Techniques and Tools for the Analysis of Timed Workflows

Jiri Srba

Department of Computer Science, Aalborg University,
Selma Lagerlöfs Vej 300, 9220 Aalborg East, Denmark

NWPT'15, Iceland, October 22nd, 2015

Joint work with Peter G. Jensen, José A. Mateo and Mathias G. Sørensen.

Workflow Definition

Workflows [Wikipedia]

A workflow consists of

- an orchestrated and repeatable pattern of business activity
- enabled by the systematic organization of resources into processes that transform materials, provide services, or process information.

Examples:

- Car assembly line.
- Insurance claim.
- Blood transfusion.

Workflow Definition

Workflows [Wikipedia]

A workflow consists of

- an orchestrated and repeatable pattern of business activity
- enabled by the systematic organization of resources into processes that transform materials, provide services, or process information.

Examples:

- Car assembly line.
- Insurance claim.
- Blood transfusion.

All these are examples of time-critical workflows.

There is a need for methods and tools for timed workflow analysis.

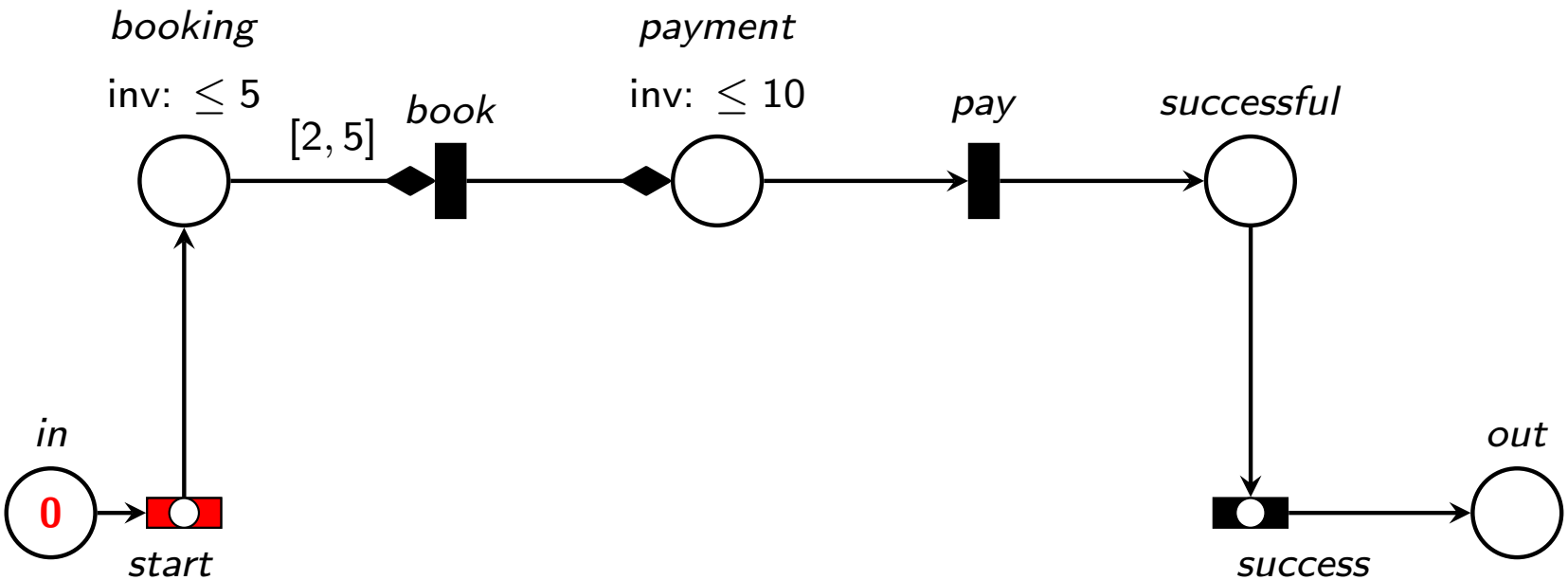
Introduction — Workflow Nets

- Workflow nets by Wil van der Aalst [ICATPN'97] are widely used for workflow modelling.
- Based on Petri nets.
- Abstraction from data, focus on execution flow.
- Early detection of design errors like deadlocks, livelocks and other abnormal behaviour.
- Classical soundness for workflow nets:
  - option to complete,
  - proper termination, and
  - absence of redundant tasks.

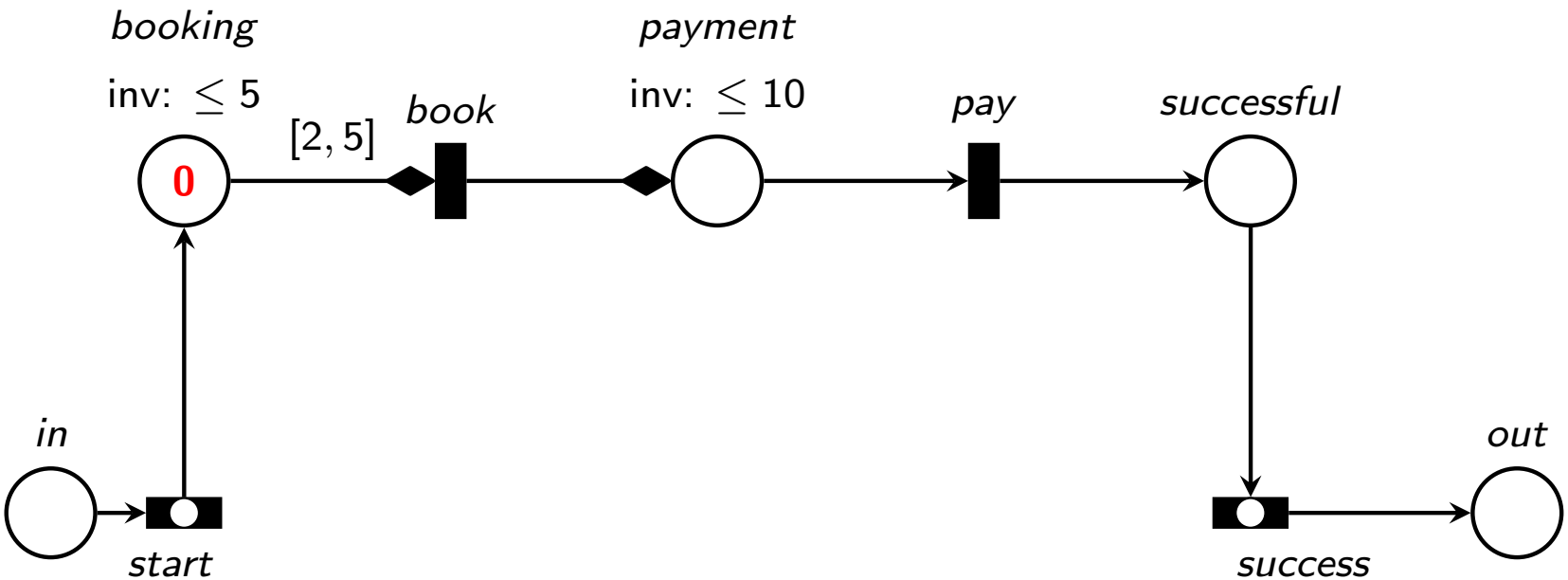
Focus of the Talk

- Theory of workflow nets based on timed-arc Petri nets.
- Definition of soundness and strong soundness.
- Results about decidability/undecidability of soundness.
- Minimum and maximum execution time of workflow nets.
- Integration within the tool TAPAAL and case studies.
- Discrete vs. continuous time.

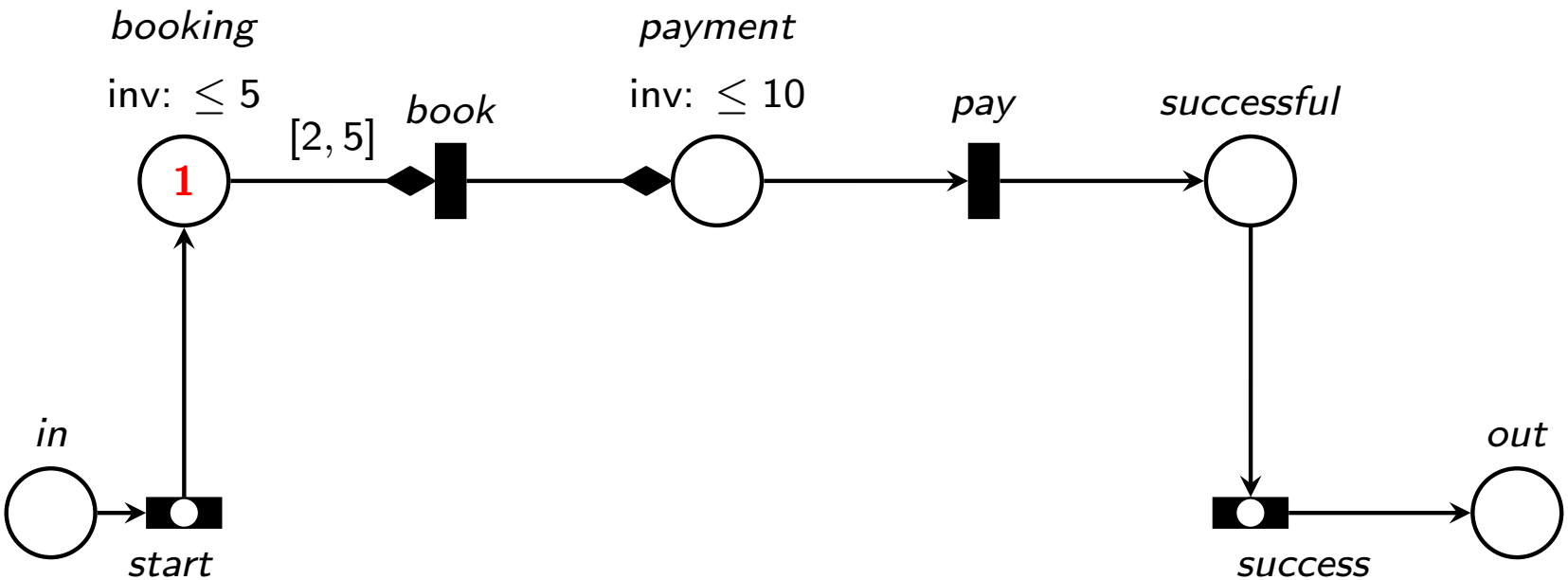
Timed-Arc Petri Net: Booking/Payment Example



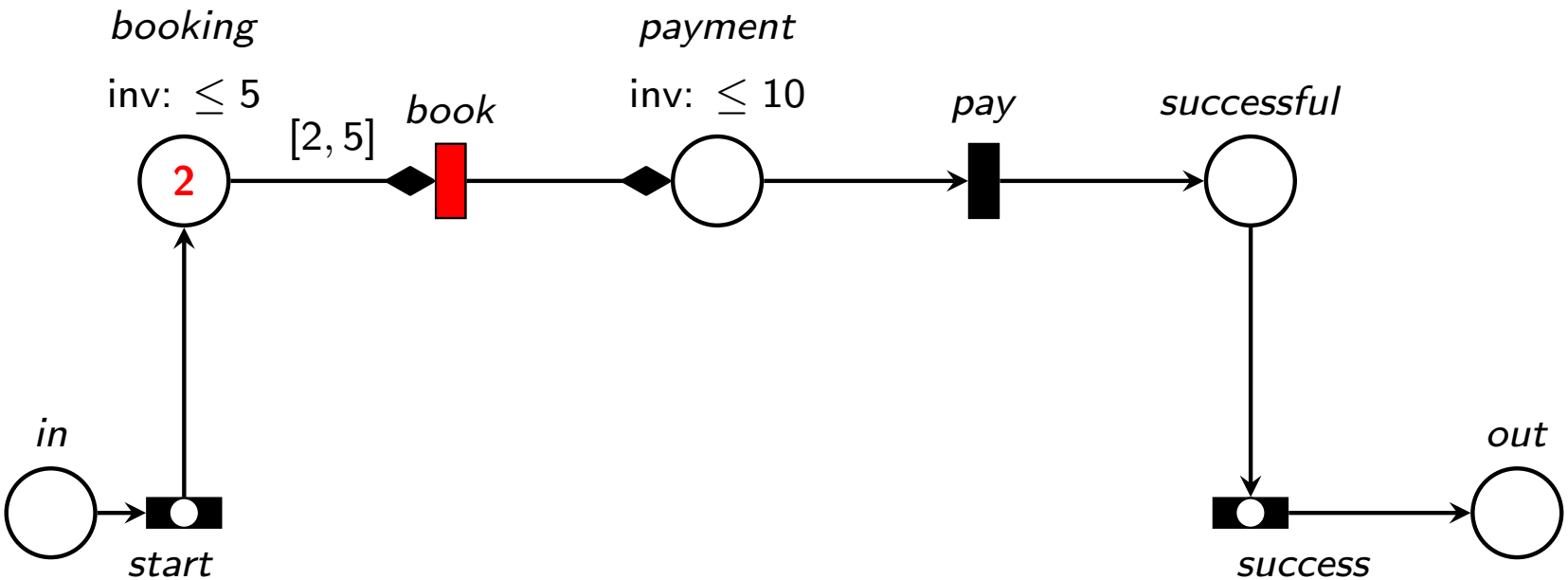
Timed-Arc Petri Net: Booking/Payment Example



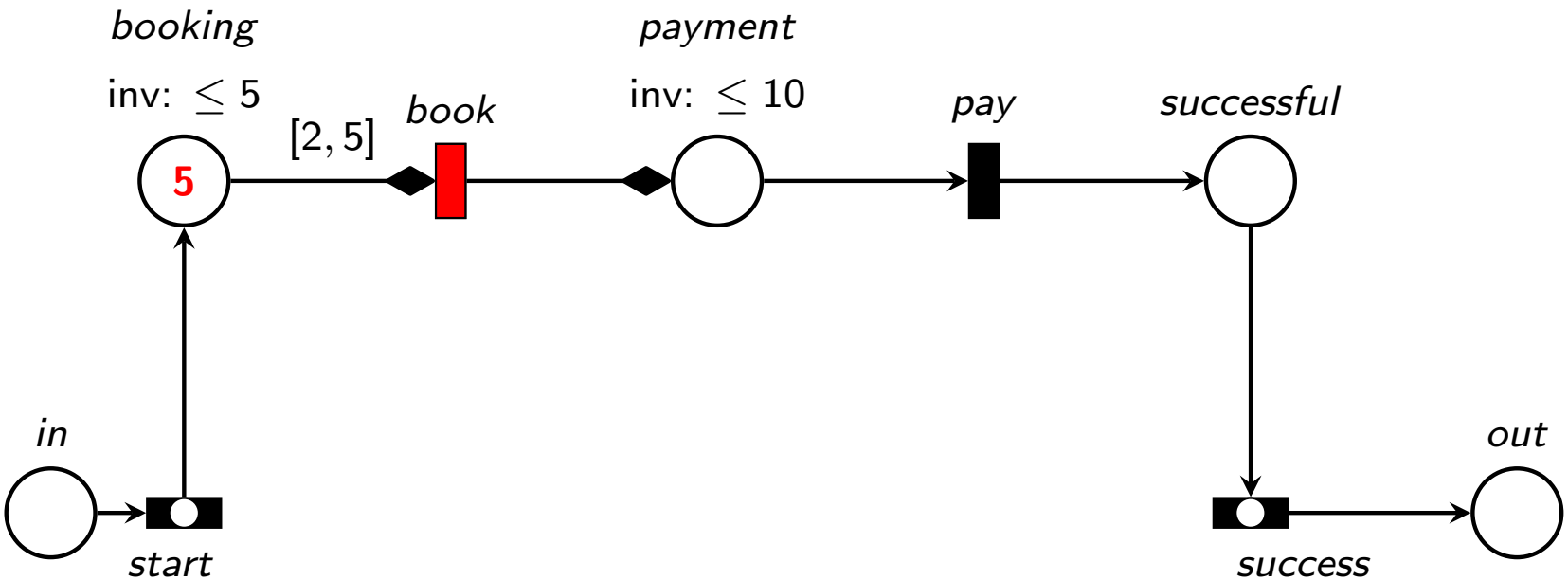
Timed-Arc Petri Net: Booking/Payment Example



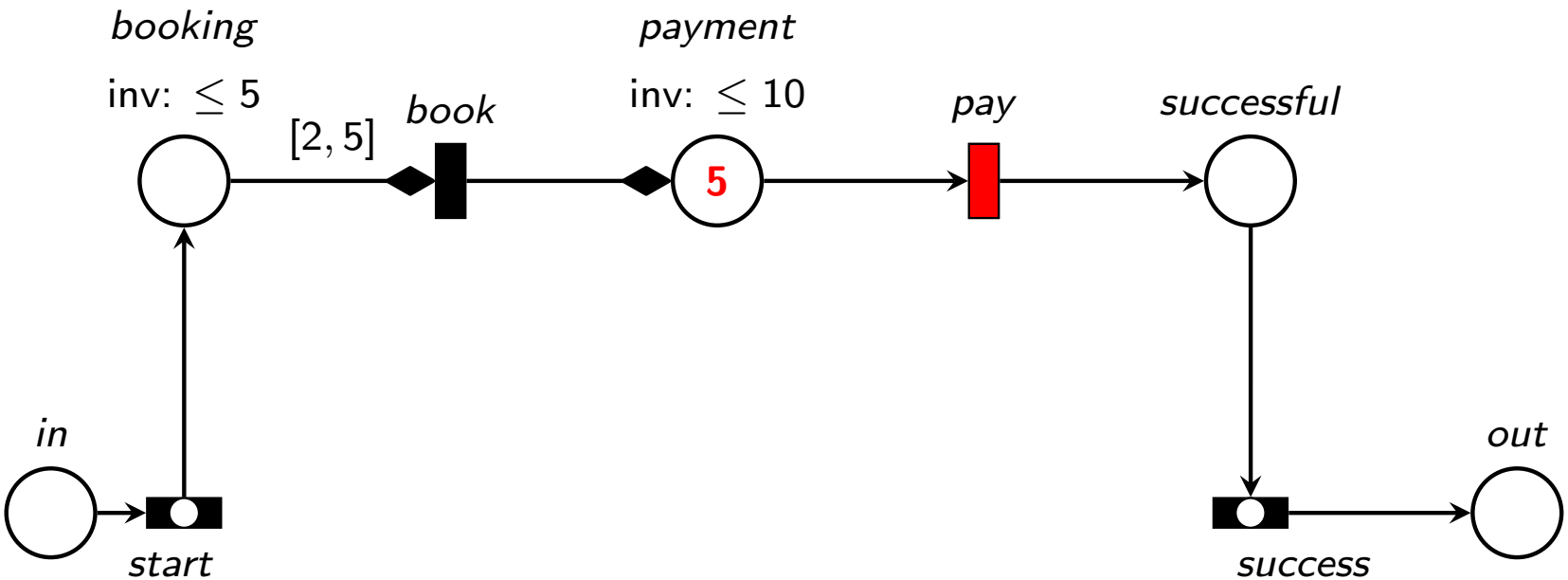
Timed-Arc Petri Net: Booking/Payment Example



