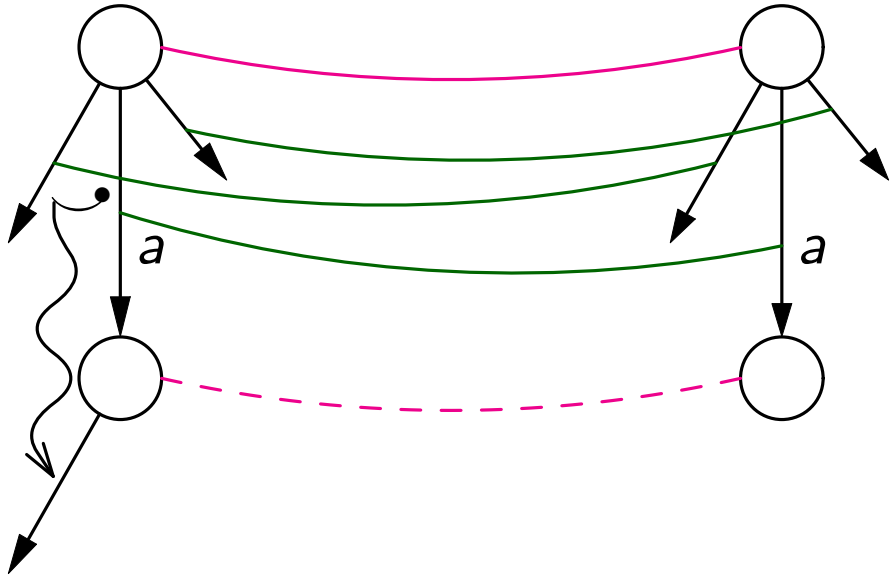
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

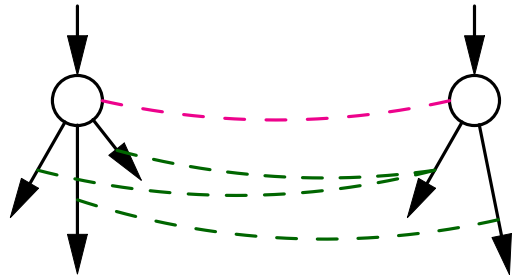
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

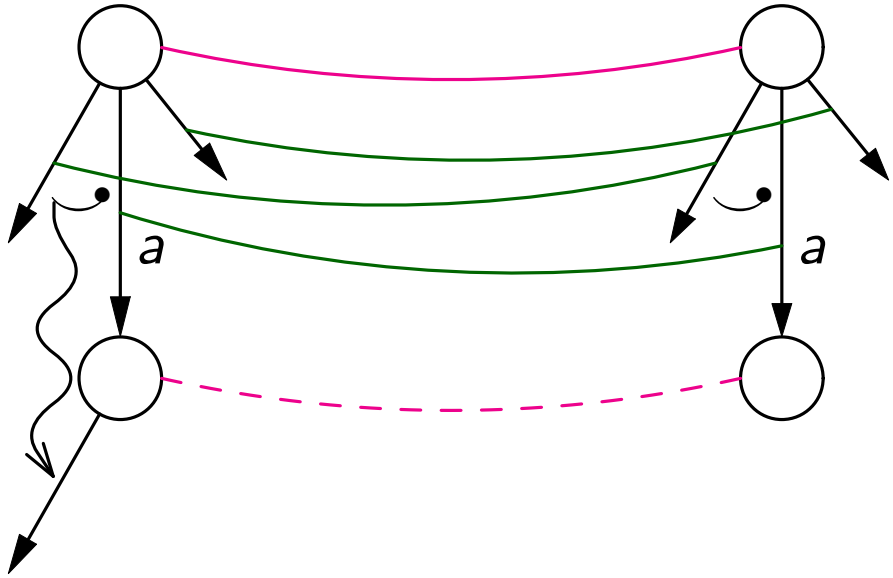
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

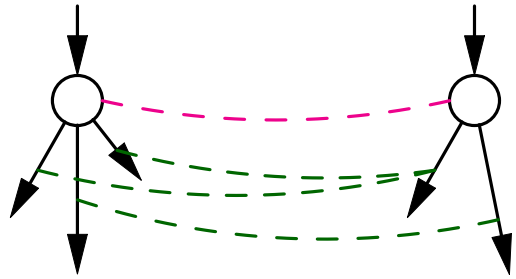
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

