

Business Processes Modelling

MPB (6 cfu, 295AA)

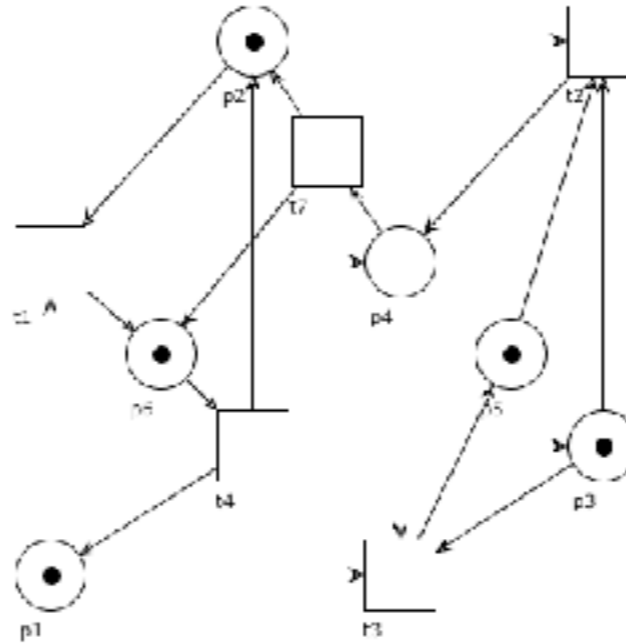
Roberto Bruni

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

08 - Petri nets basics



Object



Formalization of the basic concepts of
Petri nets

Free Choice Nets (book, optional reading)

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

Petri nets: basic definitions



Carl Adam Petri

July 12, 1926 - July 2, 2010

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

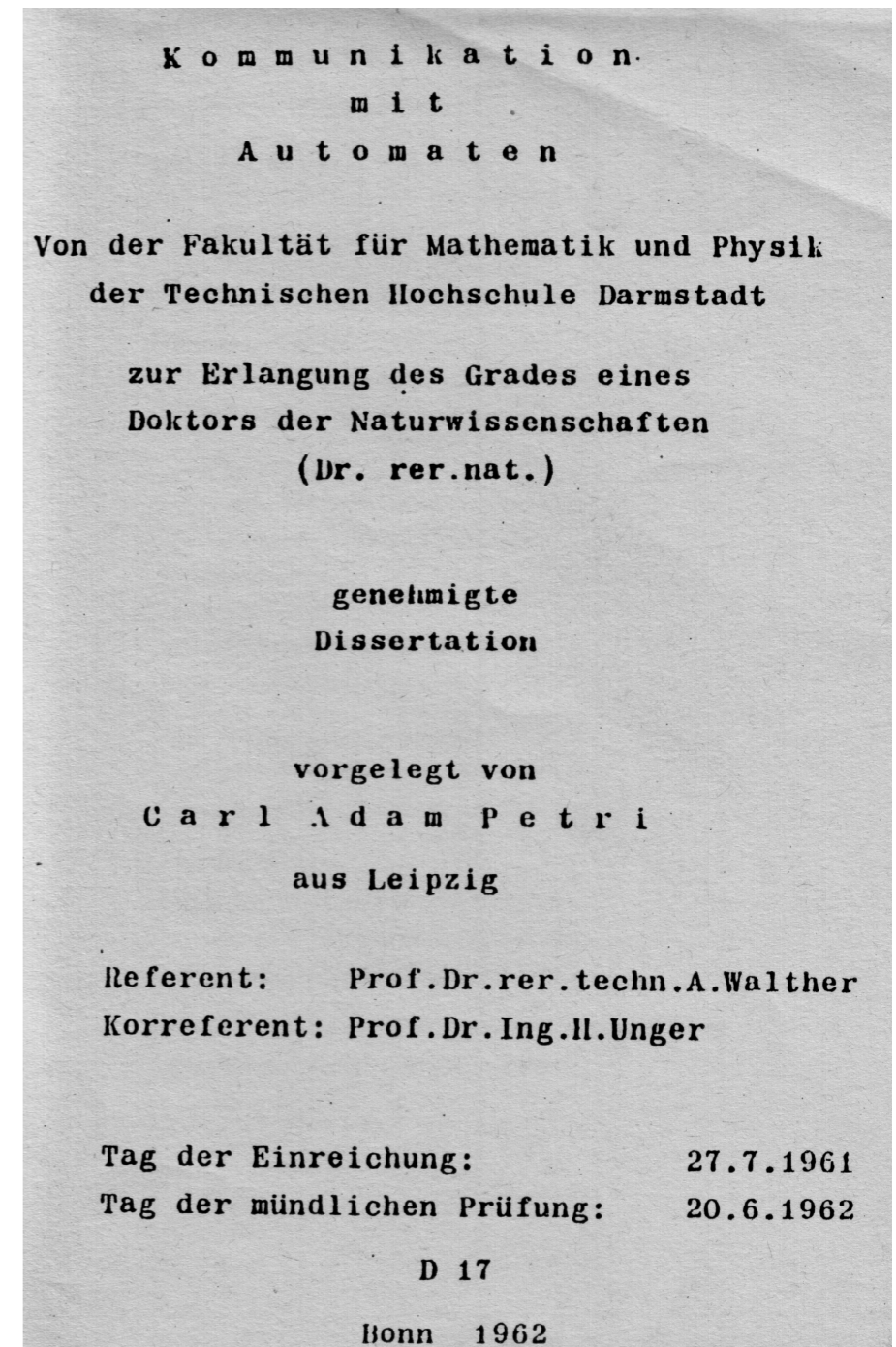
Introduced in 1962 (Petri's PhD thesis)

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

a memory location

...

Transitions

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

...

Tokens

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

...

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 $f : S \rightarrow \{0, 1\}$

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

Sets vs Multisets

Set



Multiset



Order of elements does not matter

Each element appears at most once

Order of elements does not matter

Each element can appear multiple times

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 $M_B(x) > 0$

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

Notation: sets

Empty set:

$\emptyset = \{ \}$ is such that $x \notin \emptyset$ for all $x \in S$

Set inclusion:

we write $A \subseteq B$ if $x \in A$ implies $x \in B$

Set strict inclusion:

we write $A \subset B$ if $A \subseteq B$ and $A \neq B$

Set union:

$A \cup B$ is the set s.t. $x \in (A \cup B)$ iff $x \in A$ or $x \in B$

Set difference:

$A - B$ is the set s.t. $x \in (A - B)$ iff $x \in A$ and $x \notin B$

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 $M \neq M'$

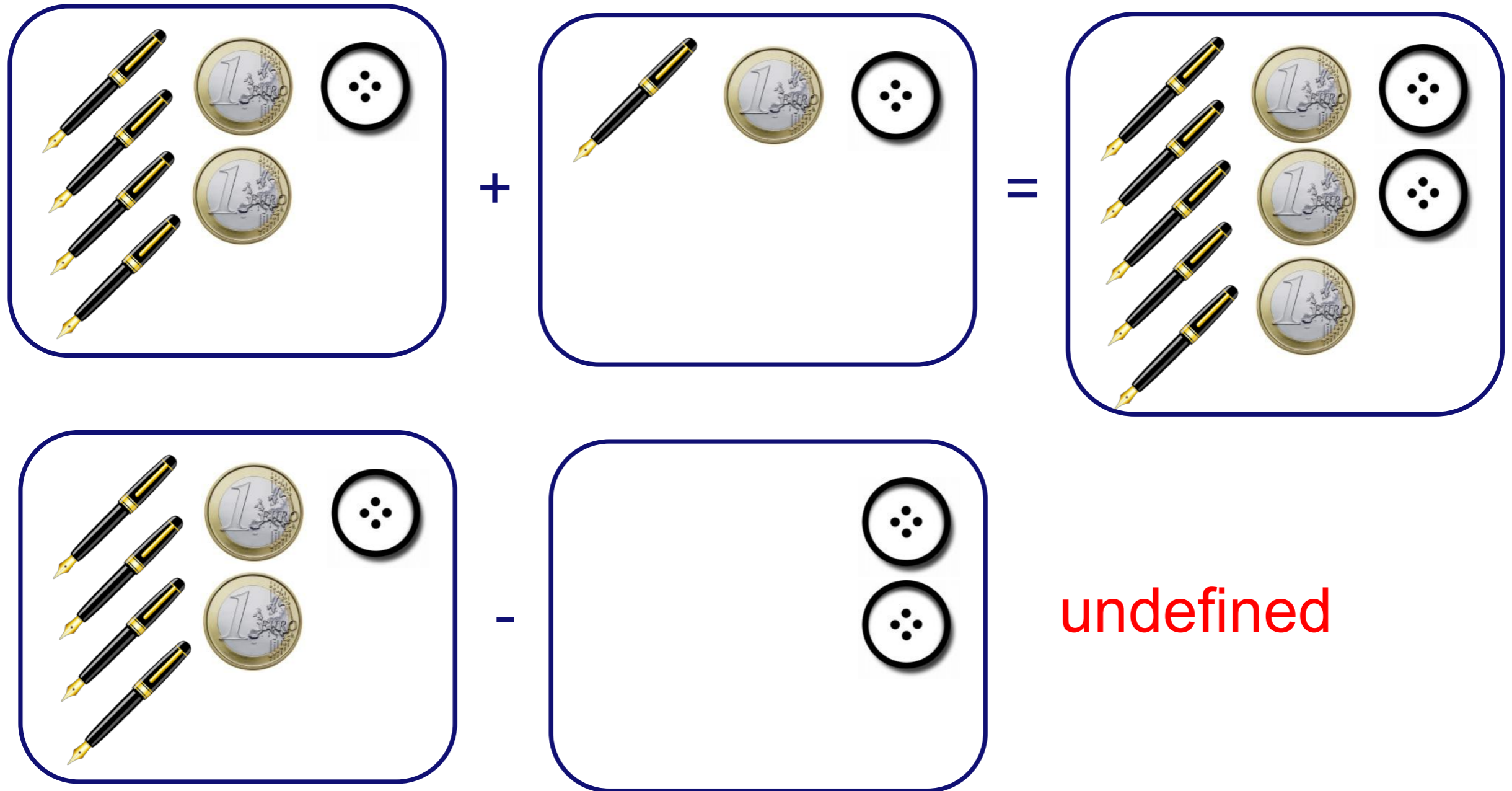
Multiset union:

$M + M'$ 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 $x \in S$

Operations on Multisets



Notation: multisets

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

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

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

Question time

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

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

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

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

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

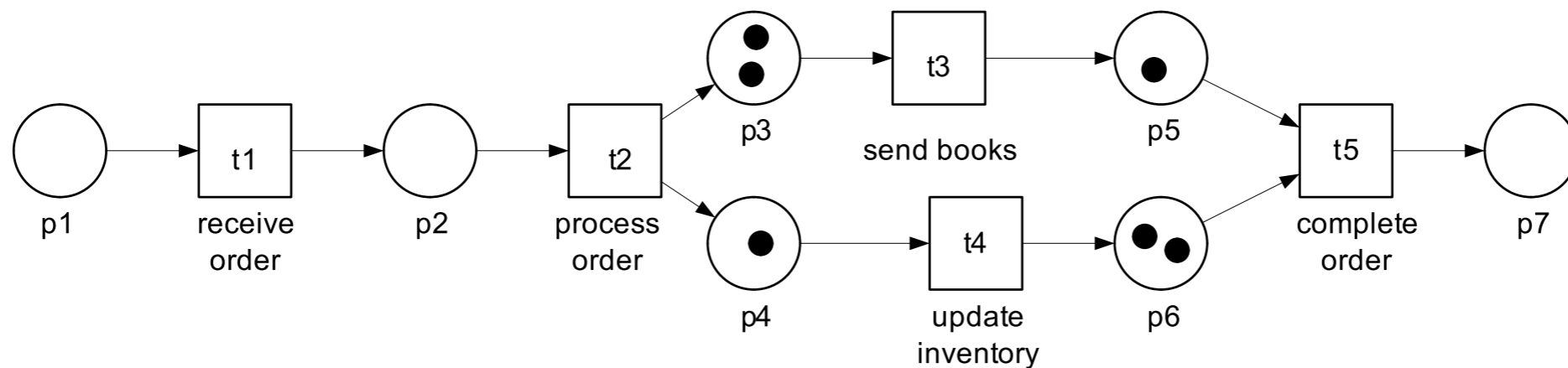
$$(2a + 2b) - (a + c) = ?$$

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. $M_0 \in \mu(P)$)

Example



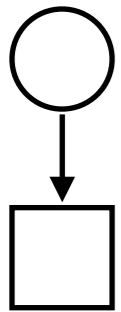
$$P = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7\}$$

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

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

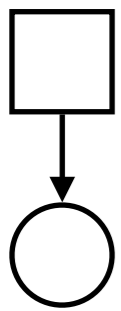
$$M_0 = 2p_3 + \dots ?$$

Pre-set and post-set



A place p is an input place for transition t iff
$$(p, t) \in F$$

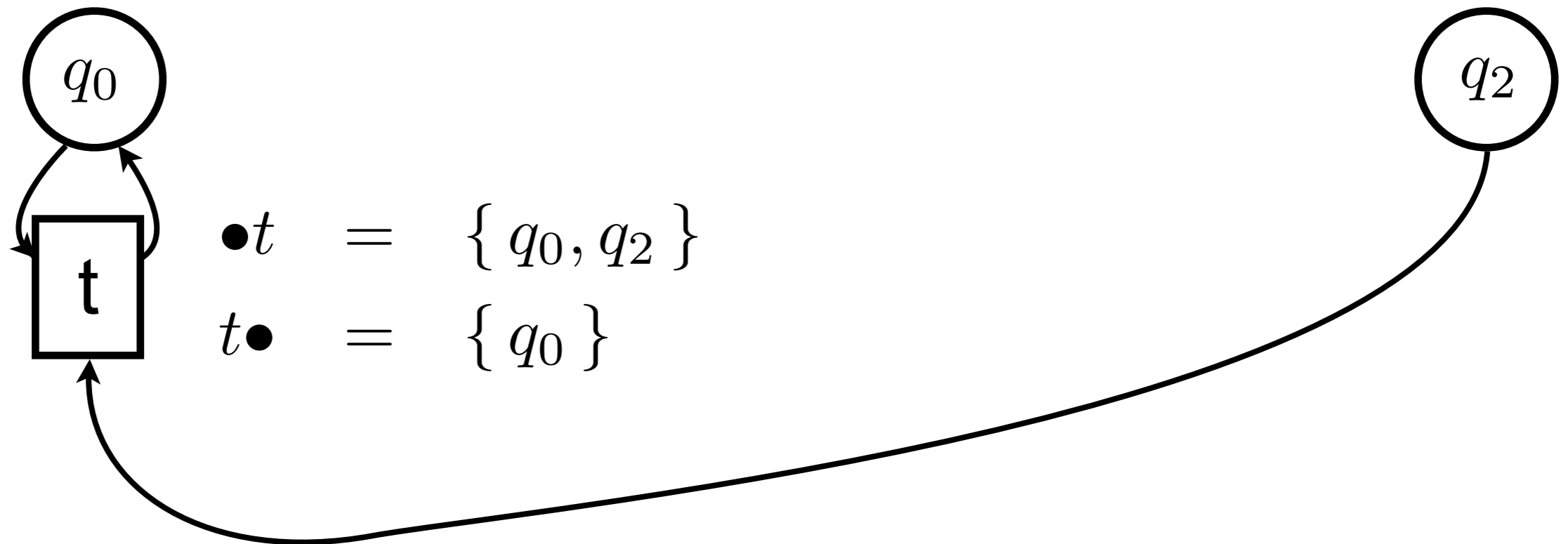
We let $\bullet t$ denote the set of input places of t .
(pre-set of t)