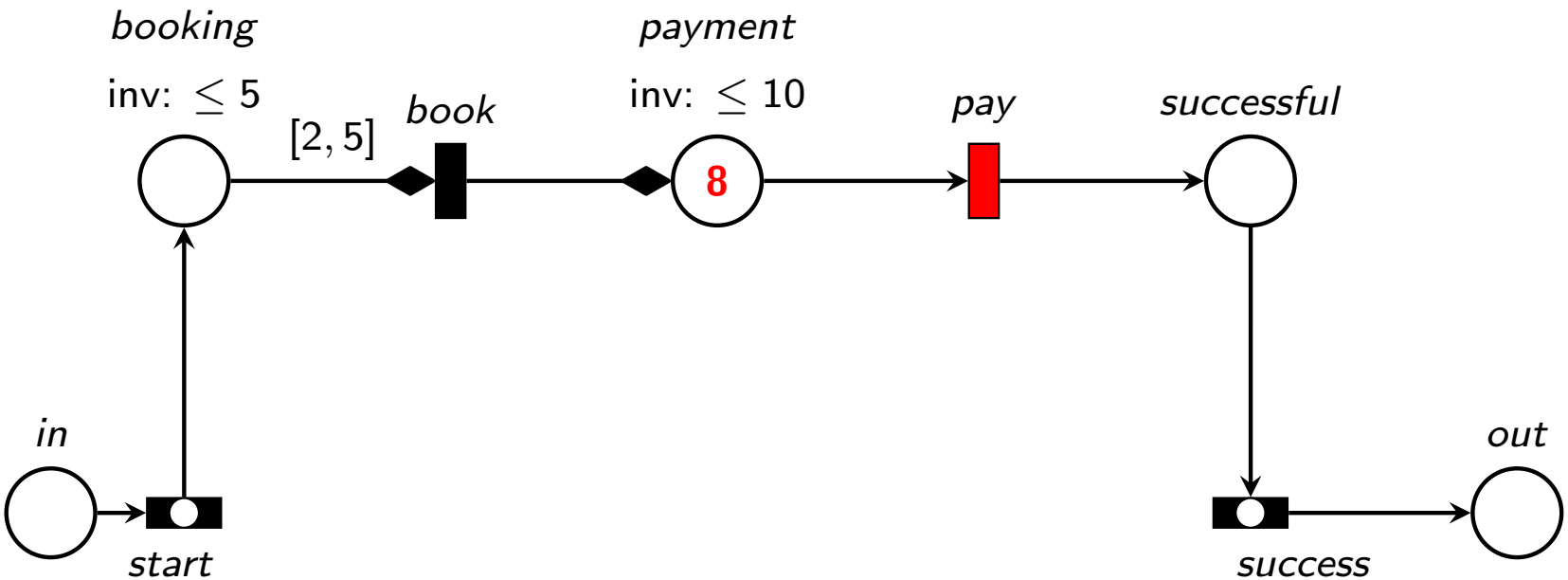
Timed-Arc Petri Net: Booking/Payment Example



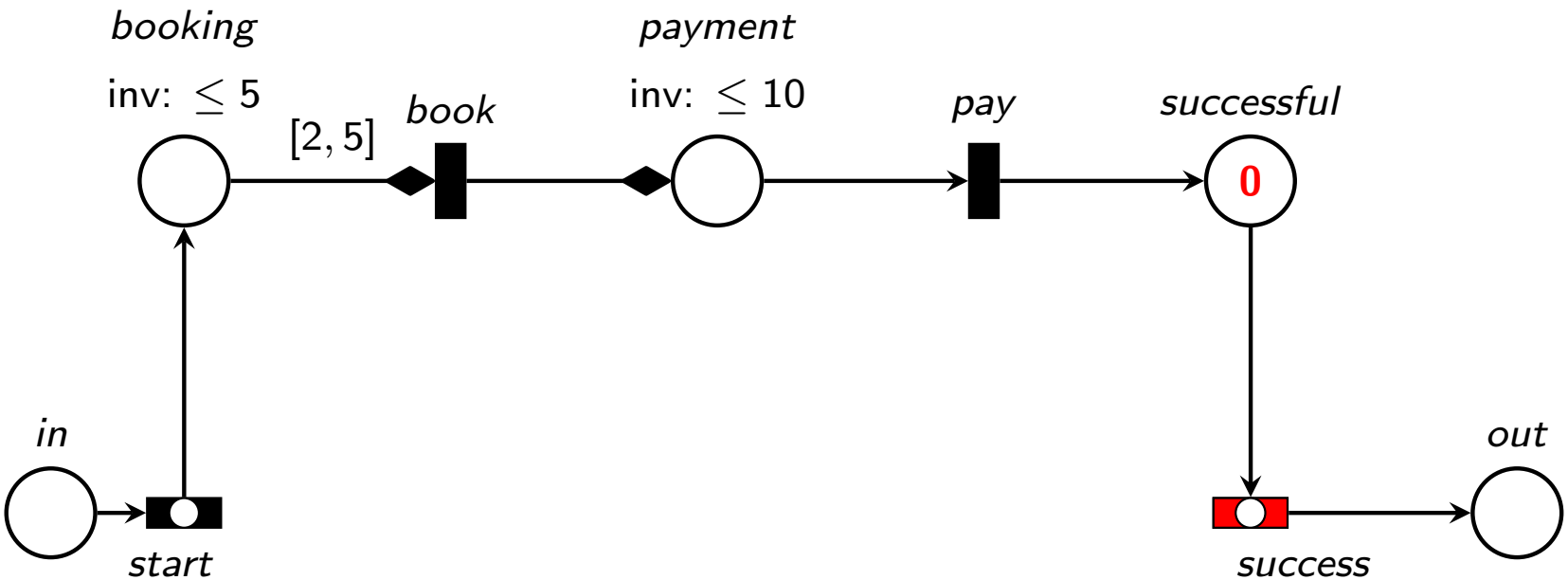
Timed-Arc Petri Net: Booking/Payment Example



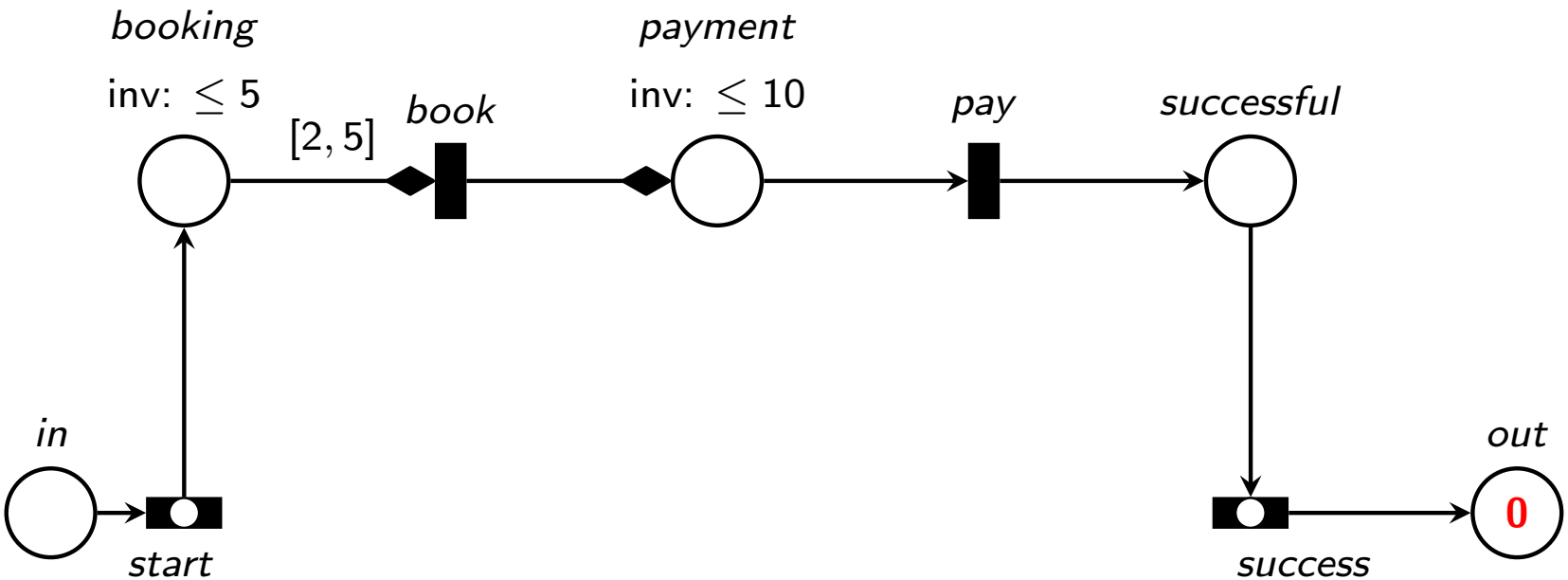
Timed-Arc Petri Net: Booking/Payment Example



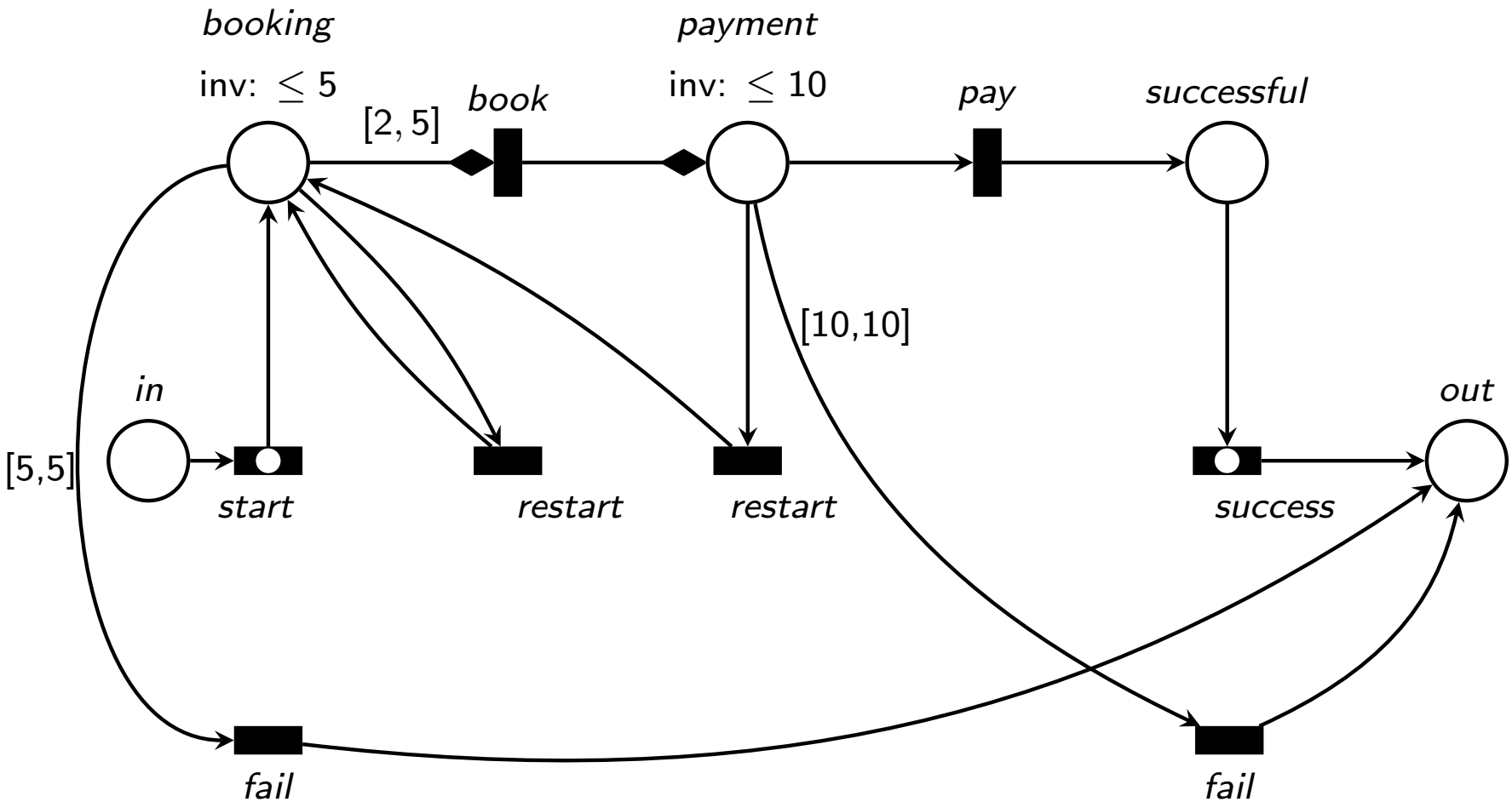
Timed-Arc Petri Net: Booking/Payment Example



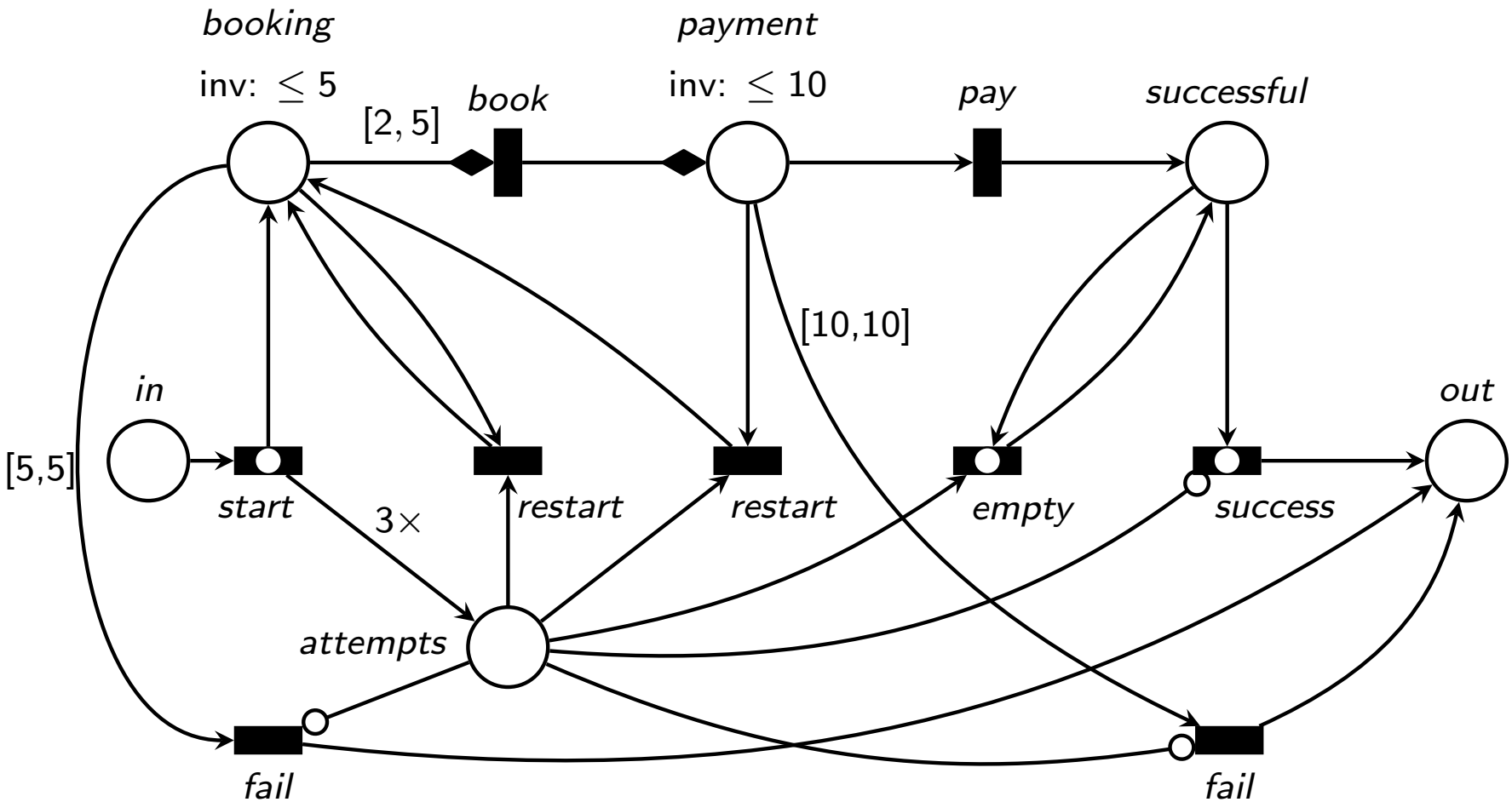
Timed-Arc Petri Net: Booking/Payment Example