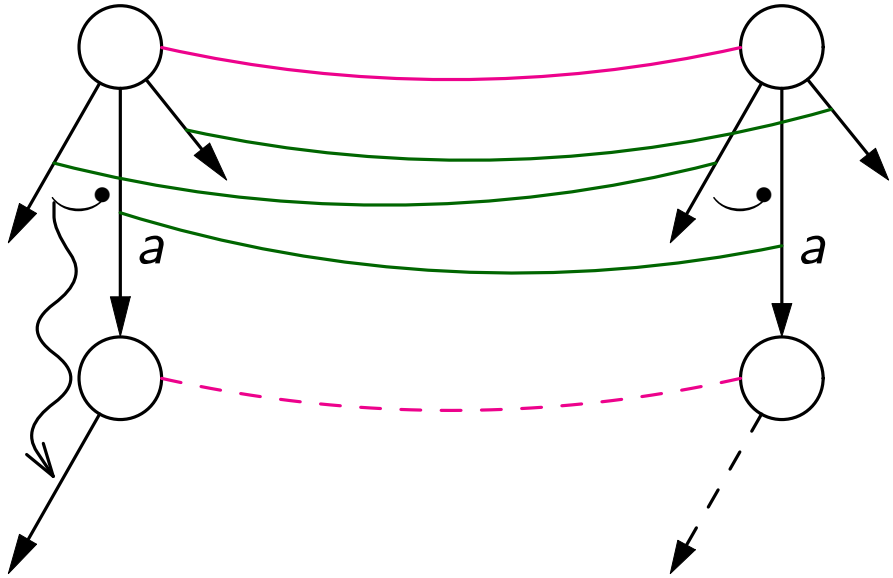
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

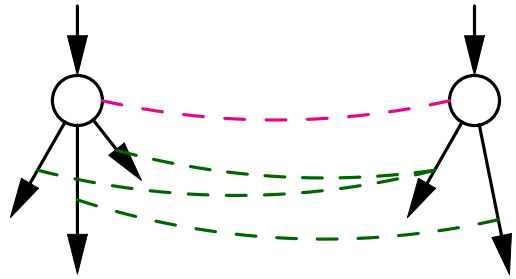
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

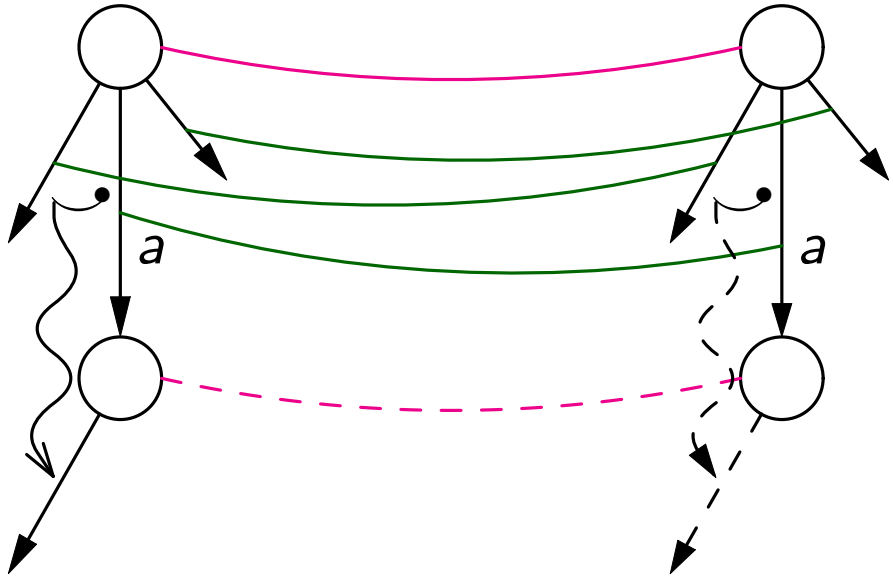
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

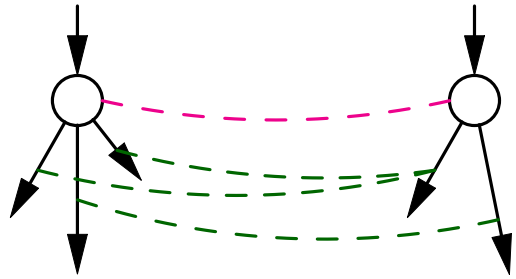
- The *transfer property* holds:



Enabling preserving bisimulation equivalence

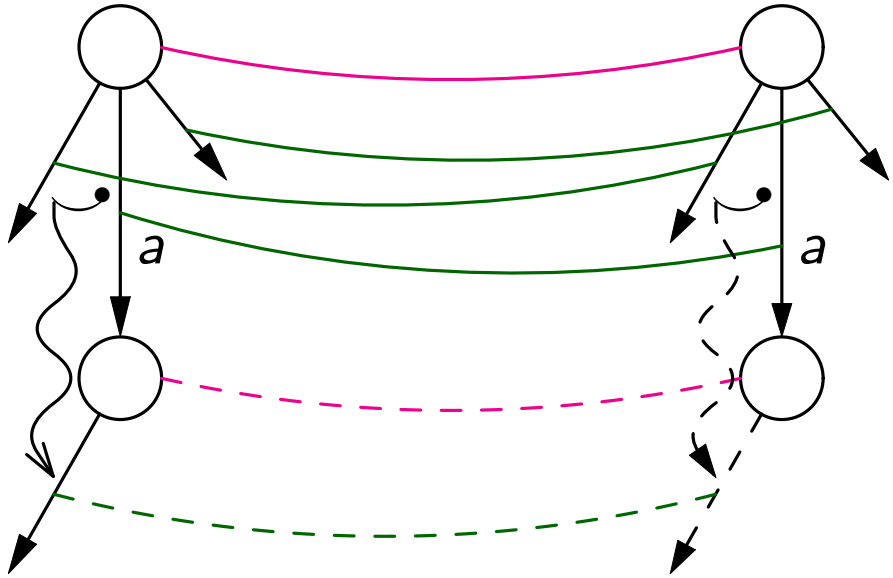
To show that two systems have the same liveness properties, one constructs an *enabling preserving bisimulation* between them.