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

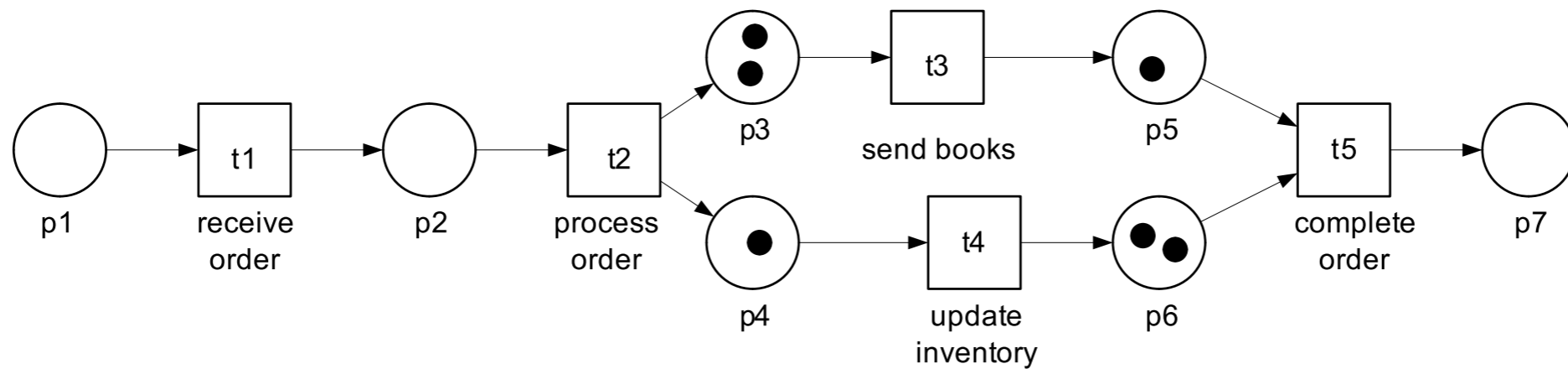
Analogously, we let

- 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:

$$\begin{aligned} \bullet x &= \{ y \mid (y, x) \in F \} && \text{pre-set} \\ x \bullet &= \{ y \mid (x, y) \in F \} && \text{post-set} \end{aligned}$$

Question time



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

- $t_1 = ?$
- $t_2 = ?$
- $t_3 = ?$
- $t_4 = ?$
- $t_5 = ?$

- $t_1 \bullet = ?$
- $t_2 \bullet = ?$
- $t_3 \bullet = ?$
- $t_4 \bullet = ?$
- $t_5 \bullet = ?$

- $p_1 = ?$
- $p_2 = ?$
- $p_3 = ?$
- $p_4 = ?$
- $p_5 = ?$
- $p_6 = ?$
- $p_7 = ?$

- $p_1 \bullet = ?$
- $p_2 \bullet = ?$
- $p_3 \bullet = ?$
- $p_4 \bullet = ?$
- $p_5 \bullet = ?$
- $p_6 \bullet = ?$
- $p_7 \bullet = ?$

Petri nets: enabling and firing

Enabling $M[t \rangle$

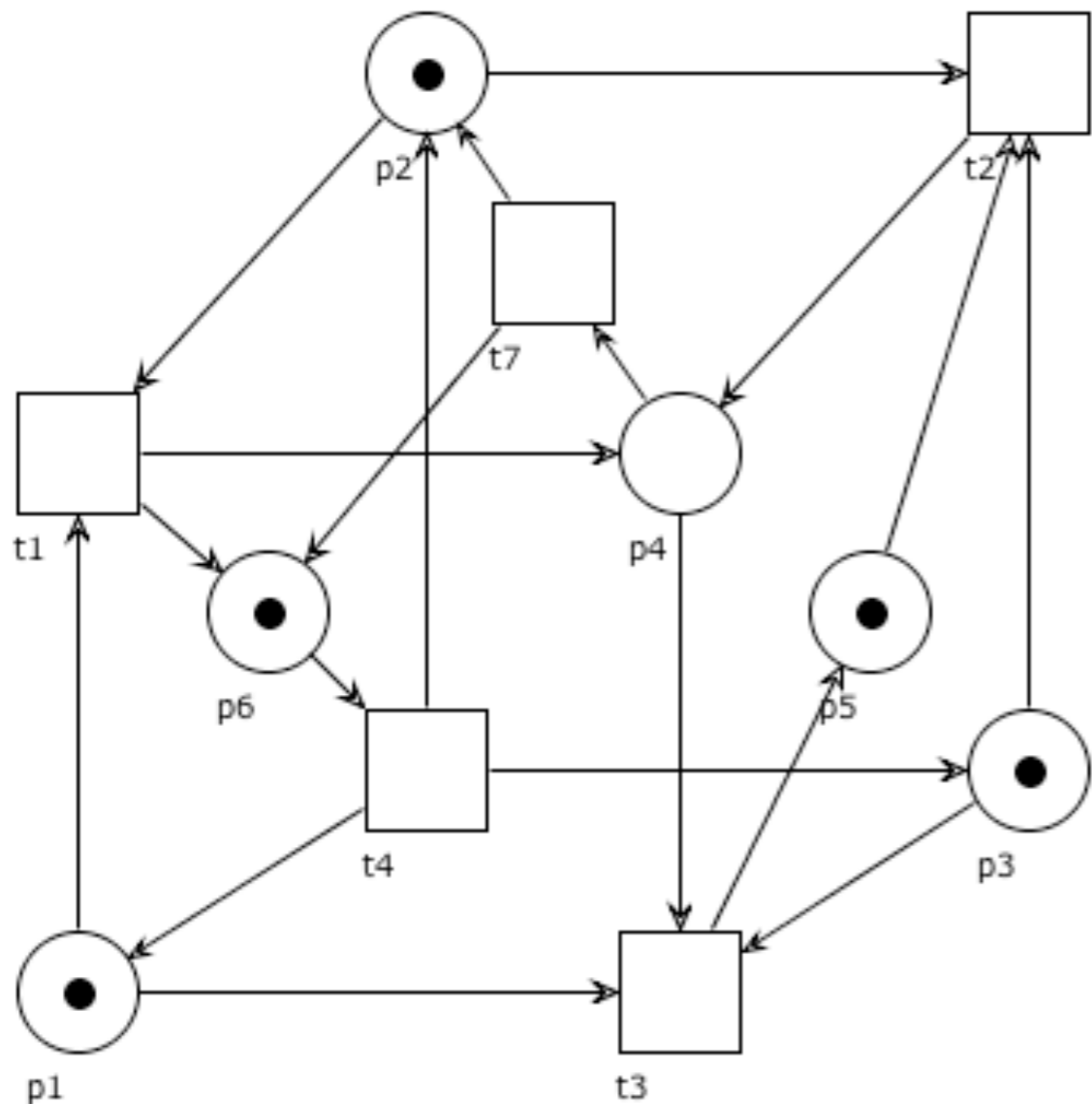
(a set can be seen
as a multiset
whose elements
have multiplicity 1)

A transition t is **enabled** at marking M iff $\bullet t \subseteq M$
and we write $M \xrightarrow{t}$ (also $M[t \rangle$)

A transition is enabled if each of its input places
contains at least one token

Question time

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$



Which of the following holds true?

- $M_0 \xrightarrow{t_1}$
- $M_0 \xrightarrow{t_2}$
- $M_0 \xrightarrow{t_3}$
- $M_0 \xrightarrow{t_7}$

Firing $M[t \rightarrow M'$

A transition t that is enabled at M can **fire**.