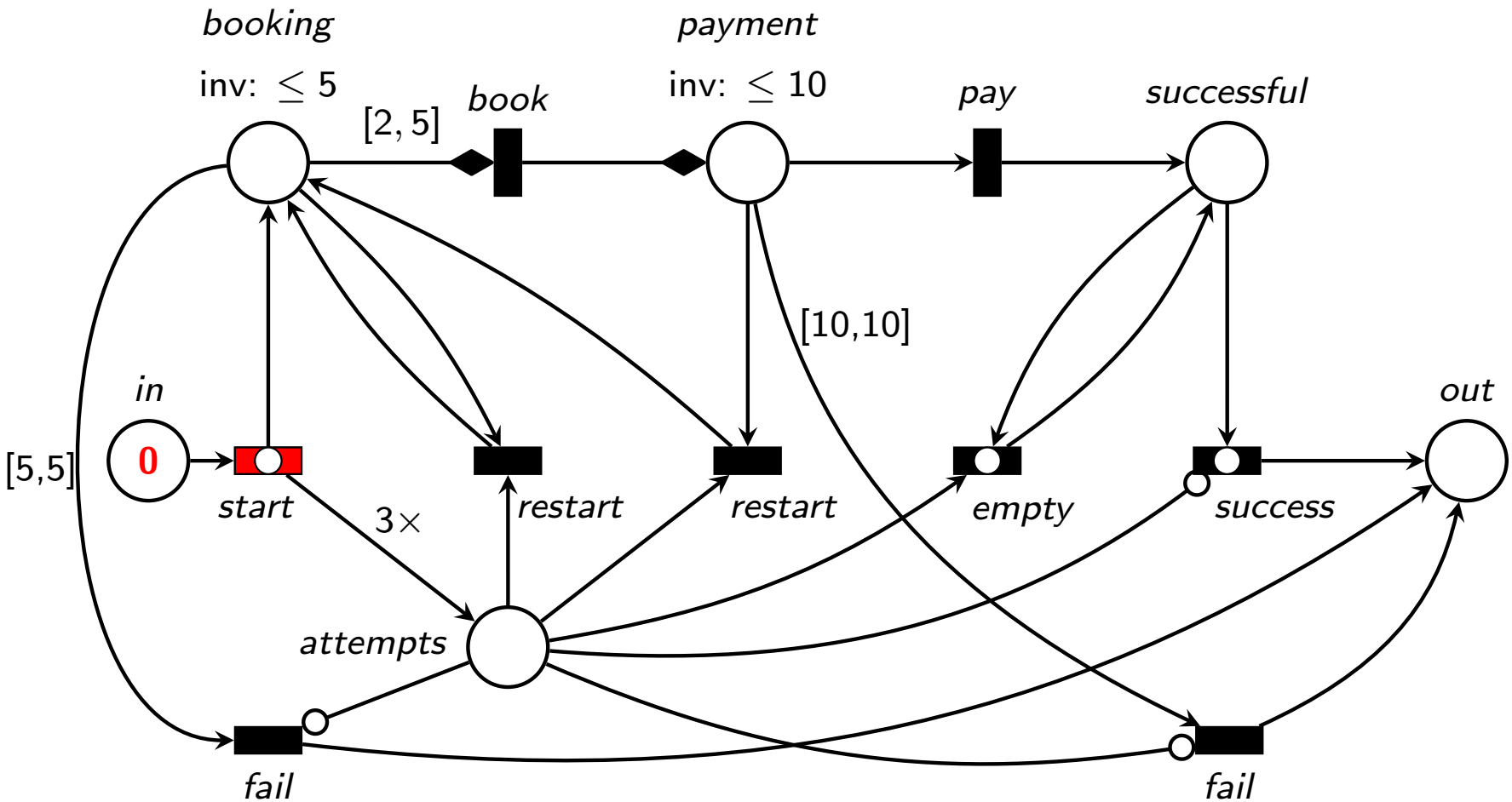
Timed-Arc Petri Net: Booking/Payment Example



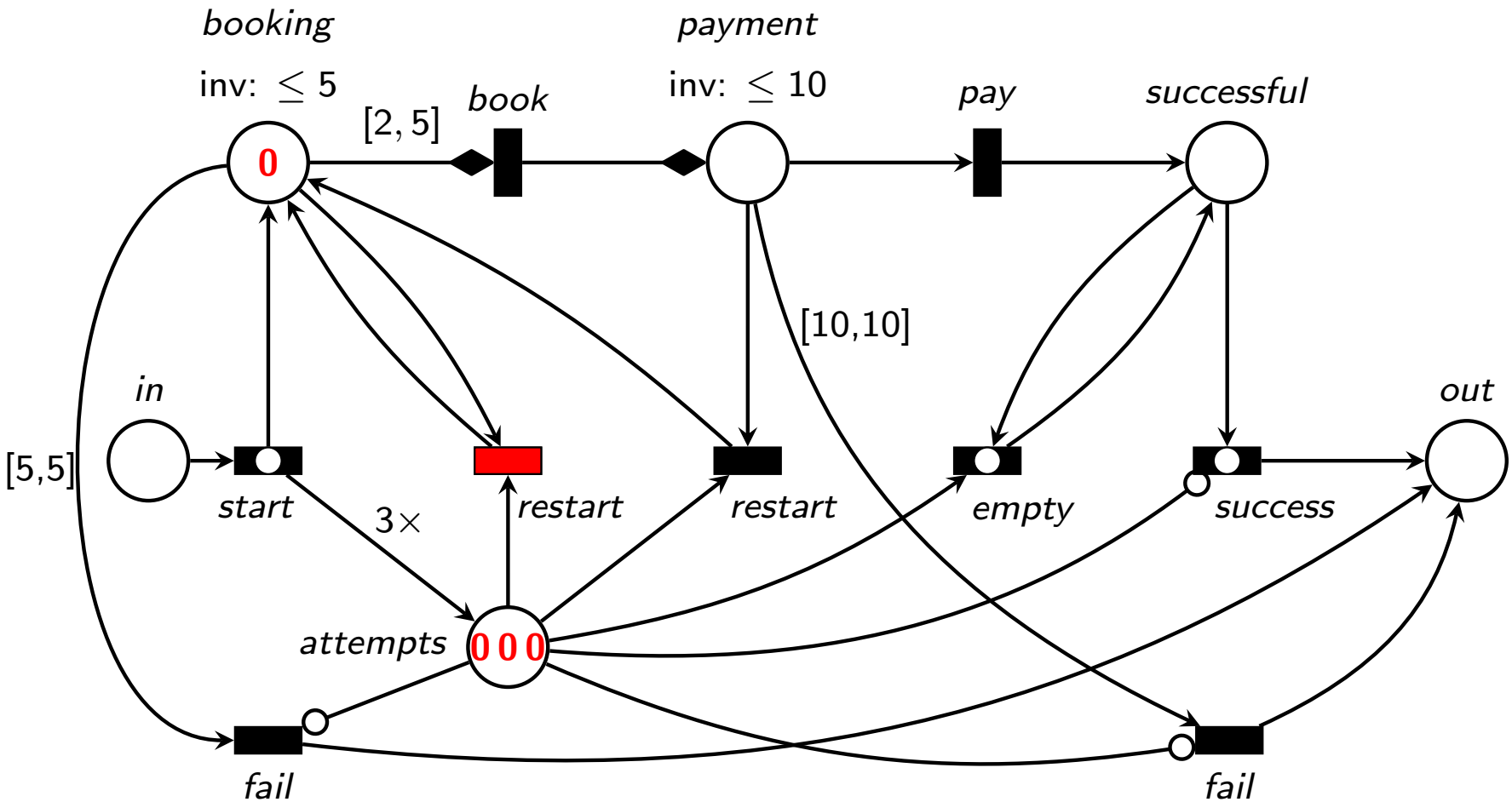
Timed-Arc Petri Net: Booking/Payment Example



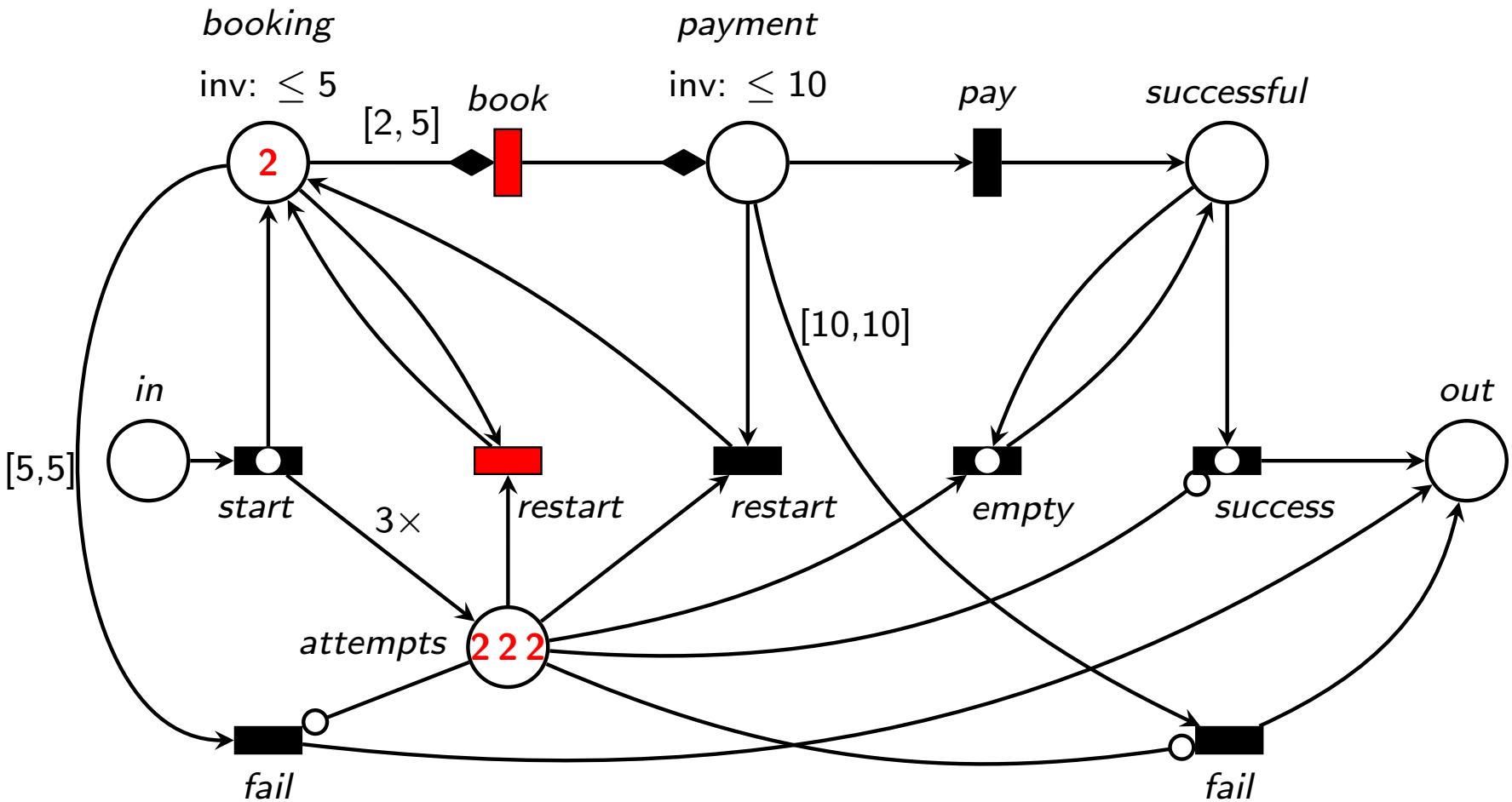
Timed-Arc Petri Net: Booking/Payment Example



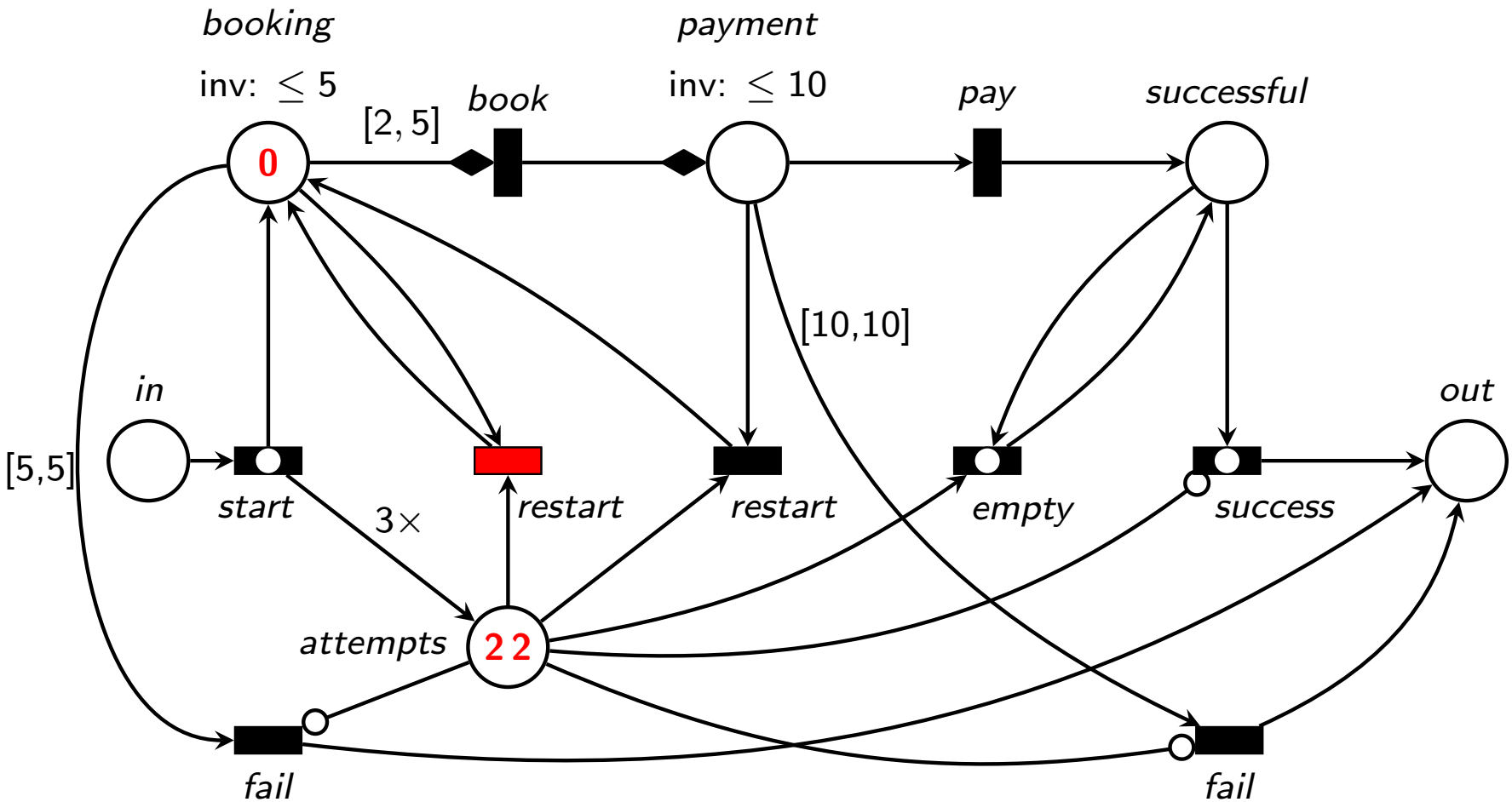
Timed-Arc Petri Net: Booking/Payment Example



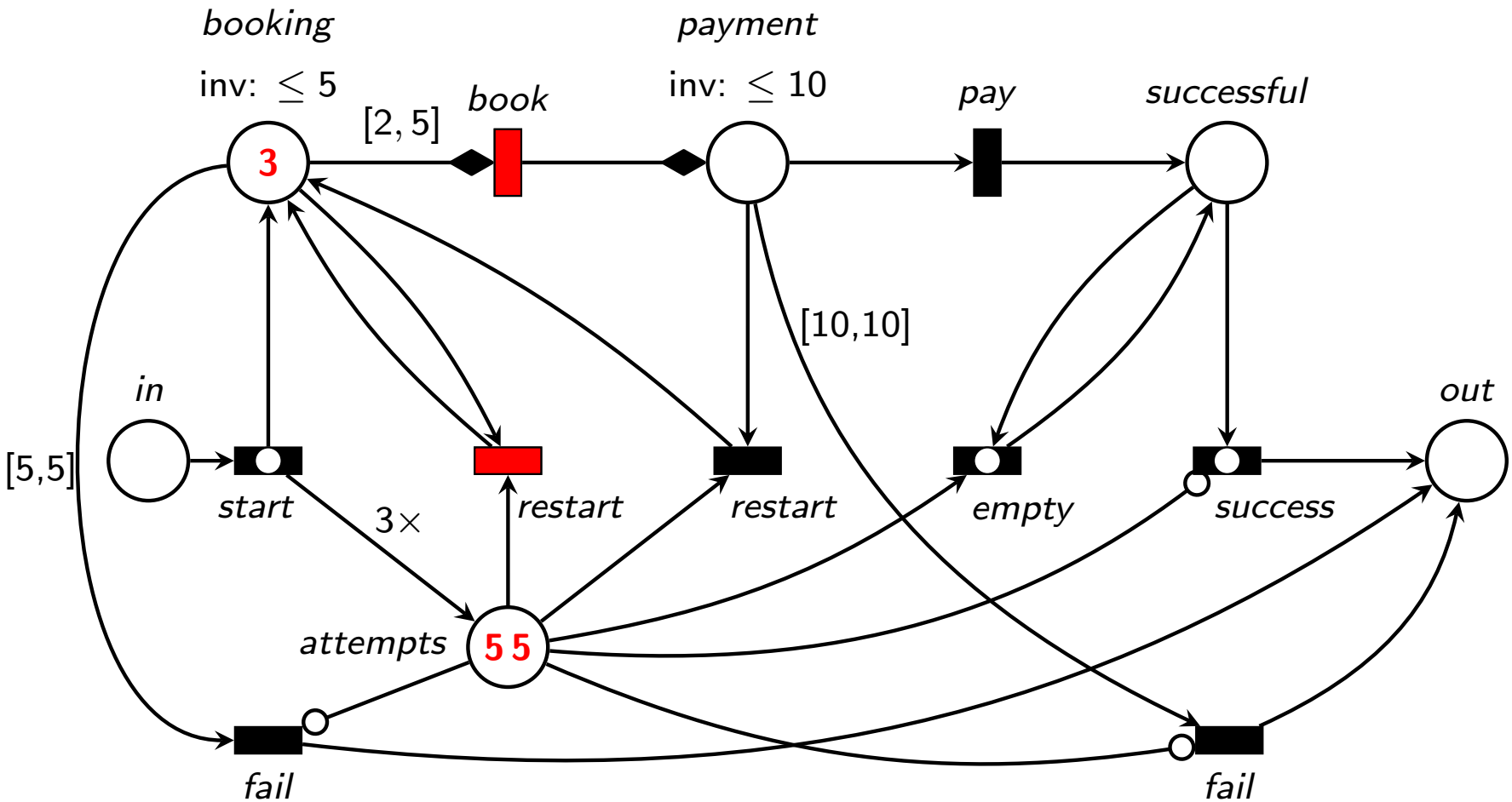
Timed-Arc Petri Net: Booking/Payment Example