This is a relation $\mathcal{R}$ between their states, where each pair of related states is equipped with a relation R between their enabled transitions, such that



- The initial states are related:

- The *transfer property* holds:



EP bisimulation is useful

This notion of bisimulation has the properties we want:

- it preserves liveness properties
  under the assumption of justness;
- it induces an equivalence relation,
- which is a congruence for parallel composition (and other operators), thus allowing compositional reasoning.

These properties, and others, are proven in a paper that will be presented at CONCUR 2021, later this month.
http://theory.stanford.edu/~rvg/abstracts#157

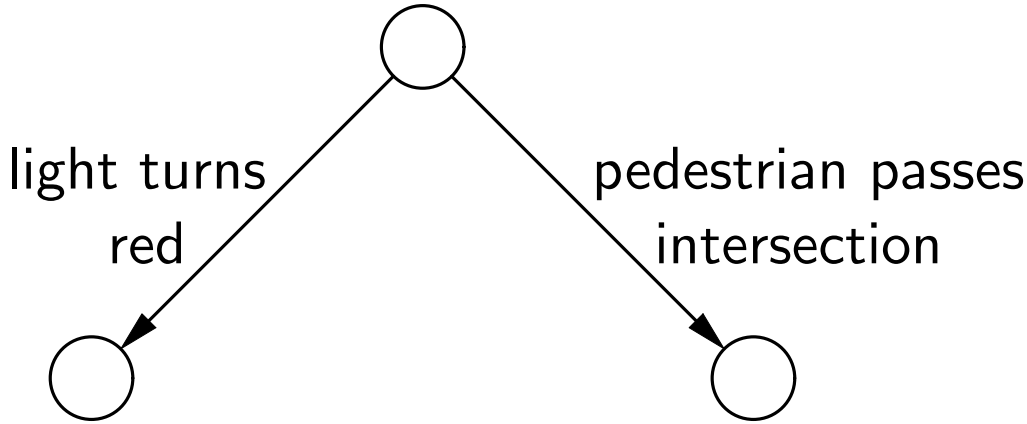
Conclusion

This new bisimulation can be used
to prove implementations equivalent to specifications
in such a way that – under the assumption of justness –

all liveness properties of the specification
also hold for the implementation.

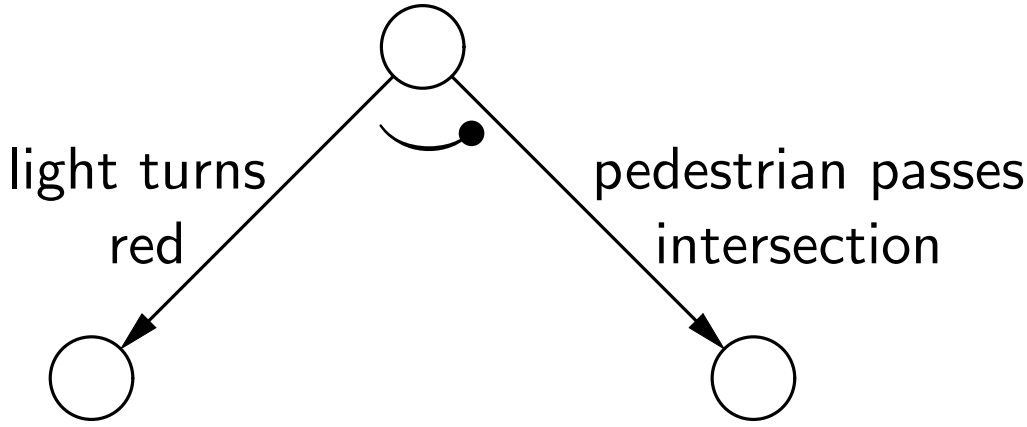
Example of an asymmetric concurrency relation

obedient pedestrian approaching a green traffic light



Example of an asymmetric concurrency relation

obedient pedestrian approaching a green traffic light



Example of an asymmetric concurrency relation

obedient pedestrian approaching a green traffic light