The **firing** of t at M changes the state to

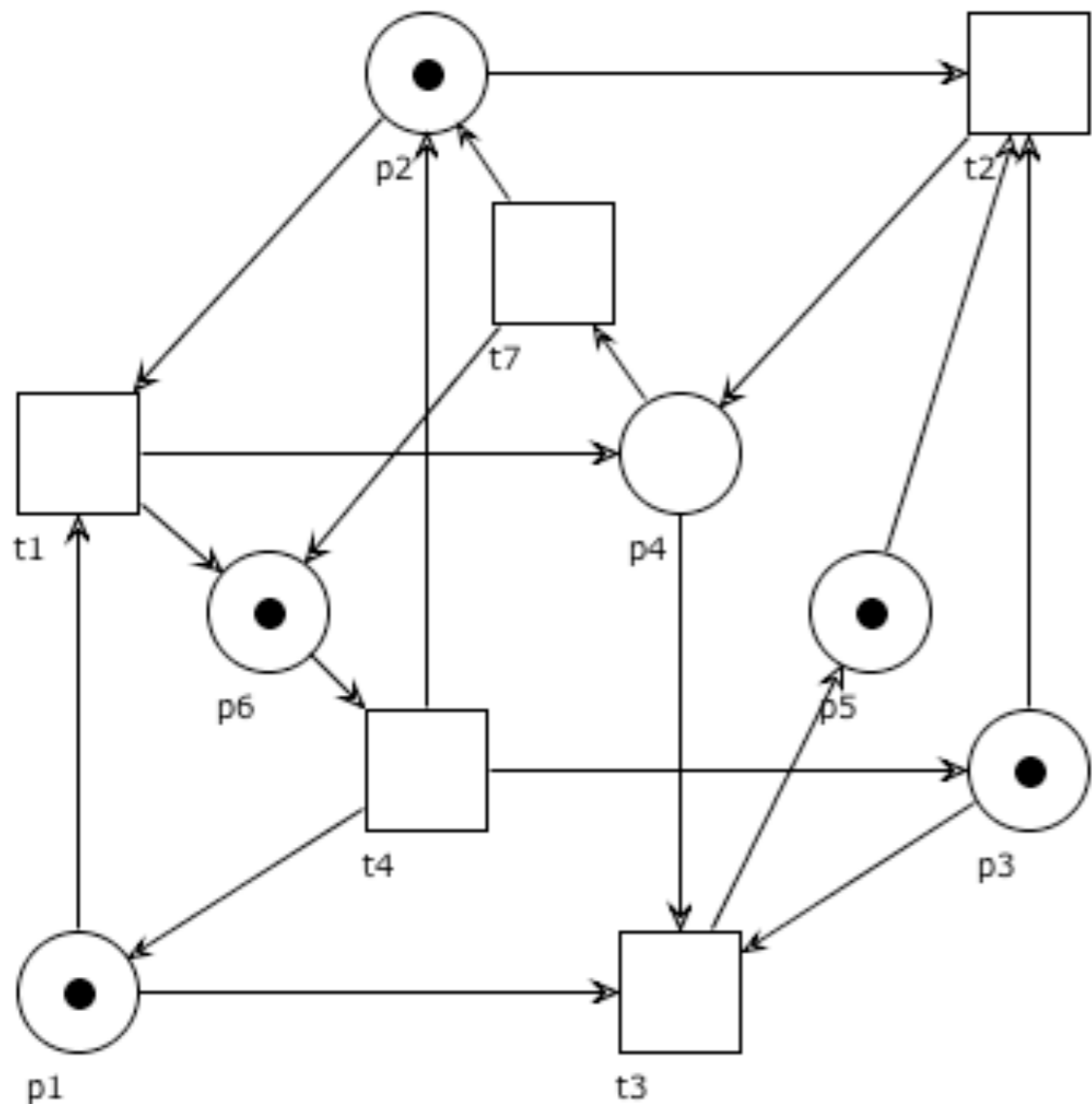
$$M' = M - \bullet t + t \bullet$$

and we write $M \xrightarrow{t} M'$ (also $M [t \rangle M'$)

When a transition fires
it consumes a token from each input place
it produces a token into each output place

Question time

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$



Which of the following holds true?

- $M_0 \xrightarrow{t_1} p_3 + p_4 + p_5 + p_6$
- $M_0 \xrightarrow{t_2} p_1 + p_4 + p_6$
- $M_0 \xrightarrow{t_4} 2p_1 + 2p_2 + 2p_3 + p_5$

Some remarks

Firing is an atomic action

Our semantics is interleaving:
multiple transitions may be enabled,
but only one fires at a time

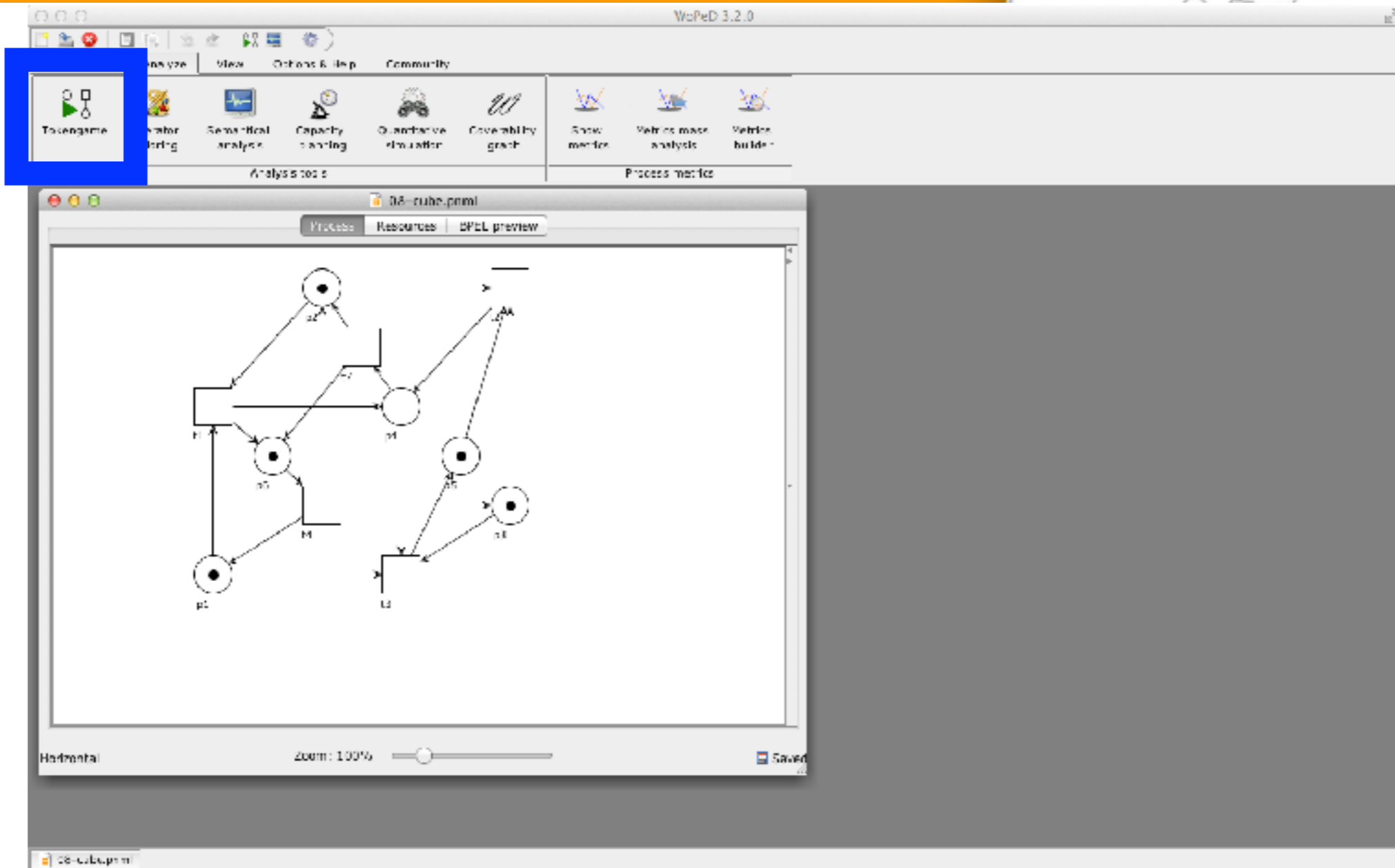
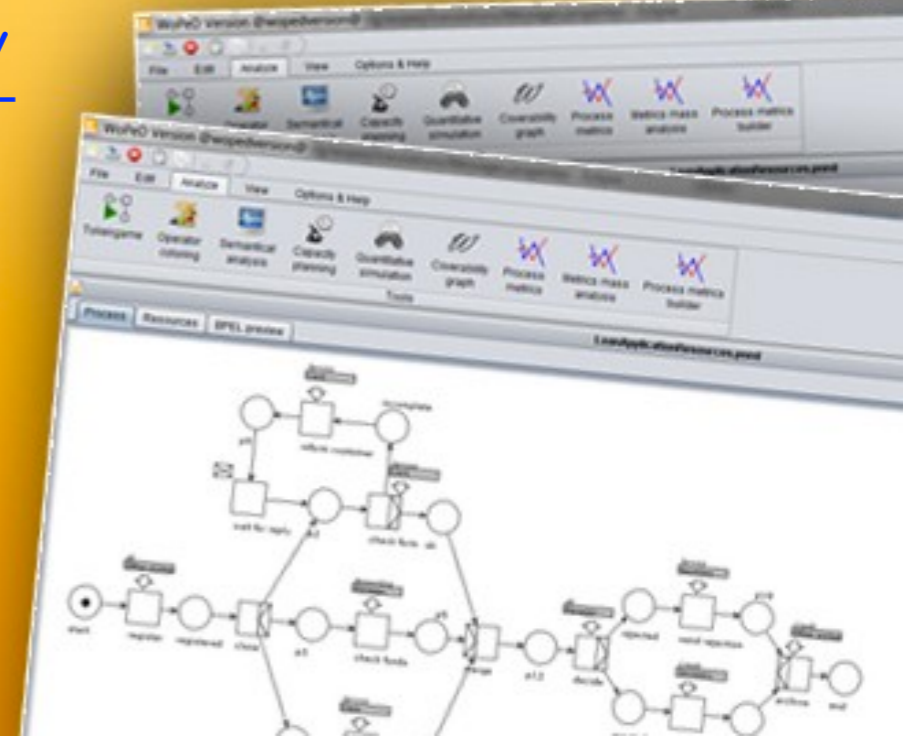
The network is static, but
the overall number of tokens may vary over time
(if transitions are fired for which the number of input
places is not equal to the number of output places)