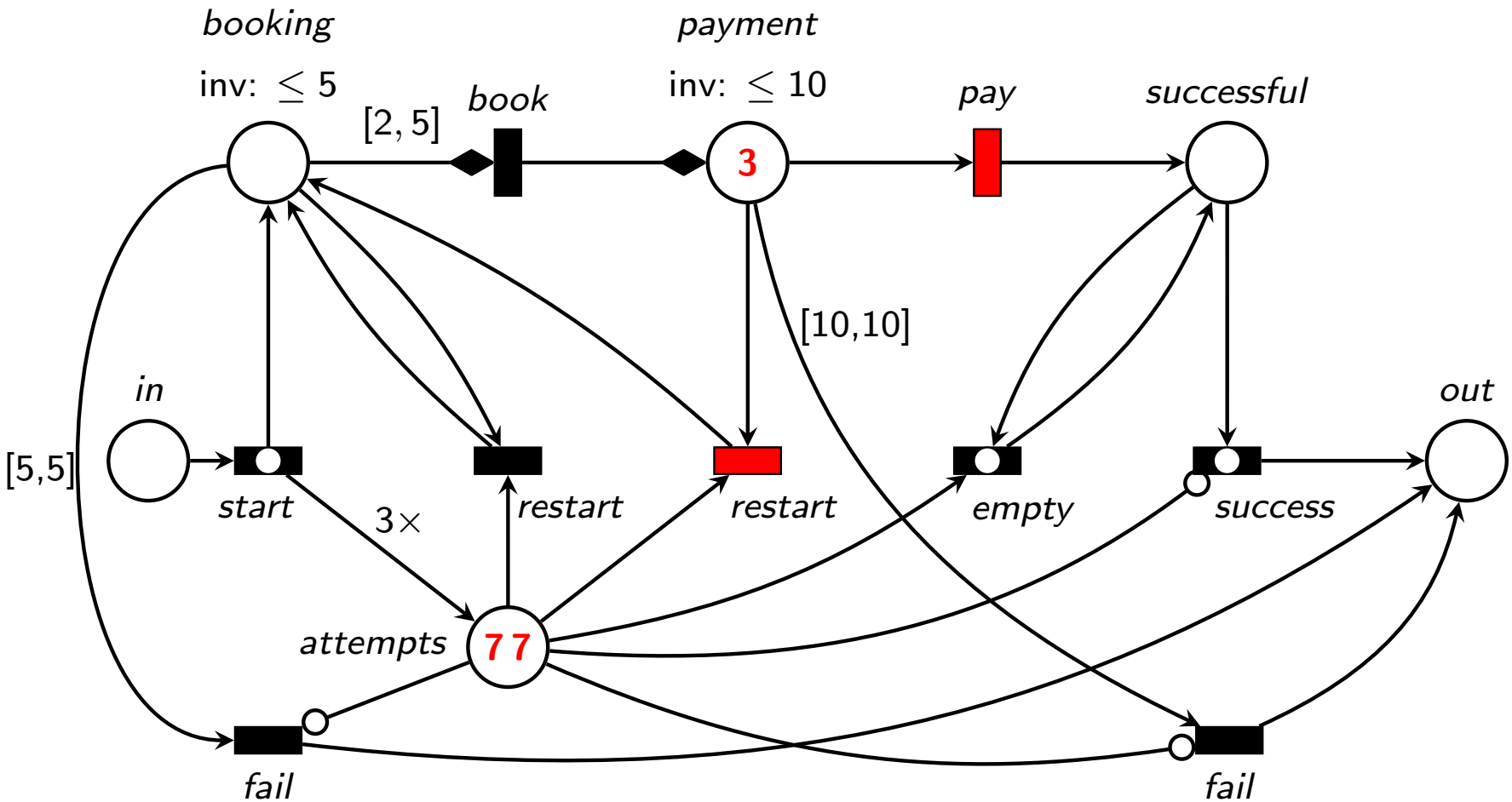
Timed-Arc Petri Net: Booking/Payment Example



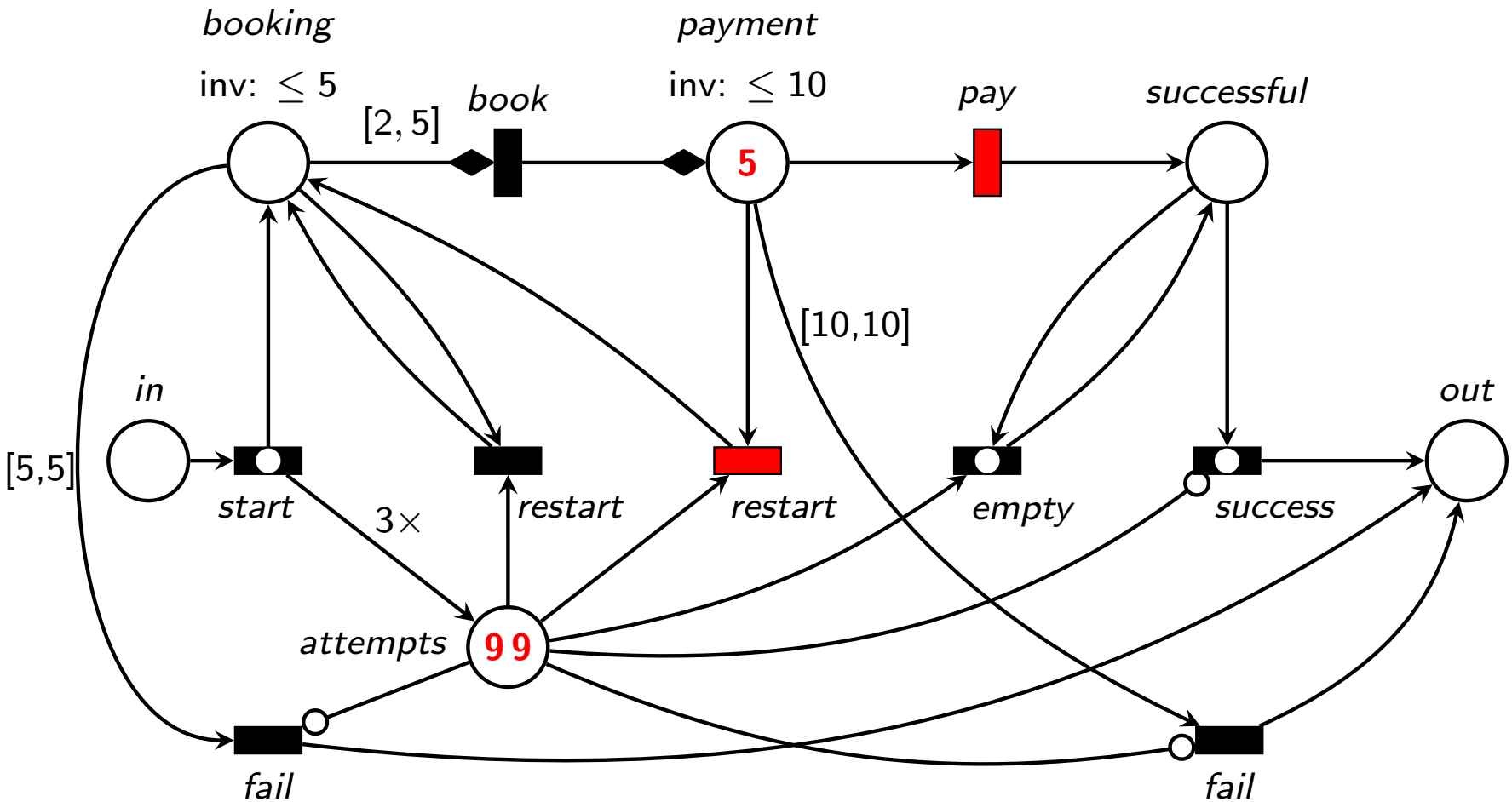
Timed-Arc Petri Net: Booking/Payment Example



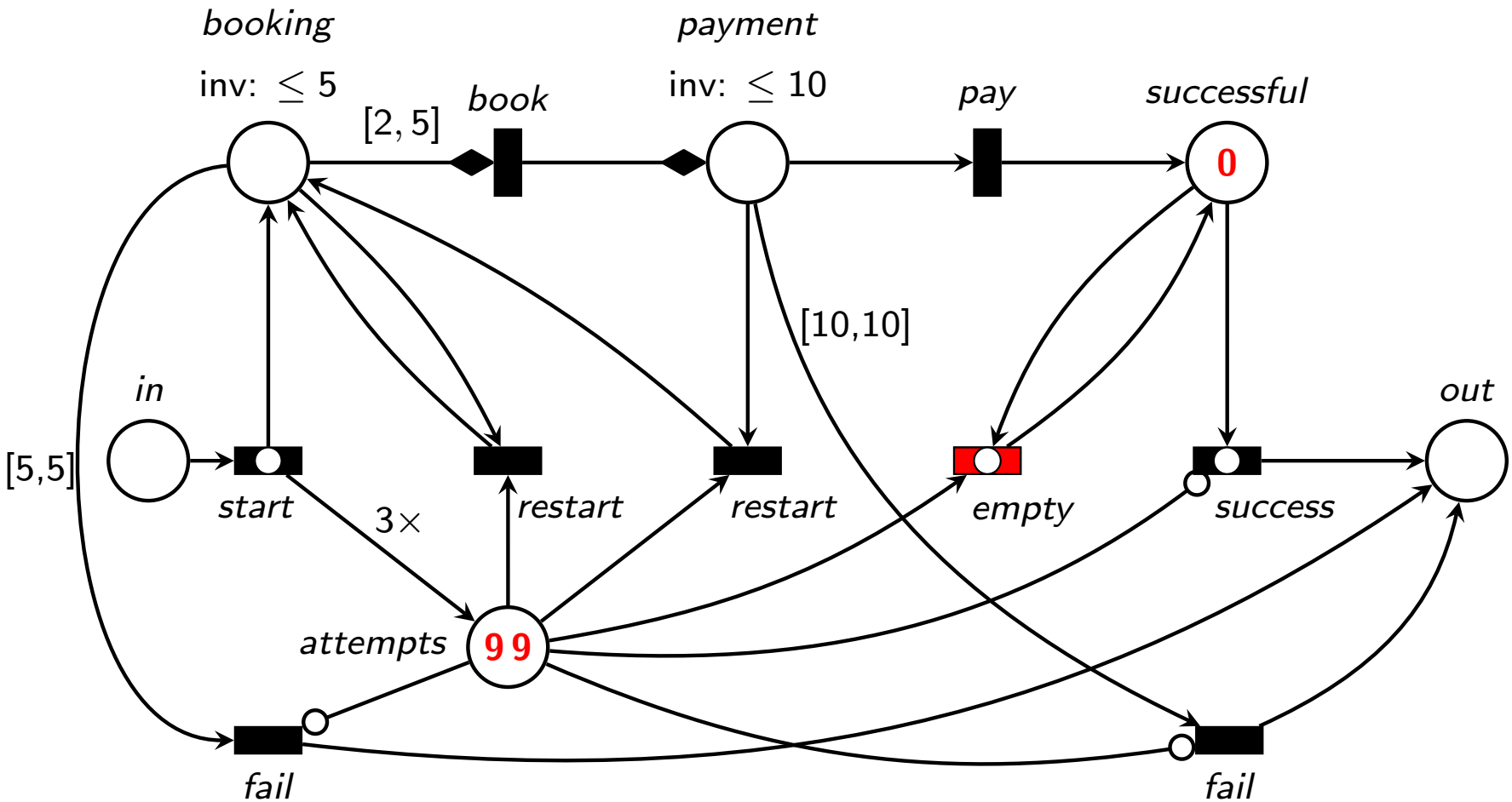
Timed-Arc Petri Net: Booking/Payment Example



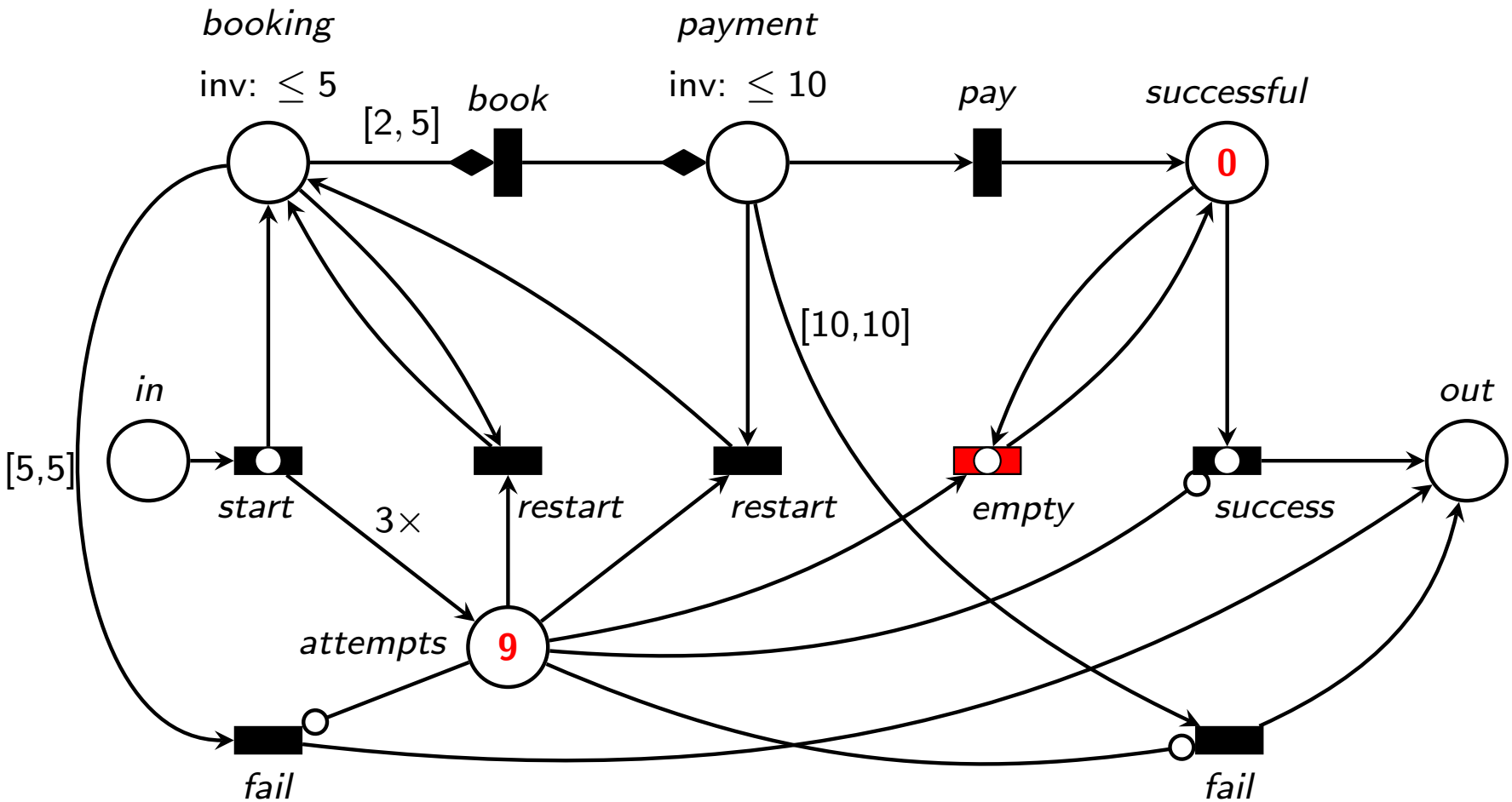
Timed-Arc Petri Net: Booking/Payment Example



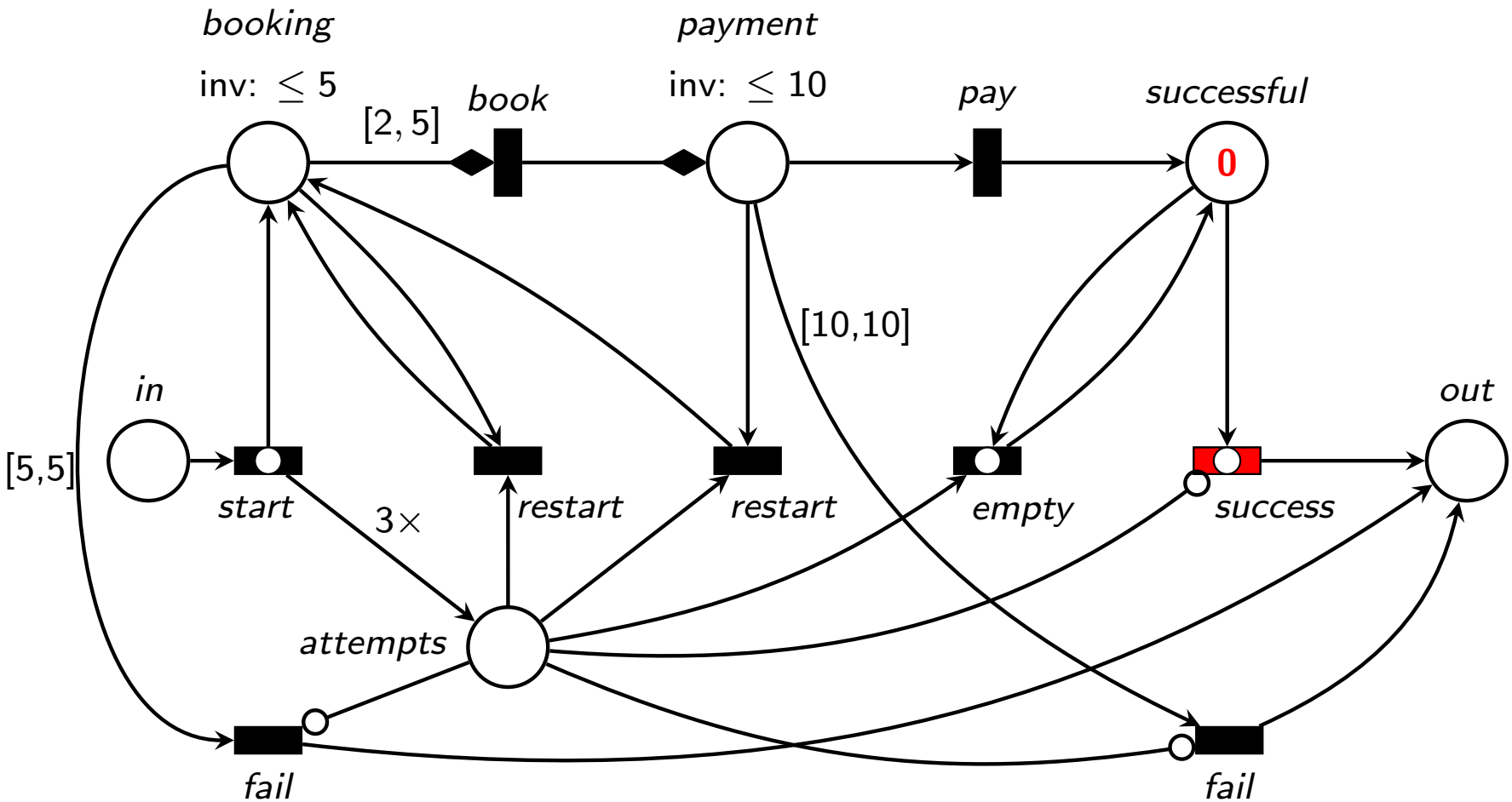
Timed-Arc Petri Net: Booking/Payment Example