<http://woped.dhbw-karlsruhe.de/woped/>

WoPeD (3.7.1)

Workflow Petri Net Designer

Download WoPeD at sourceforge!



Notation

We write $M \rightarrow$ if $M \xrightarrow{t}$ for some transition t

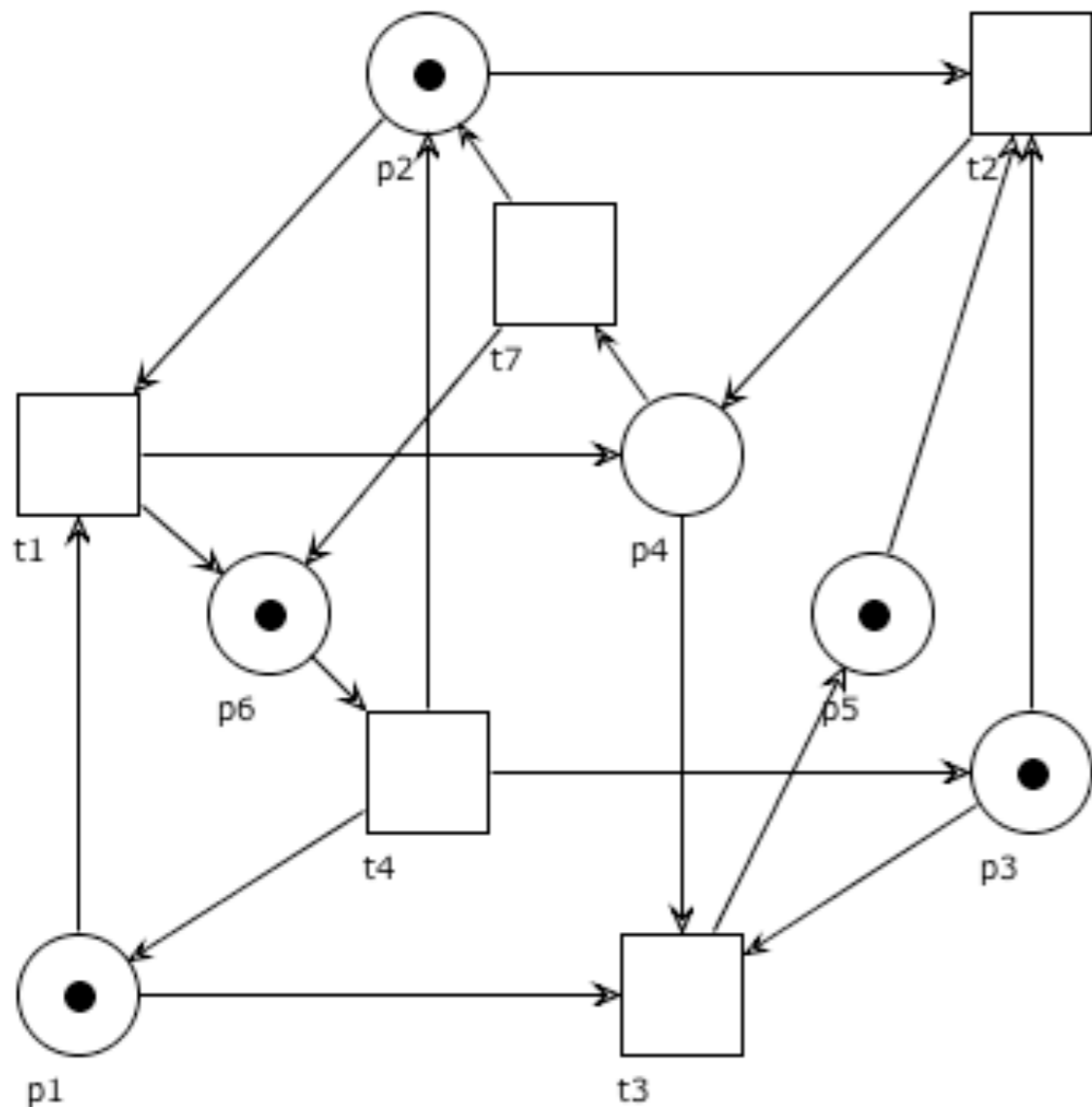
We write $M \rightarrow M'$ if $M \xrightarrow{t} M'$ for some transition t

We write $M \not\xrightarrow{t}$ if transition t is not enabled at M

We write $M \not\rightarrow$ if no transition is enabled at M

Example

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$



We can write that

- $M_0 \longrightarrow$
- $M_0 \longrightarrow p_1 + p_4 + p_6$ (by firing t2)
- $M_0 \not\stackrel{t_7}{\longrightarrow}$
- $p_1 + p_5 \not\longrightarrow$

Firing sequence

Let $\sigma = t_1 t_2 \dots t_{n-1} \in T^*$ be a sequence of transitions.

We write $M \xrightarrow{\sigma} M'$ (and $M \xrightarrow{\sigma}$) if:

there is a sequence of markings M_1, \dots, M_n

with $M = M_1$ and $M' = M_n$

and $M_i \xrightarrow{t_i} M_{i+1}$ for $1 \leq i < n$

(i.e. $M = M_1 \xrightarrow{t_1} M_2 \xrightarrow{t_2} \dots \xrightarrow{t_{n-1}} M_n = M'$)

Reachable markings $[M\rangle$

We write $M \xrightarrow{*} M'$ if $M \xrightarrow{\sigma} M'$ for some $\sigma \in T^*$

A marking M' is **reachable from** M if $M \xrightarrow{*} M'$

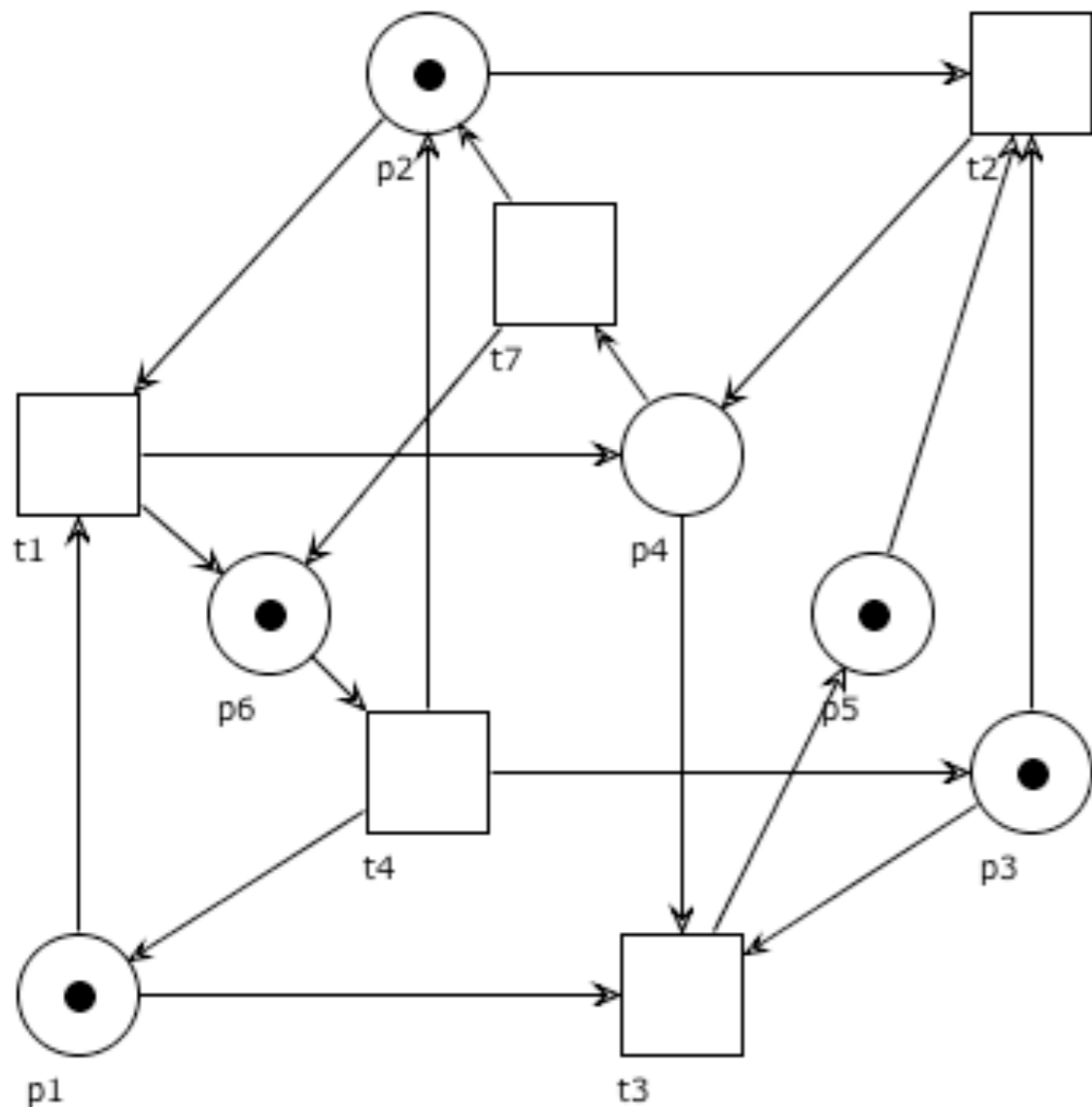
Note that $M \xrightarrow{\epsilon} M$ for ϵ the empty sequence

The set of markings reachable from M is often denoted:

reach(M) or also $[M\rangle$

Question time

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$

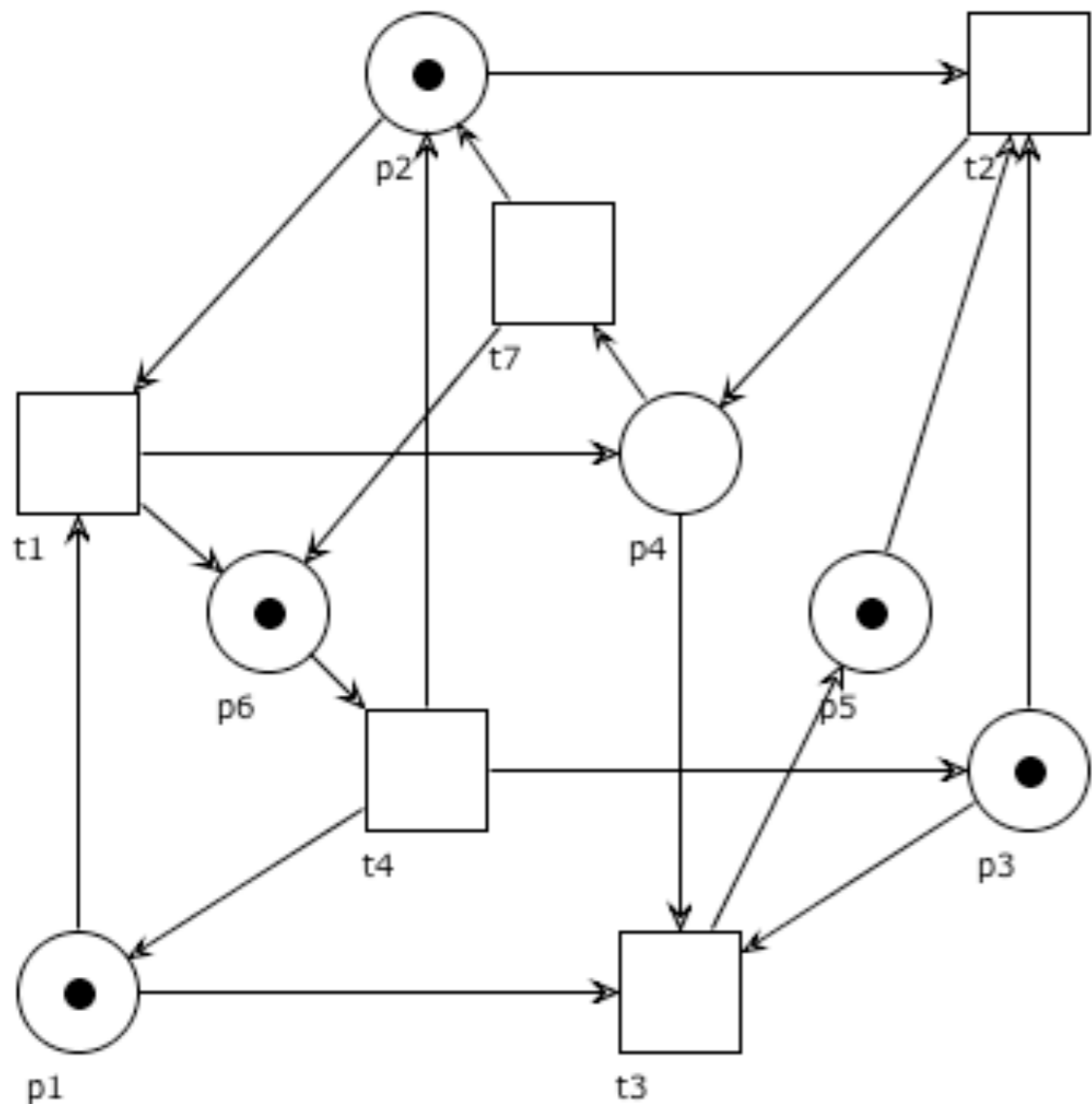


Which of the following holds true?

- $M_0 \xrightarrow{t_1 t_4 t_2 t_3}$
- $M_0 \xrightarrow{t_2 t_7 t_4}$
- $M_0 \xrightarrow{t_1 t_2 t_7}$
- $M_0 \xrightarrow{t_1 t_4 t_2 t_1}$

Example

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$



We have that

- $M_0 \xrightarrow{t_1 t_4 t_2 t_3} p_4 + p_5 + p_6$
- $M_0 \xrightarrow{t_2 t_7 t_4} 2p_1 + 2p_2 + p_3 + p_6$
- $M_0 \xrightarrow{t_1 t_4 t_3 t_2 t_7} p_2 + p_5 + 2p_6$

Infinite sequence

Let $\sigma = t_1 t_2 \dots \in T^\omega$ be an infinite sequence of transitions.