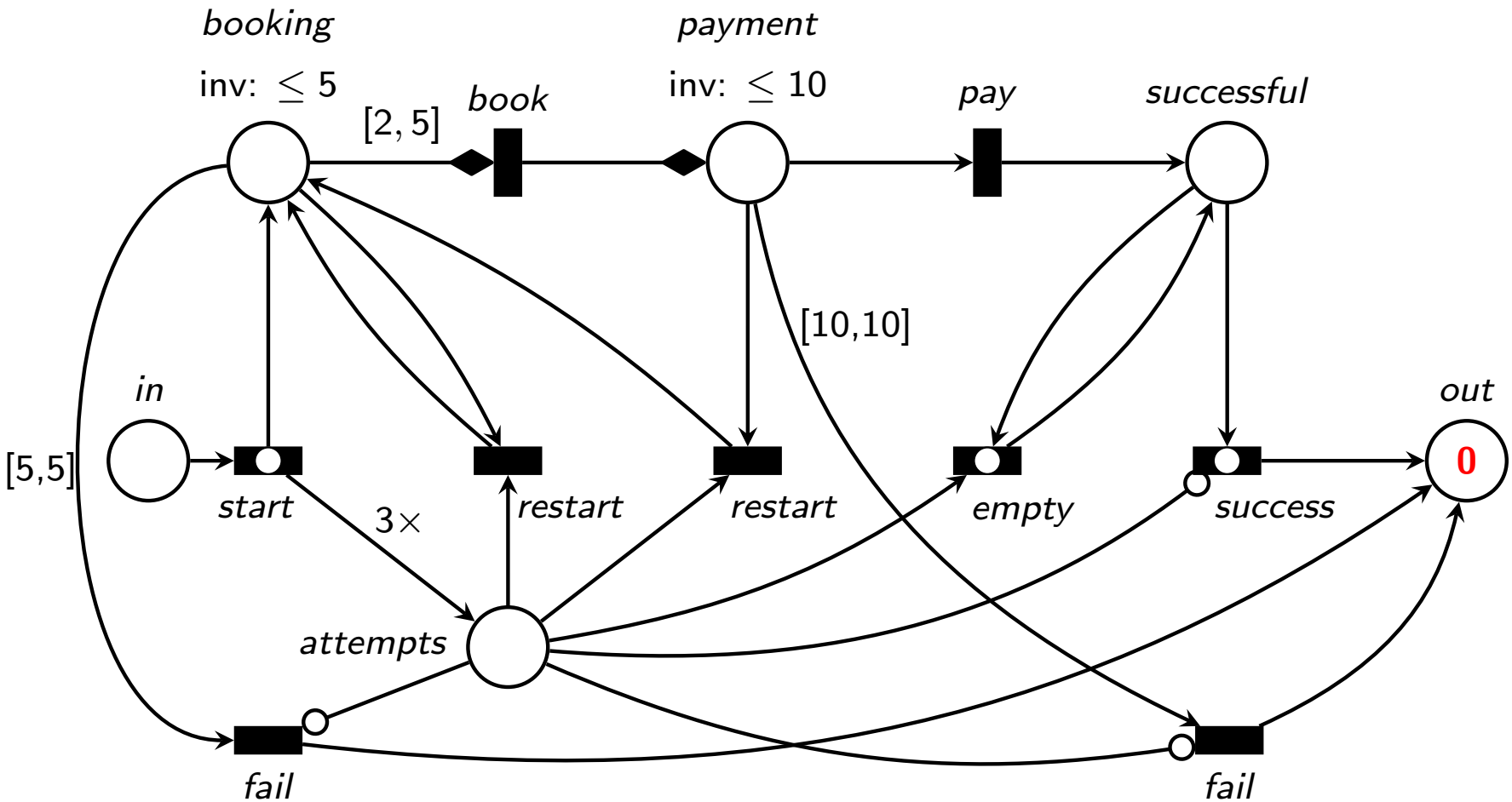
Timed-Arc Petri Net: Booking/Payment Example



Timed-Arc Petri Net: Booking/Payment Example



Timed-Arc Petri Net: Booking/Payment Example



Monotonic Timed-Arc Petri Nets

Timed-Arc Petri Nets (TAPN) Modelling Features:

- Timed tokens, intervals (guards) on arcs.
- Weighted arcs.
- Transport arcs.
- Inhibitor arcs.
- Age invariants.
- Urgent transitions.

Monotonic Timed-Arc Petri Nets

Timed-Arc Petri Nets (TAPN) Modelling Features:

- Timed tokens, intervals (guards) on arcs.
- Weighted arcs.
- Transport arcs.
- Inhibitor arcs.
- Age invariants.
- Urgent transitions.

Monotonic Timed-Arc Petri Nets (MTAPN)

No inhibitor arcs, no age invariants, no urgent transitions.

We consider the integer-delay (discrete-time) semantics (for now).

Marking Extrapolation

Marking in TAPN

$$M : P \to \mathcal{B}(\mathbb{N}_0)$$

Problem

Infinitely many markings even for bounded nets.

We define $cut(M)$ extrapolation for a marking M:

- compute for each place maximum relevant token ages

$$C_{max} : P \to (\mathbb{N}_0 \cup \{-1\})$$

- change the age of each token in place p exceeding the bound $C_{max}(p)$ into $C_{max}(p) + 1$.

Monotonicity Lemma for MTAPN

Monotonicity Lemma (t is transition, d is delay)

Let M and M' be markings in an MTAPN s.t. $cut(M) \sqsubseteq cut(M')$.

- If $M \xrightarrow{t} M_1$ then $M' \xrightarrow{t} M'_1$ and $cut(M_1) \sqsubseteq cut(M'_1)$.
- If $M \xrightarrow{d} M_1$ then $M' \xrightarrow{d} M'_1$ and $cut(M_1) \sqsubseteq cut(M'_1)$.

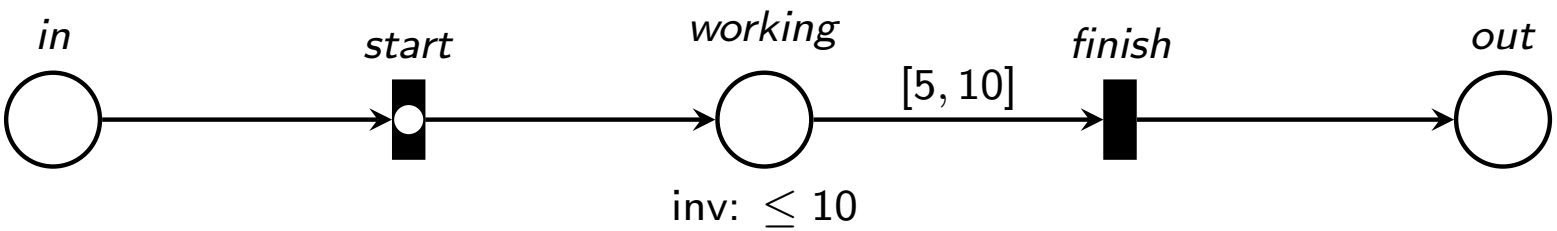
Fact: inhibitor arcs, age invariants and urgency break monotonicity.

Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.

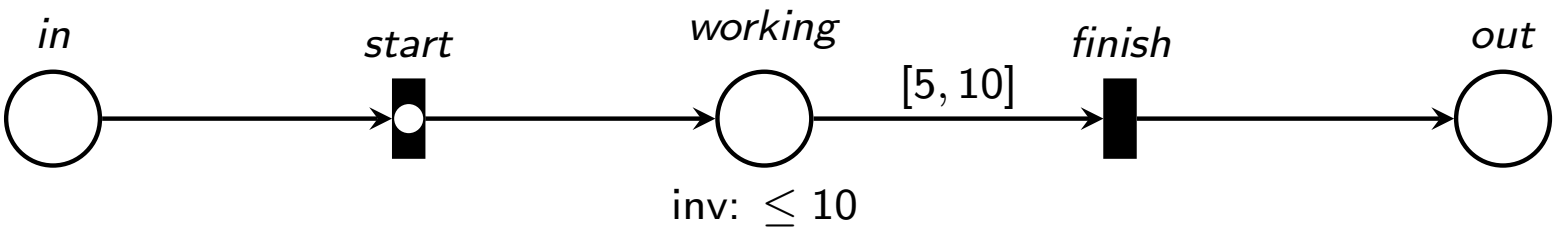


Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.



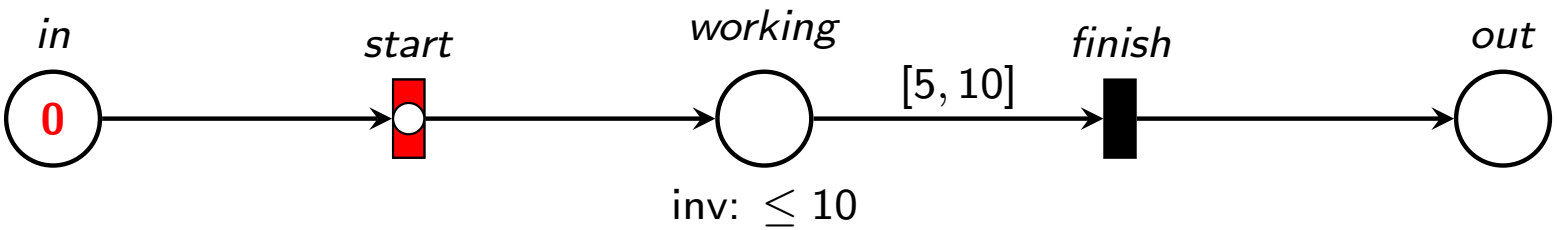
- An initial marking has just one token of age 0 in the place *in*.
- A final marking has exactly one token in place *out* and all other places are empty.

Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.



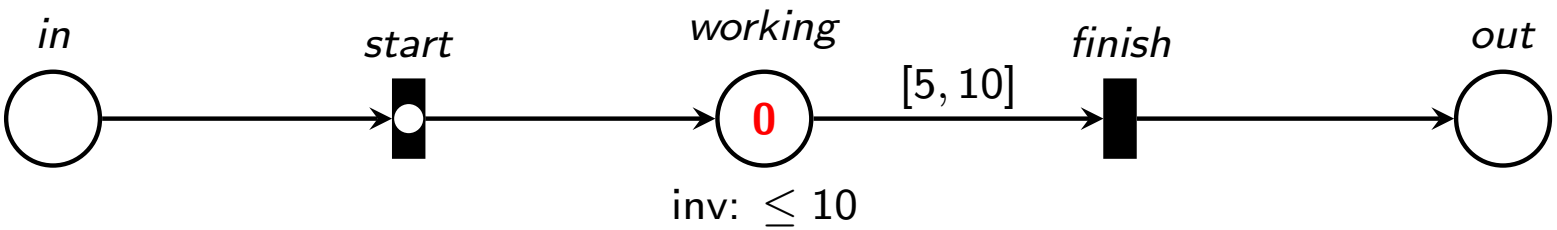
- An initial marking has just one token of age 0 in the place *in*.
- A final marking has exactly one token in place *out* and all other places are empty.

Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.



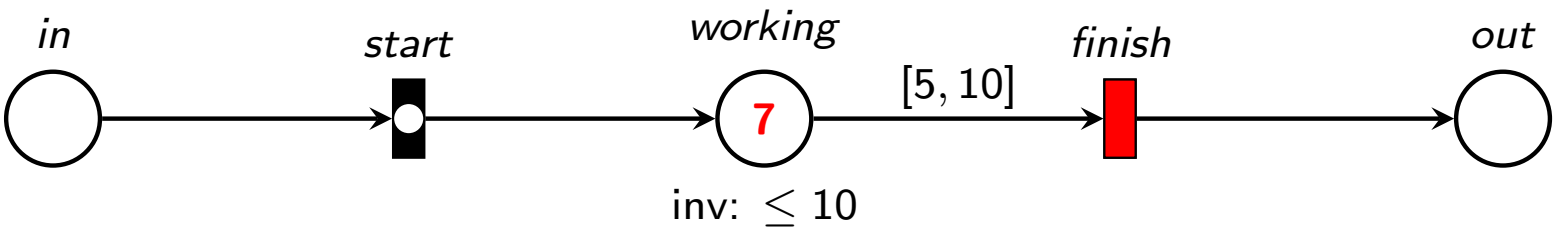
- An initial marking has just one token of age 0 in the place *in*.
- A final marking has exactly one token in place *out* and all other places are empty.

Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.



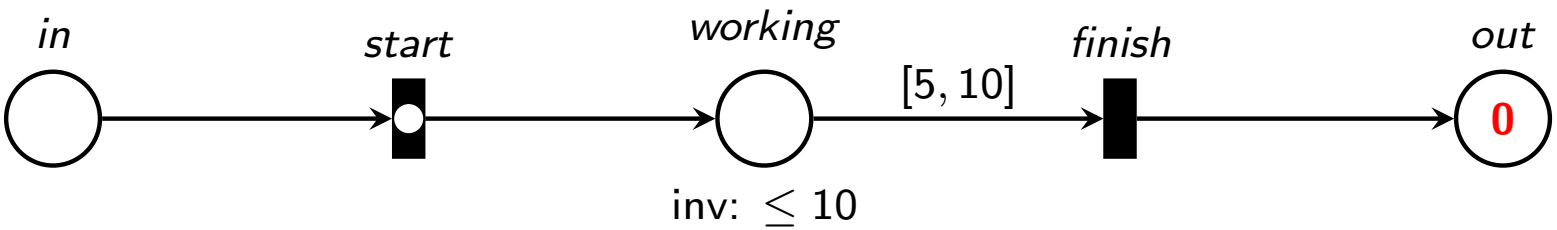
- An initial marking has just one token of age 0 in the place *in*.
- A final marking has exactly one token in place *out* and all other places are empty.

Timed-Arc Workflow Net

Definition

A TAPN is called a timed-arc workflow net if

- it has a unique place $in \in P$ s.t. ${}^\bullet in = \emptyset$ and $in^\bullet \neq \emptyset$,
- it has a unique place $out \in P$ s.t. $out^\bullet = \emptyset$ and ${}^\bullet out \neq \emptyset$,
- ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$ for all $p \in P \setminus \{in, out\}$, and
- ${}^\bullet t \neq \emptyset$ for all $t \in T$.



- An initial marking has just one token of age 0 in the place *in*.
- A final marking has exactly one token in place *out* and all other places are empty.

Soundness of Timed-Arc Workflow Nets

Definition

A timed-arc workflow net is sound if for any marking M reachable from the initial marking holds:

1. from M it is possible to reach some final marking, and
2. if $M(out)$ contains a token then M is a final marking.

Soundness of Timed-Arc Workflow Nets

Definition

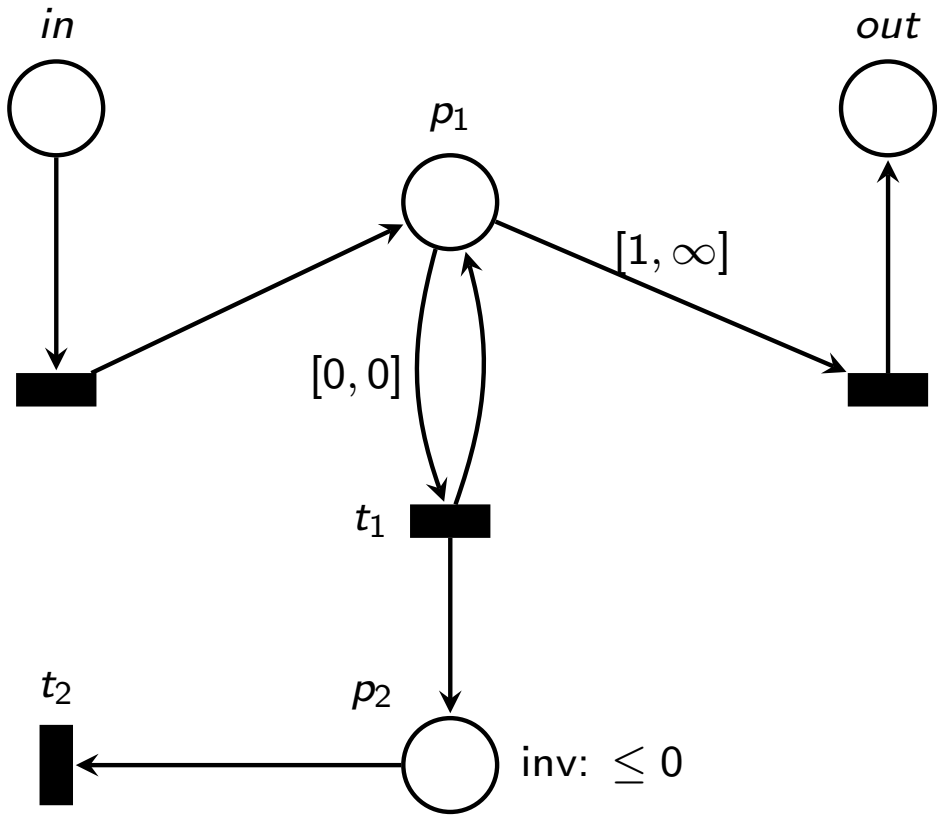
A timed-arc workflow net is sound if for any marking M reachable from the initial marking holds:

1. from M it is possible to reach some final marking, and
2. if $M(out)$ contains a token then M is a final marking.

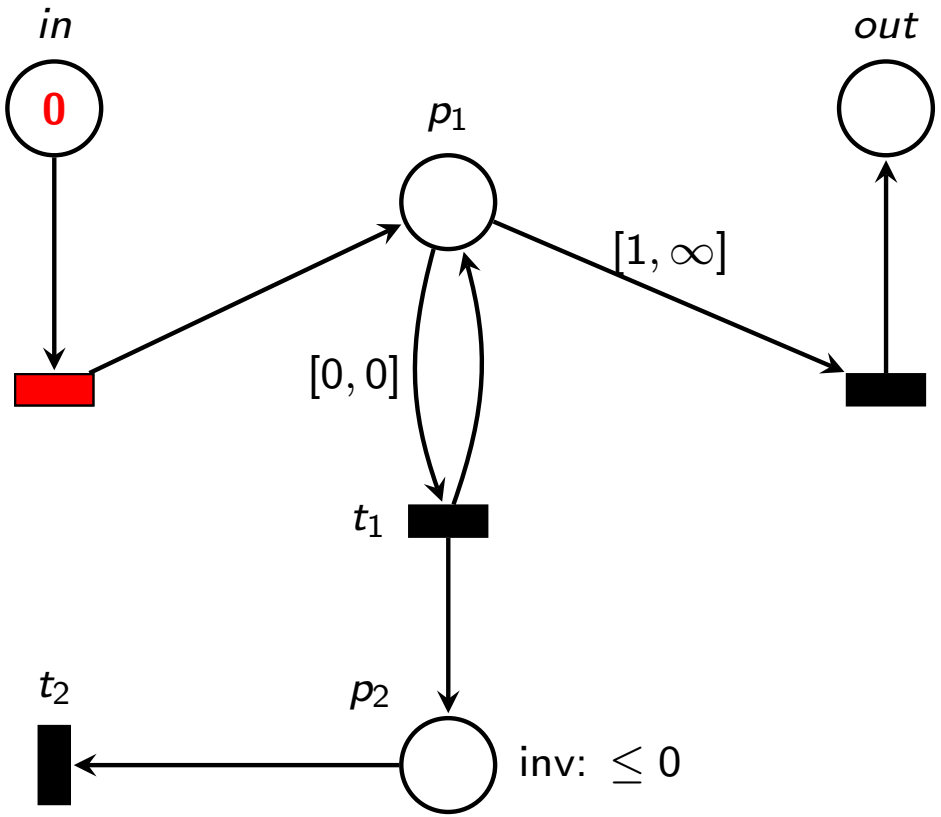
Soundness Implies Boundedness

If N is a sound and monotonic timed-arc workflow net then N is bounded.

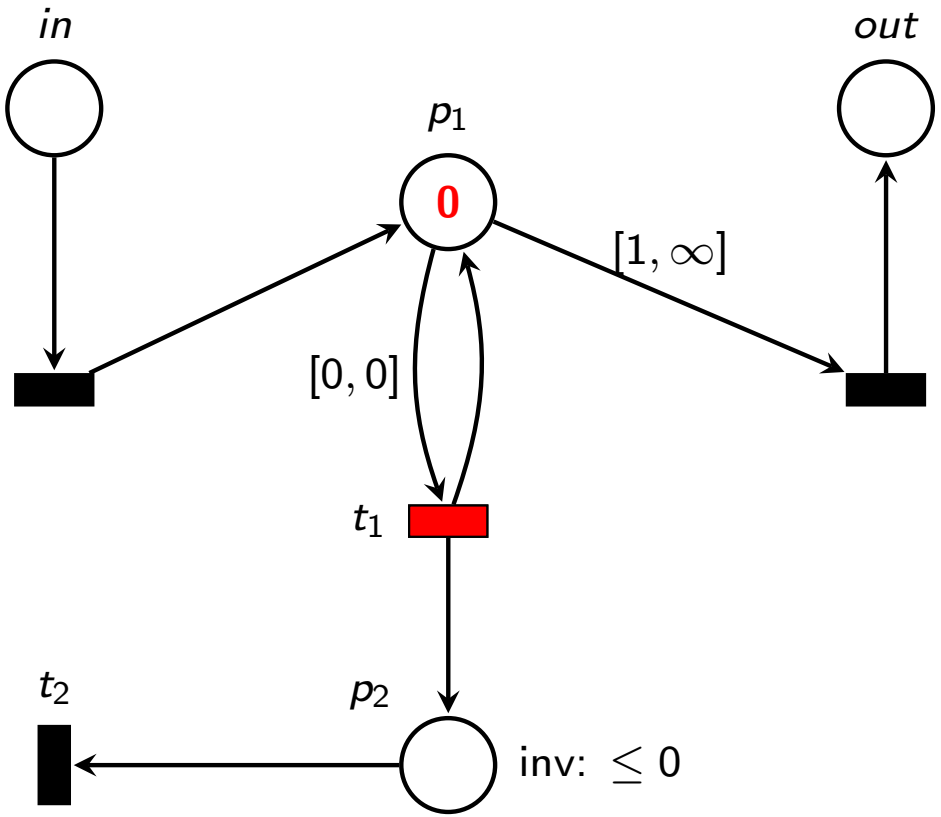
Sound and Unbounded Net with Age Invariants



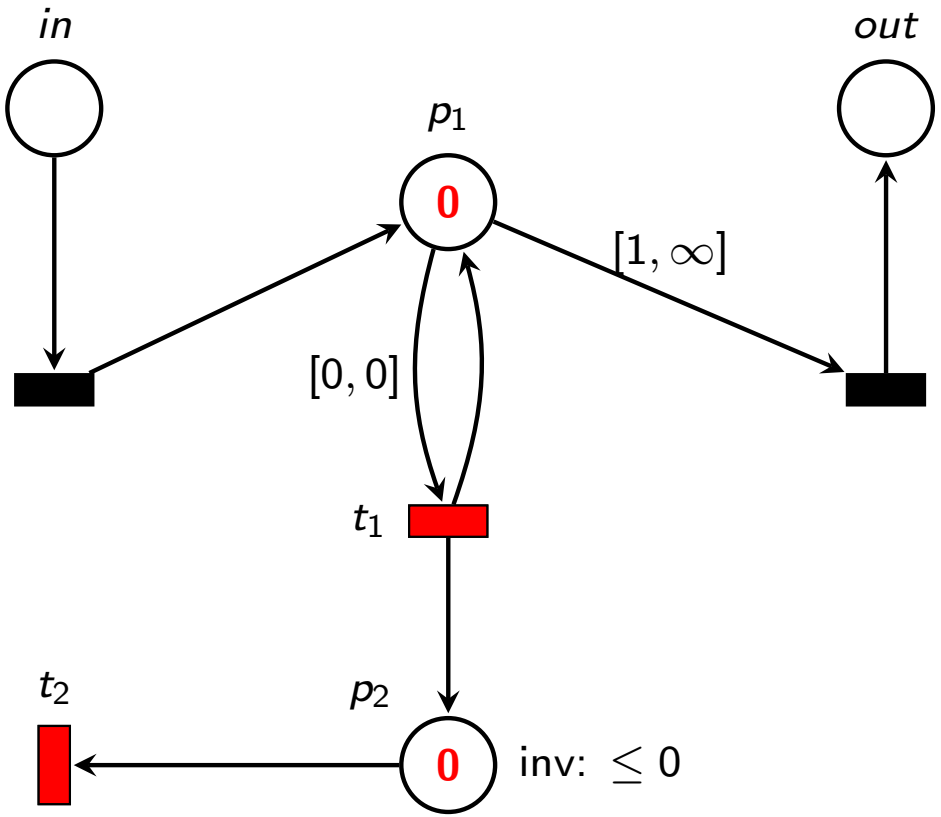
Sound and Unbounded Net with Age Invariants



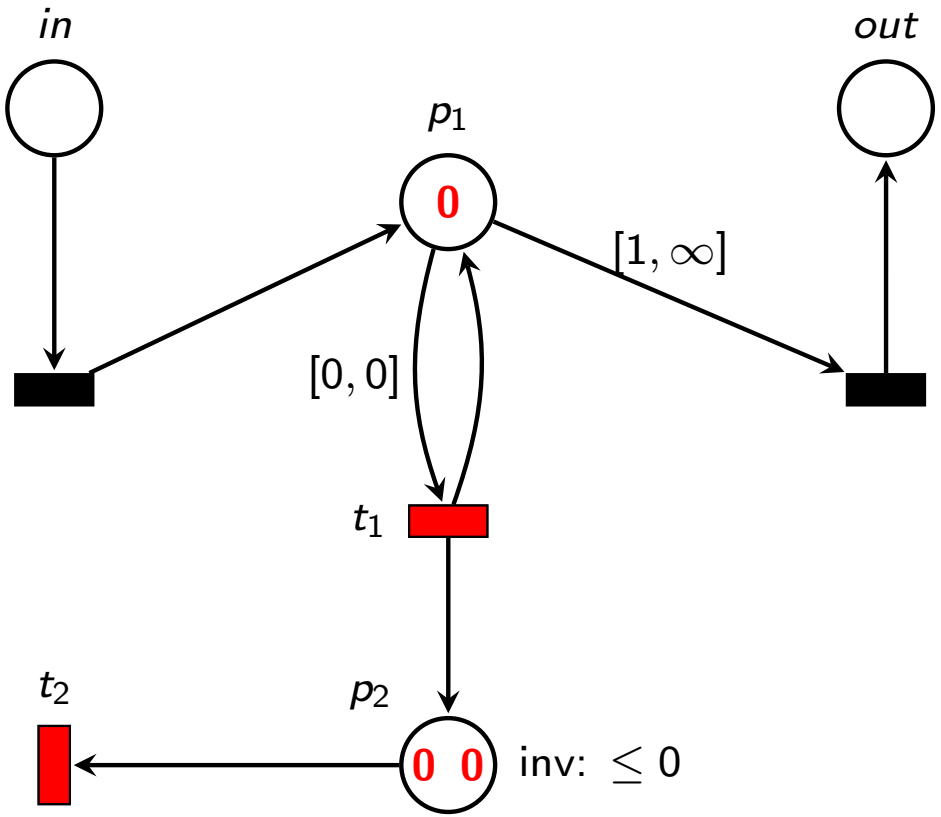
Sound and Unbounded Net with Age Invariants



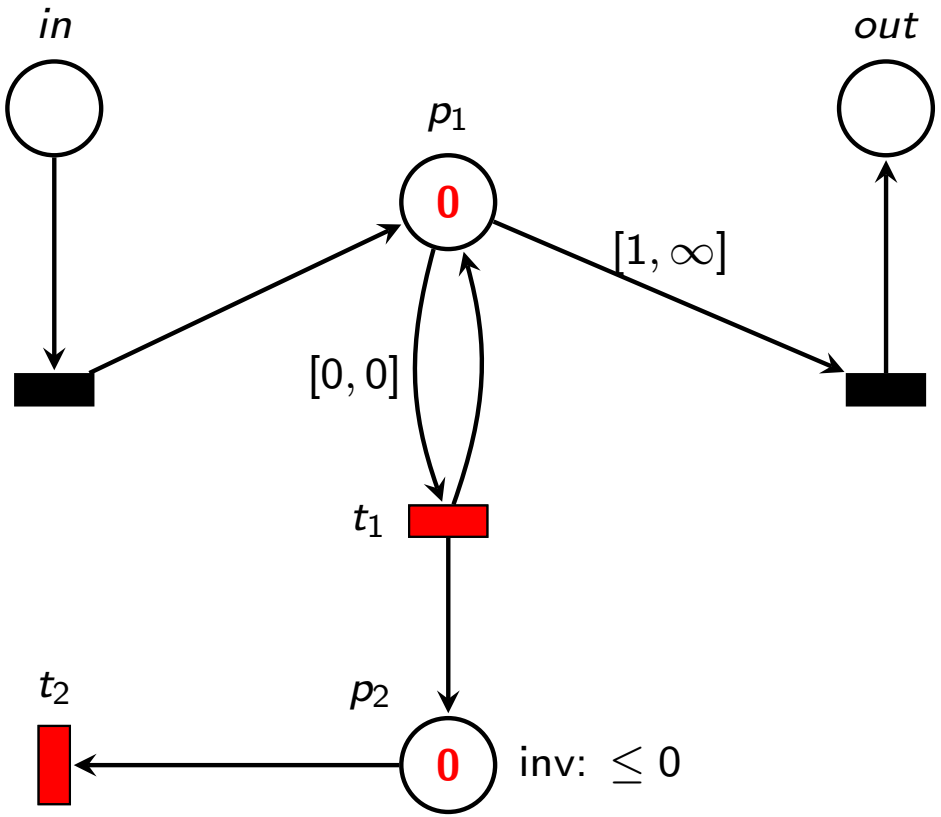
Sound and Unbounded Net with Age Invariants



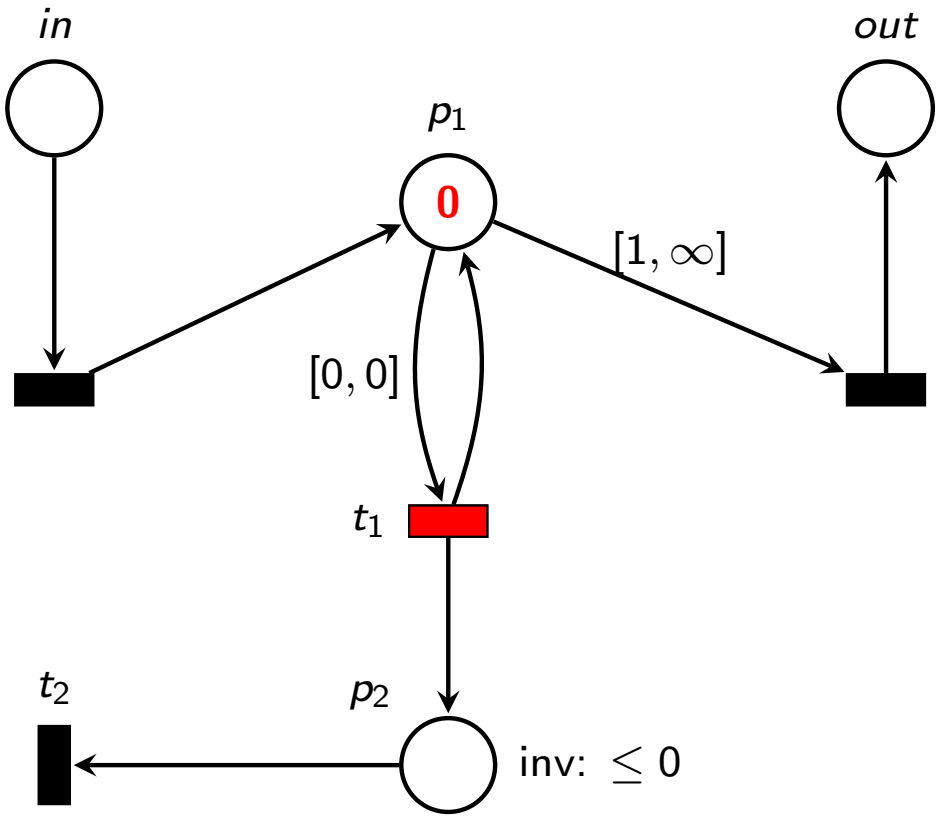
Sound and Unbounded Net with Age Invariants



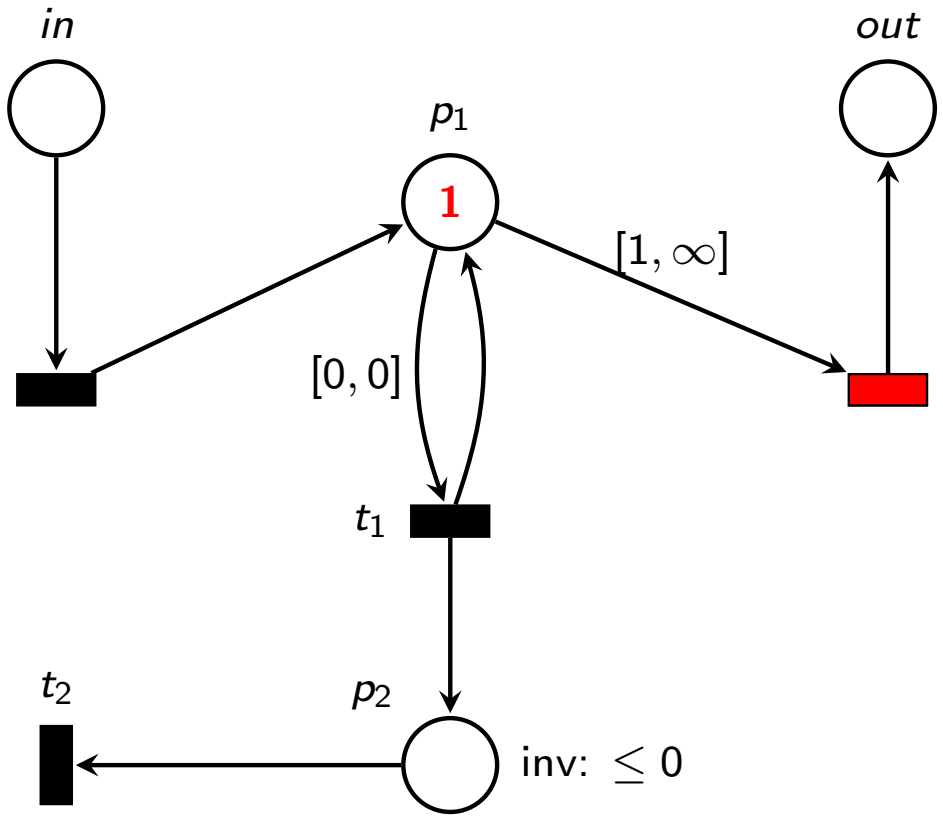
Sound and Unbounded Net with Age Invariants



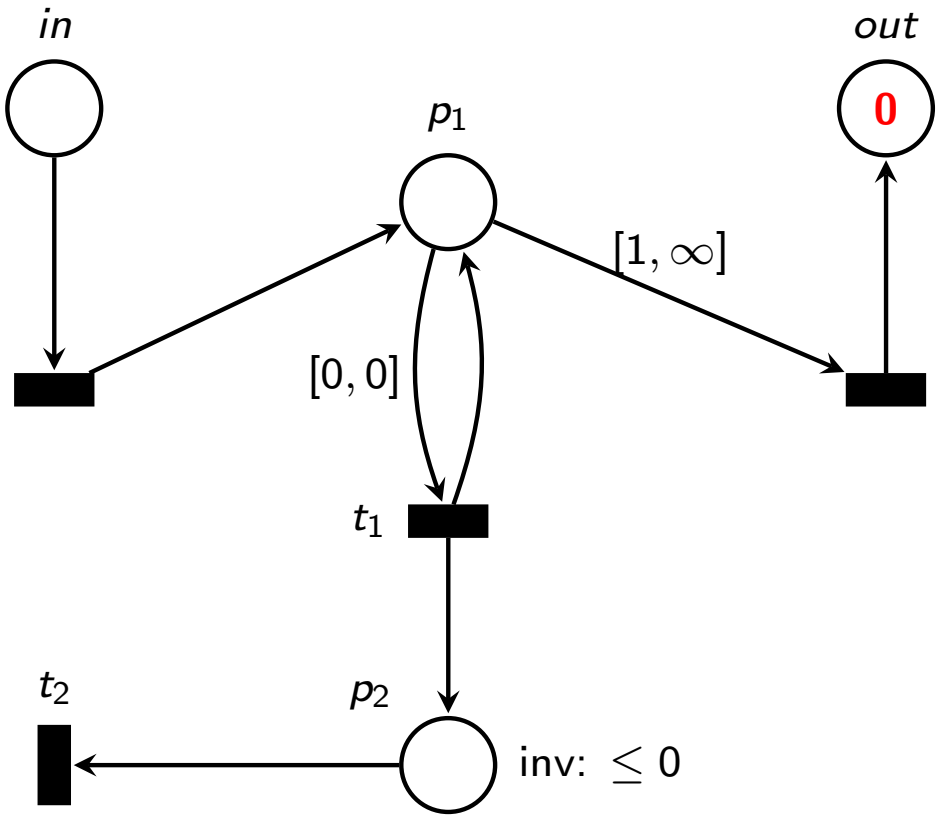
Sound and Unbounded Net with Age Invariants