We write $M \xrightarrow{\sigma}$ if:

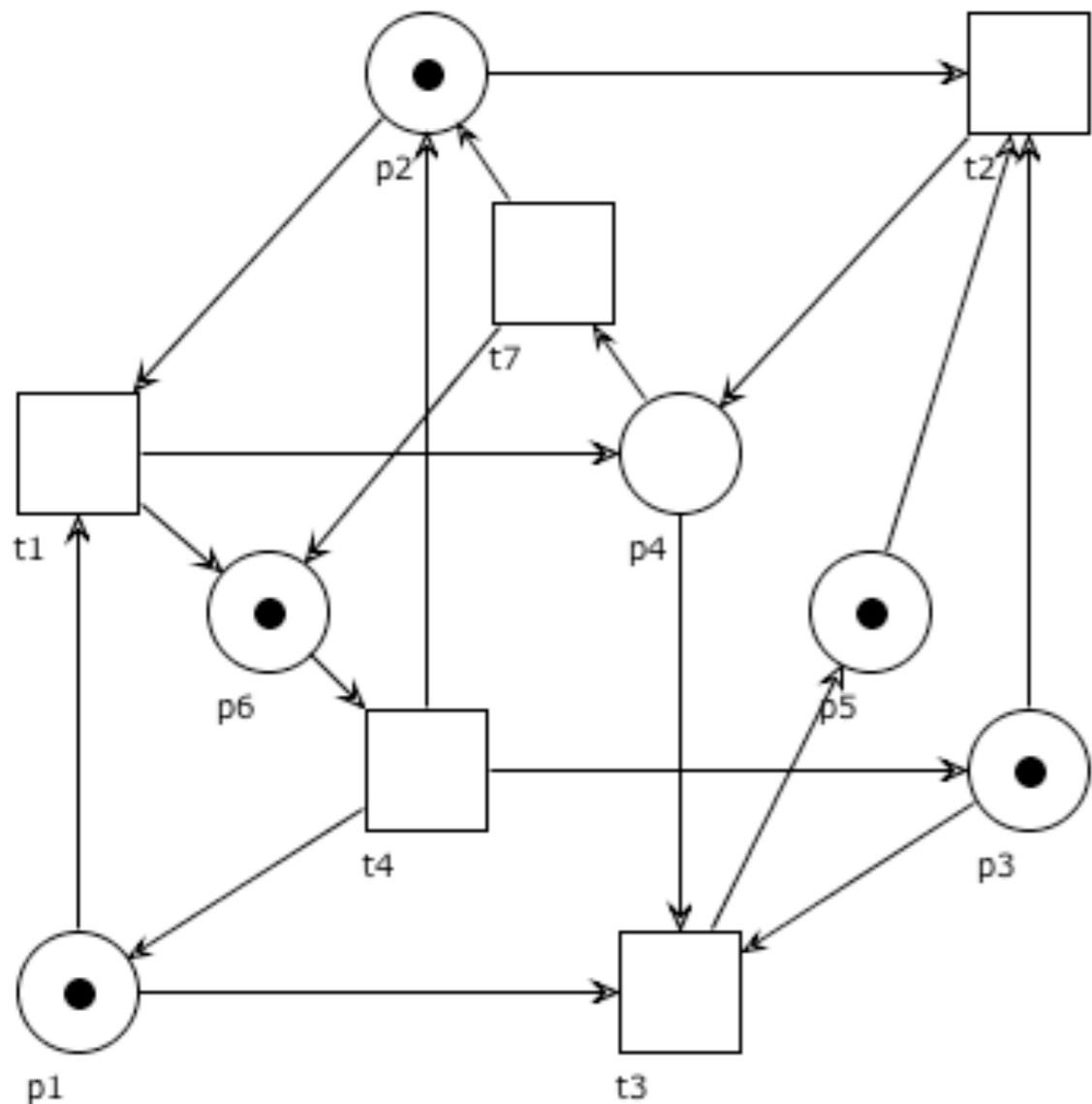
there is an infinite sequence of markings M_1, M_2, \dots

with $M = M_1$ and $M_i \xrightarrow{t_i} M_{i+1}$ for $1 \leq i$

(i.e. $M = M_1 \xrightarrow{t_1} M_2 \xrightarrow{t_2} \dots$)

Example

$$M_0 = p_1 + p_2 + p_3 + p_5 + p_6$$



We have that

- $M_0 \xrightarrow{t_1 t_4 t_1 t_4 t_1 t_4 \dots}$
- $M_0 \xrightarrow{t_1 t_4 t_7 t_1 t_4 t_7 t_1 t_4 t_7 \dots}$

Enabled sequence

We say that an occurrence sequence σ is **enabled** if $M \xrightarrow{\sigma}$

(σ can be finite or infinite)

Note that an infinite sequence can be represented as a map $\sigma : \mathbb{N} \rightarrow T$, where $\sigma(i) = t_i$

More on sequences: concatenation & prefix

Concatenation:

finite + finite = finite

for $\sigma_1 = a_1 \dots a_n$ and $\sigma_2 = b_1 \dots b_m$, we let $\sigma_1 \sigma_2 = a_1 \dots a_n b_1 \dots b_m$

for $\sigma_1 = a_1 \dots a_n$ and $\sigma_2 = b_1 b_2 \dots$, we let $\sigma_1 \sigma_2 = a_1 \dots a_n b_1 b_2 \dots$

finite + infinite = infinite

σ is a **prefix** of σ' if $\sigma = \sigma'$ or $\sigma \sigma'' = \sigma'$ for some $\sigma'' \neq \epsilon$

σ is a **proper prefix** of σ' if $\sigma \sigma'' = \sigma'$ for some $\sigma'' \neq \epsilon$

Enabledness

Proposition: $M \xrightarrow{\sigma}$ iff $M \xrightarrow{\sigma'}$ for every prefix σ' of σ

(\Rightarrow) immediate from definition

(\Leftarrow) trivial if σ is finite (σ itself is a prefix of σ)

When σ is infinite: taken any $i \in \mathbb{N}$ we need to prove that $t_i = \sigma(i)$ is enabled after the firing of the prefix $\sigma' = t_1 t_2 \dots t_{i-1}$ of σ .

But this is obvious, because

$$M \xrightarrow{t_1} M_1 \xrightarrow{t_2} \dots \xrightarrow{t_{i-1}} M_{i-1} \xrightarrow{t_i} M_i$$

is also a finite prefix of σ and therefore $M_{i-1} \xrightarrow{t_i}$

More on sequences: projection

Restriction: (also extraction / projection)
given $T' \subseteq T$ we inductively define $\sigma|_{T'}$ as:

$$\epsilon|_{T'} = \epsilon \quad (t\sigma)|_{T'} = \begin{cases} t(\sigma|_{T'}) & \text{if } t \in T' \\ \sigma|_{T'} & \text{if } t \notin T' \end{cases}$$

Example

$$(t_1 t_4 t_7 t_1 t_4 t_7) |_{\{t_1, t_4\}} = t_1 (t_4 t_7 t_1 t_4 t_7) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 (t_7 t_1 t_4 t_7) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 (t_1 t_4 t_7) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 t_1 (t_4 t_7) |_{\{t_1, t_4\}}$$