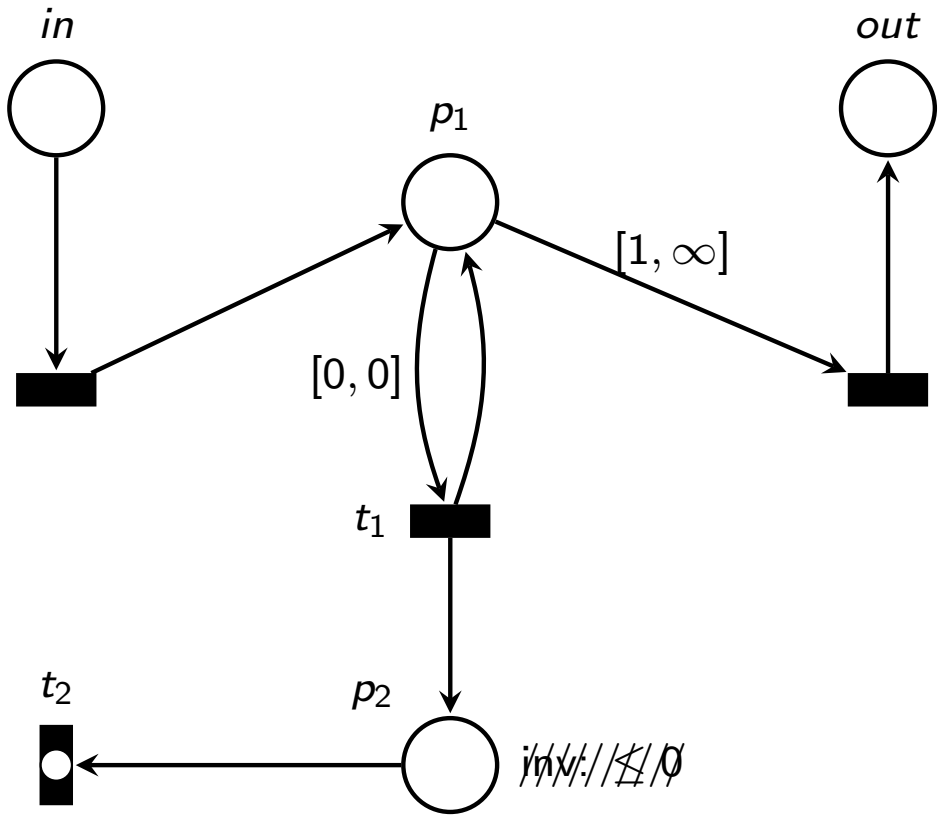
Sound and Unbounded Net with Age Invariants



Sound and Unbounded Net with Age Invariants



Sound and Unbounded Net with Age Invariants



Sound and Unbounded Net with Urgent Transitions

Remove age invariant ≤ 0 at place p_2 and make t_2 urgent.

Decidability of Soundness

Theorem

Soundness is undecidable for timed-arc workflow nets.

Undecidable even for monotonic nets with only inhibitor arcs, or only age invariants, or only urgent transitions.

Decidability of Soundness

Theorem

Soundness is undecidable for timed-arc workflow nets.

Undecidable even for monotonic nets with only inhibitor arcs, or only age invariants, or only urgent transitions.

Theorem

Soundness is decidable for

- bounded timed-arc workflow nets, and for
- monotonic timed-arc workflow nets.

Proof: Forward and backward search through the extrapolated state-space (using the function *cut*). Termination for MTAPN due to the monotonicity lemma.

Compare Decidability of Soundness with Reachability

Notice that for the subclass of monotonic timed-arc Petri nets

- reachability is undecidable [Ruiz, Gomez, Escrig'99], but
- soundness is decidable.

Compare Decidability of Soundness with Reachability

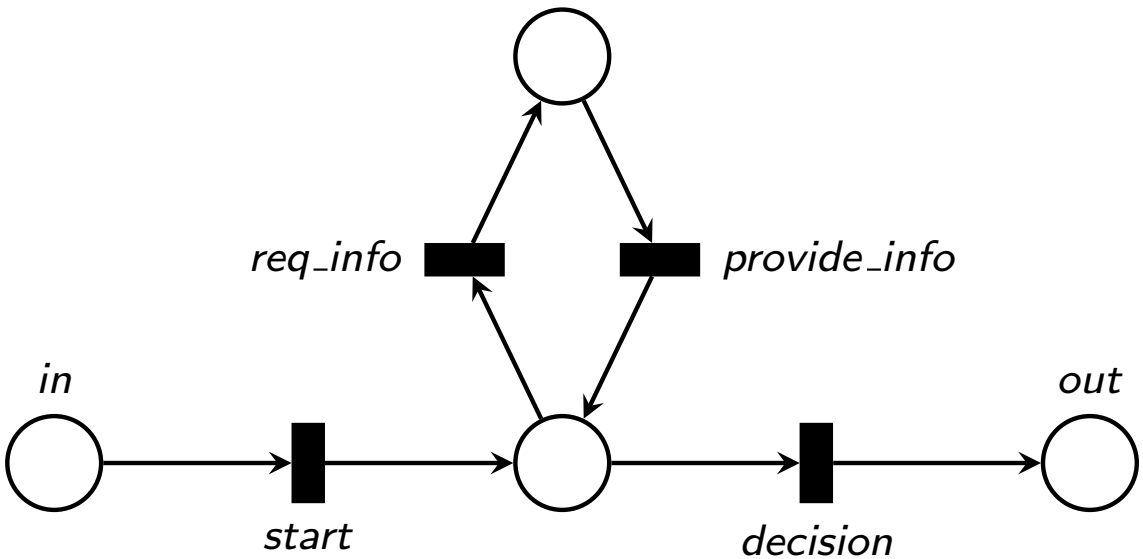
Notice that for the subclass of monotonic timed-arc Petri nets

- reachability is undecidable [Ruiz, Gomez, Escrig'99], but
- soundness is decidable.

Question

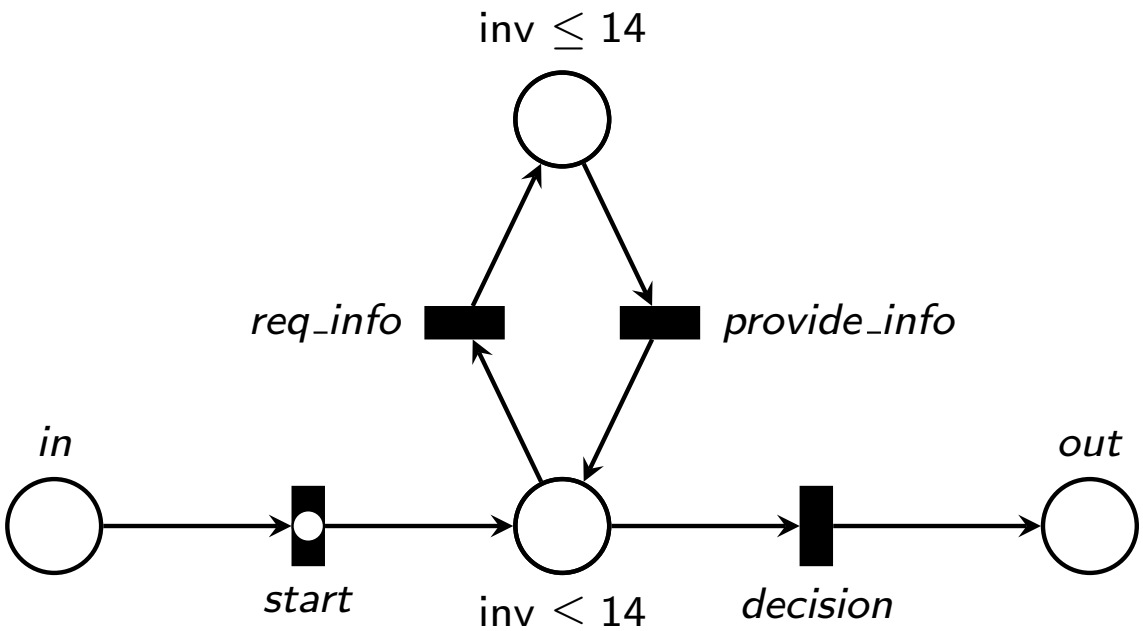
Is soundness always sufficient for timed workflows?

Customer Complaint Workflow



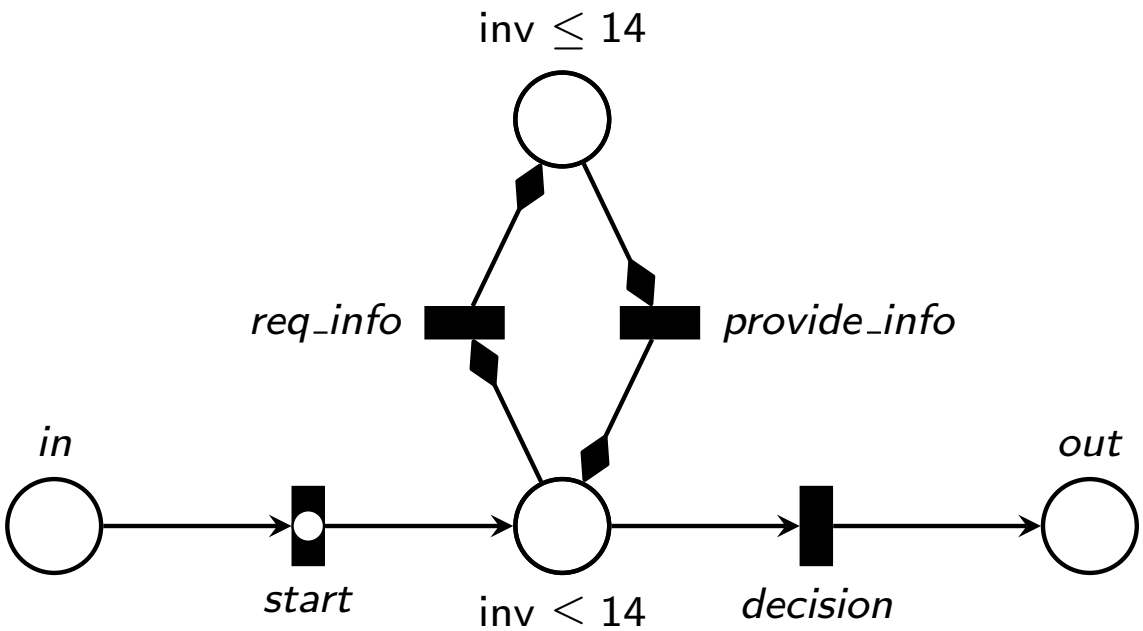
Sound workflow, no timing information, no progress.

Customer Complaint Workflow



Progress is ensured, infinite time-divergent behaviour.

Customer Complaint Workflow



Strongly sound workflow with time-bounded execution.

Strong Soundness

Definition

A timed-arc workflow net is strongly sound if

- it is sound,
- has no time-divergent markings (except for the final ones), and
- every infinite computation is time-bounded.

We can define maximum execution time for strongly sound nets.

Strong Soundness

Definition

A timed-arc workflow net is strongly sound if

- it is sound,
- has no time-divergent markings (except for the final ones), and
- every infinite computation is time-bounded.

We can define maximum execution time for strongly sound nets.

Theorem

Strong soundness of timed-arc workflow nets is undecidable.

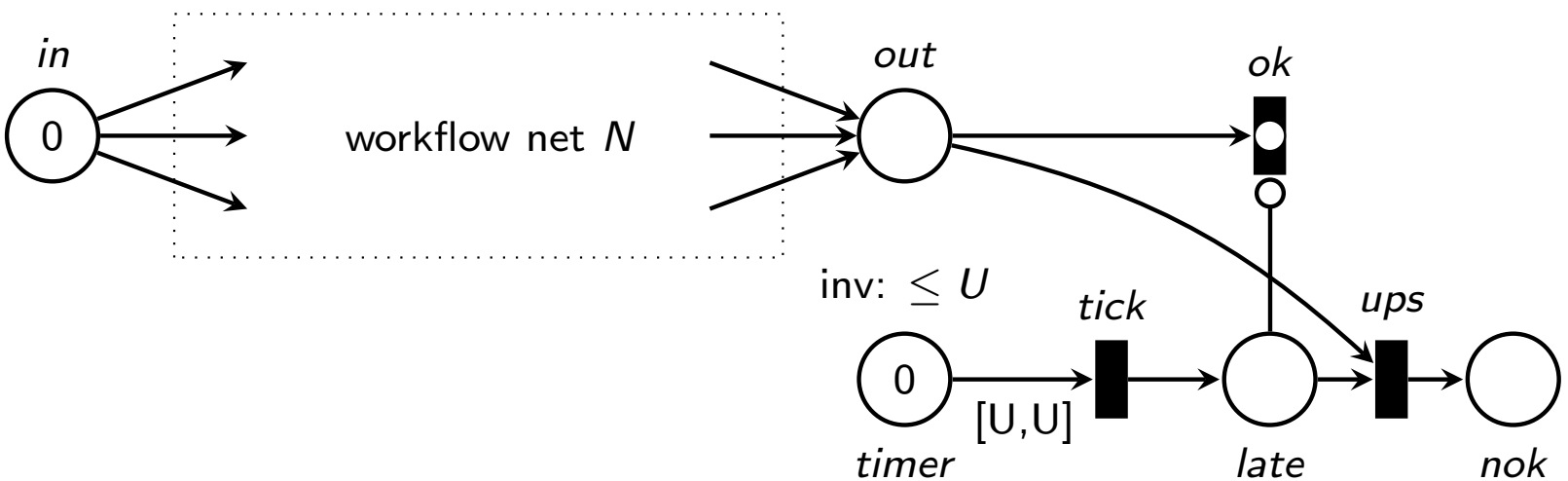
Theorem

Strong soundness of bounded timed-arc workflow nets is decidable.

Proof: By reduction to reachability on timed-arc Petri nets.

Decidability of Strong Soundness (Proof Sketch)

- Perform normal soundness check and remember the size S of its state-space (in the extrapolated semantics).
- Let B be the maximum possible delay in any marking.
- Check if the given workflow net can delay more than $U = S \cdot B + 1$ time units before reaching a final marking.
  - If yes, it is not strongly sound.
  - If no, it is strongly sound.



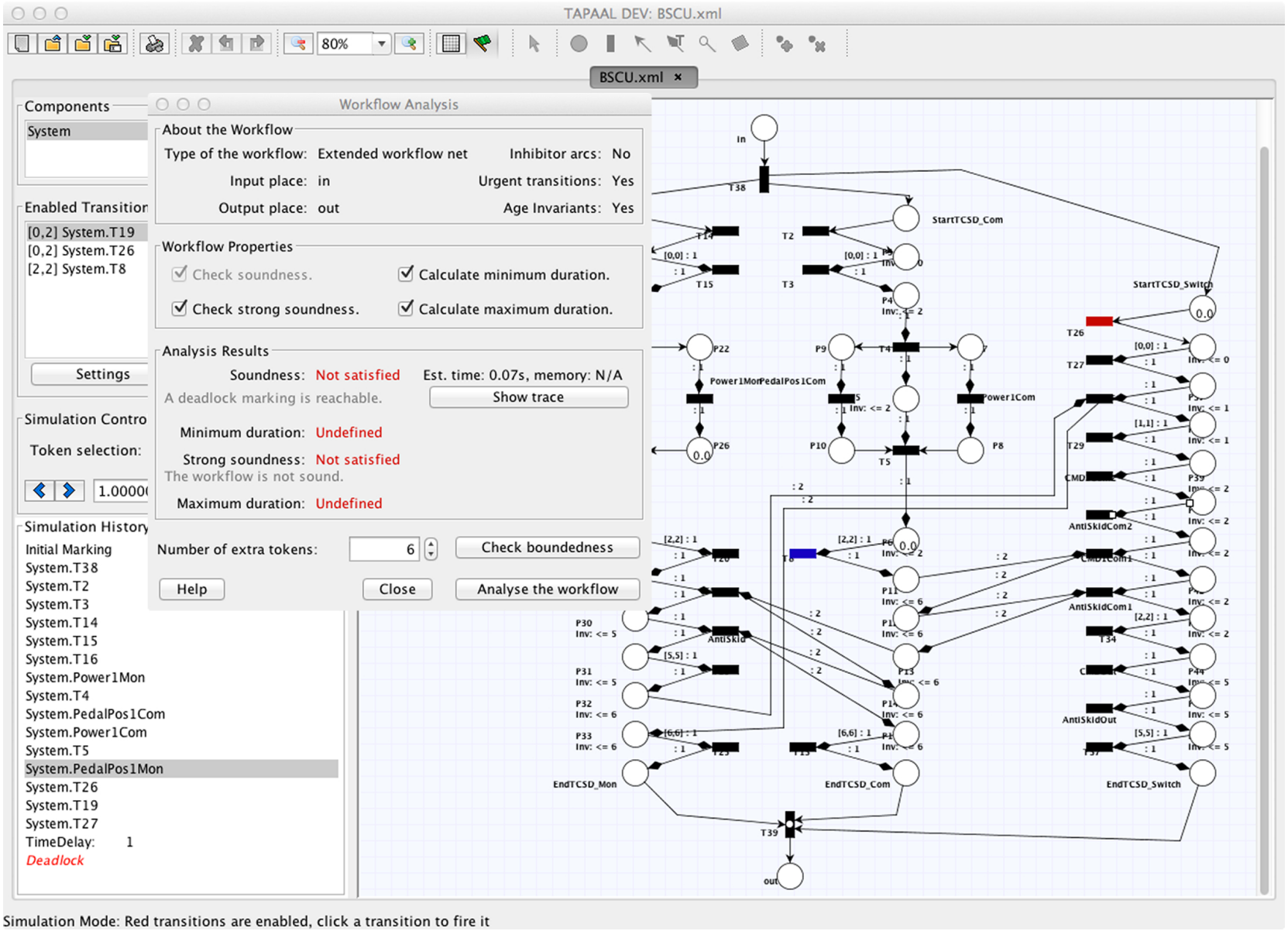
Implementation and Experiments

- All algorithms implemented within TAPAAL (www.tapaal.net).
- Publicly available and open-source.
- Graphical editor with components, visual simulator.
- Efficient engine implementation (including further optimizations).

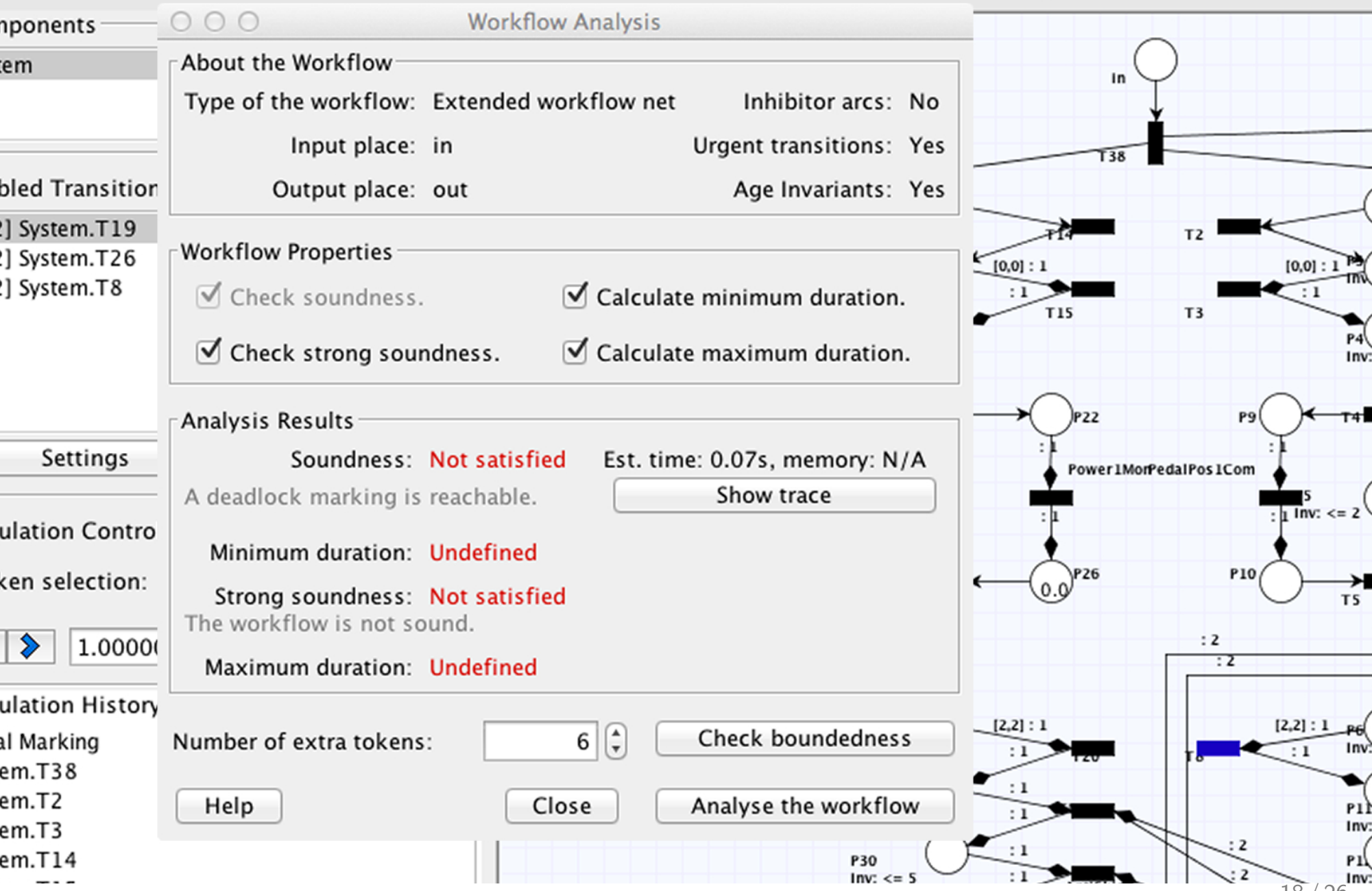
Case studies:

- Break System Control Unit, a part of the SAE standard ARP4761 (certification of civil aircrafts).
- MPEG-2 encoding algorithm on multi-core processors.
- Blood transfusion workflow, a larger benchmarking case-study described in little-JIL workflow language.
- Home automation system for light control in a family house with 16 lights/25 buttons, motion sensors and alarm.

TAPAAL Verification of Break System Control Unit



TAPAAL Verification of Break System Control Unit



Recent TAPAAL Development

- TAPAAL is being continuously improved and extended (MPEG-2 workflow analysis with two B-frames took 10s last year, now it takes only 1.4s).
- Memory preserving data structure PTrie.

MPEG-2 with three B-frames

	soundness	strong soundness
no PTrie	33s / 1071MB	30s / 970MB
PTrie	42s / 276MB	45s / 191MB

- Approximate analysis (smaller constants, less precision).
- Compositional, resource-aware analysis.

Future TAPAAL Development

- Resources with quantitative aspects (cost, energy).
- Two player timed workflow games (also with stochastic opponent).
- Integration with UPPAAL Stratego.
- Workflow analysis in the continuous time semantics.

Continuous Semantics vs. Discrete Semantics

Theorem (For Closed TAPNs)

Let M_0 be a marking with integer ages only. If

$$M_0 \xrightarrow{d_0,t_0} M_1 \xrightarrow{d_1,t_1} M_2 \xrightarrow{d_2,t_2} \dots \xrightarrow{d_{n-1},t_{n-1}} M_n$$

where $d_i \in \mathbb{R}^{\geq 0}$ then also

$$M_0 \xrightarrow{d'_0,t_0} M'_1 \xrightarrow{d'_1,t_1} M'_2 \xrightarrow{d'_2,t_2} \dots \xrightarrow{d'_{n-1},t_{n-1}} M'_n$$

where $d'_i \in \mathbb{N}_0$.

- We construct a set of linear inequalities that describe all possible delays allowed in the real-time execution.
- We only need difference constraints, hence the corresponding matrix in LP is totally unimodular.
- As the instance of LP has a real solution, it has also an optimal integral solution.

Continuous Semantics Implies Discrete Semantics

Theorem

If a timed-arc workflow net is sound in the continuous semantics then it is also sound in the discrete semantics.

Proof:

- Let N be sound in the continuous semantics.
- Let M be a marking reachable from the initial marking M_{in} in the discrete semantics.
- Hence some final marking M_{out} is reachable from M in the continuous semantics.
- We can conclude using the theorem that a marking M'_{out} with the same distribution of tokens as M_{out} is reachable from M also in the discrete semantics.

Discrete Semantics Implies Continuous Semantics

Theorem

If a timed-arc workflow net with no age invariants and no urgent transitions is sound in the discrete semantics then it is sound also in the continuous semantics.

Proof:

- We can arbitrarily delay in any marking.
- Hence the token ages exceed the maximum constants.
- Now there is no difference between discrete and continuous semantics.

Discrete Semantics Implies Continuous Semantics

Theorem

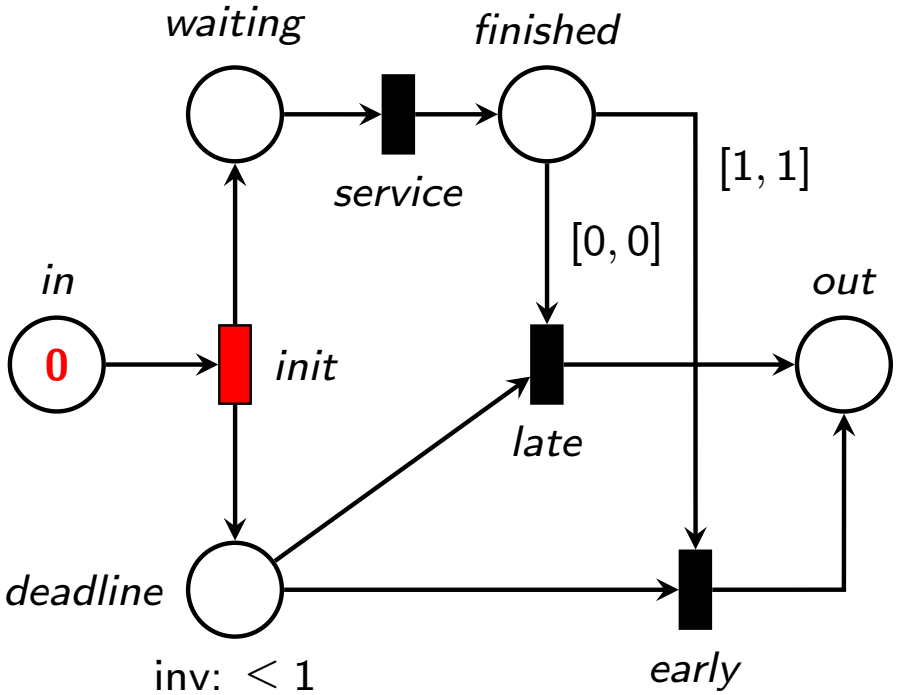
If a timed-arc workflow net with no age invariants and no urgent transitions is sound in the discrete semantics then it is sound also in the continuous semantics.

Proof:

- We can arbitrarily delay in any marking.
- Hence the token ages exceed the maximum constants.
- Now there is no difference between discrete and continuous semantics.

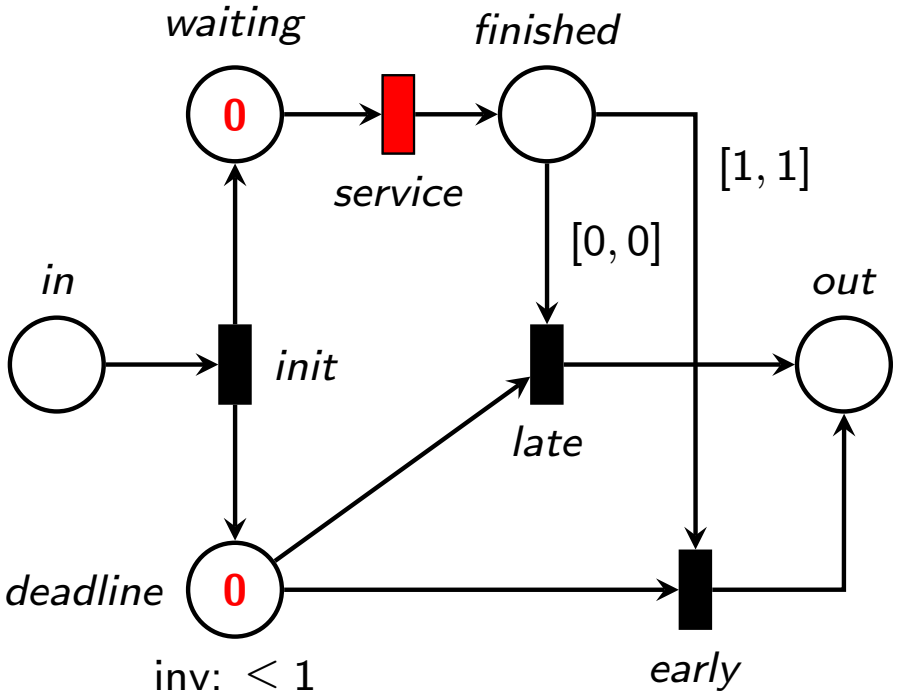
The theorem does not hold for general timed-arc workflow nets.

Continuous Semantics Challenge



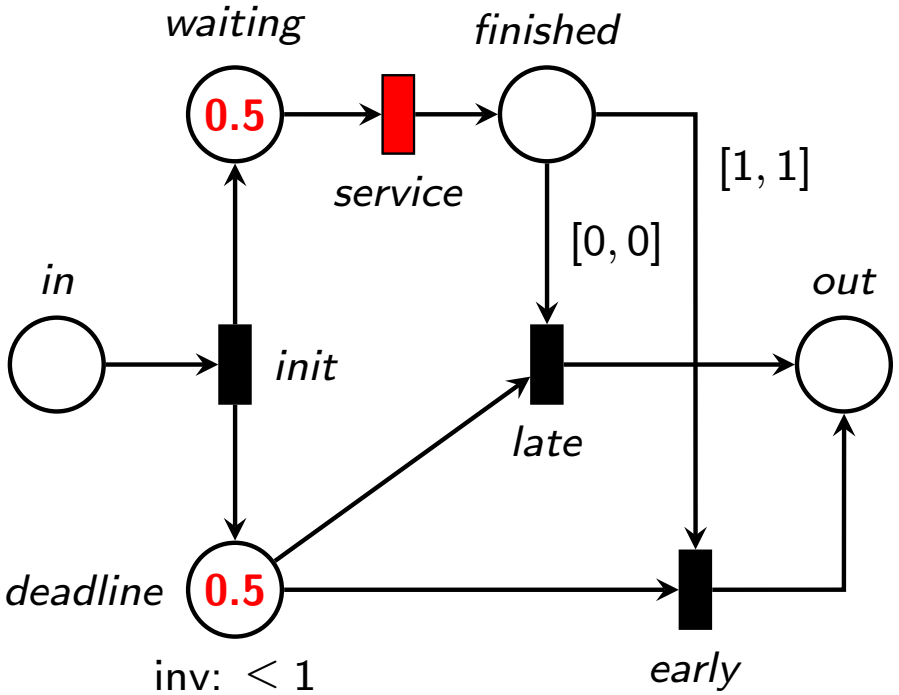
Sound in discrete semantics but unsound in continuous semantics.

Continuous Semantics Challenge



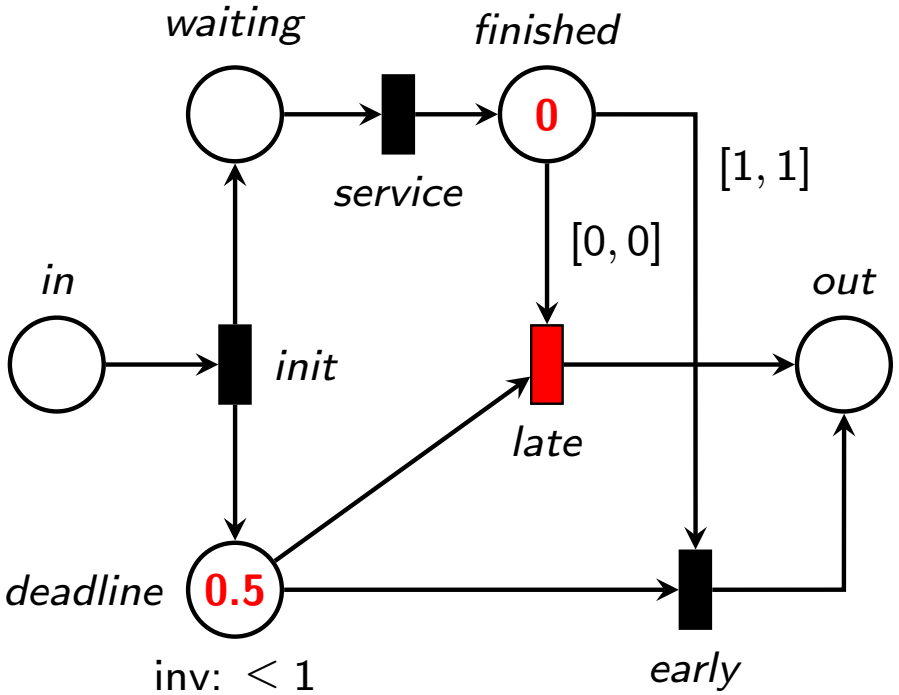
Sound in discrete semantics but unsound in continuous semantics.

Continuous Semantics Challenge



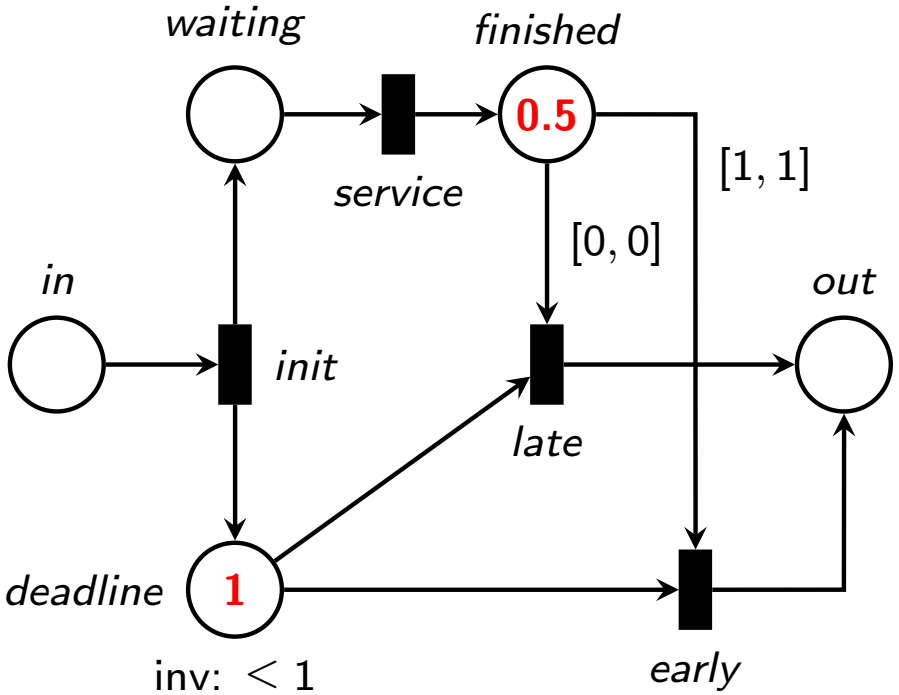
Sound in discrete semantics but unsound in continuous semantics.

Continuous Semantics Challenge



Sound in discrete semantics but unsound in continuous semantics.

Continuous Semantics Challenge



Sound in discrete semantics but unsound in continuous semantics.

Continuous Semantics Summary

- Continuous soundness implies discrete soundness.
- Opposite implication holds only for nets without urgency.
- Strong soundness is not an issue.

Continuous Semantics Summary

- Continuous soundness implies discrete soundness.
- Opposite implication holds only for nets without urgency.
- Strong soundness is not an issue.

Theorem

Let N be a workflow net is sound in the continuous-time semantics.

The net N is strongly sound in the discrete-time semantics iff it is strongly sound in the continuous-time semantics.

Conclusion

- Framework for the study of timed-arc workflow nets.
- Undecidability of soundness and strong soundness.
- Efficient algorithms for the decidable subclasses.
- Relationship to continuous soundness.
- Integration into the tool TAPAAL.

Conclusion

- Framework for the study of timed-arc workflow nets.
- Undecidability of soundness and strong soundness.
- Efficient algorithms for the decidable subclasses.
- Relationship to continuous soundness.
- Integration into the tool TAPAAL.

www.tapaal.net

Silver medal at Model Checking Contest 2014 and 2015.
(reachability category)