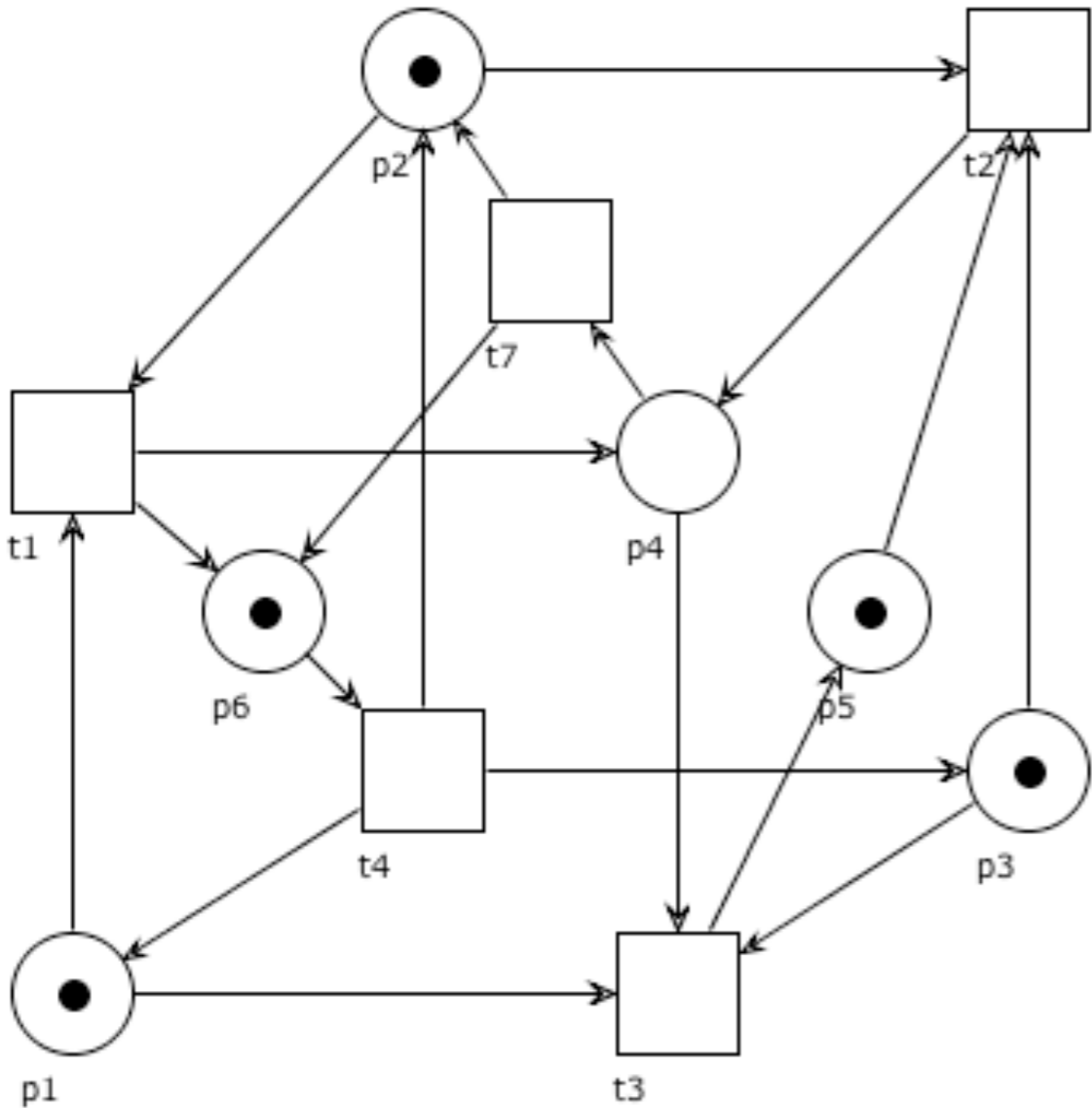
$$= t_1 t_4 t_1 t_4 (t_7) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 t_1 t_4 (t_7 \epsilon) |_{\{t_1, t_4\}}$$

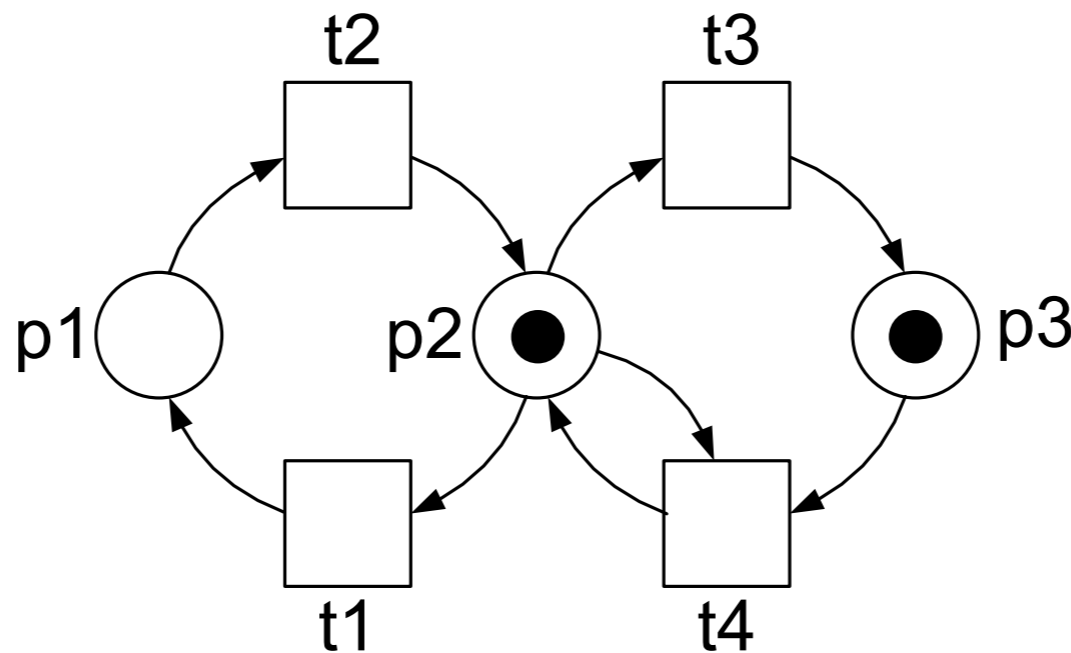
$$= t_1 t_4 t_1 t_4 (\epsilon) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 t_1 t_4 \epsilon$$

$$= t_1 t_4 t_1 t_4$$



Exercises



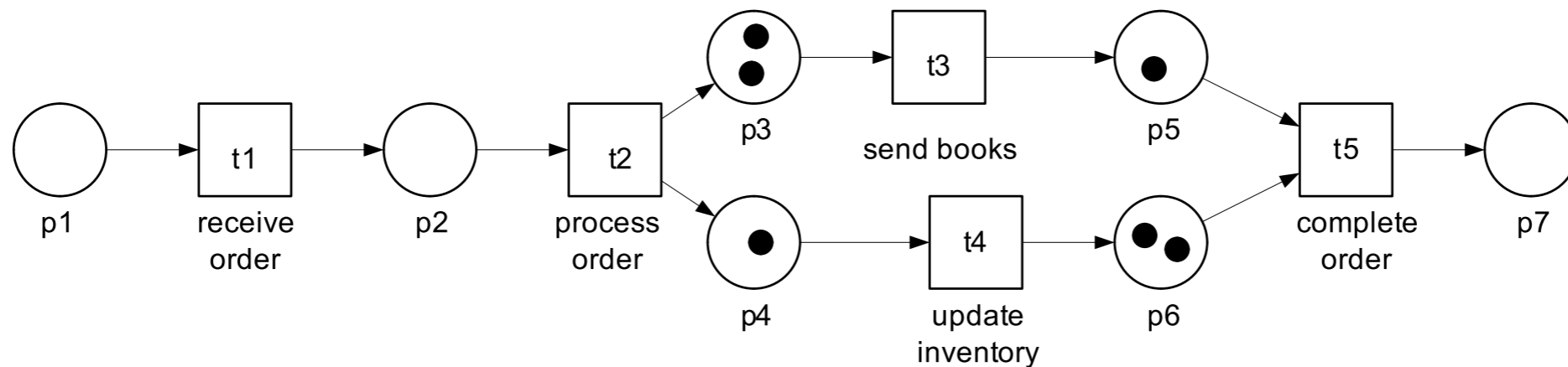
Determine the pre- and post-set of each element

Which are the currently enabled transitions?

For each of them, which state would the firing lead to?

What are the reachable states?

Exercises



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

Which are the currently enabled transitions?

For each of them, which state would the firing lead to?

What are the reachable states?

Petri nets: occurrence graph

Occurrence graph (aka Reachability graph)

The reachability graph is a graph that represents all possible occurrence sequences of a net

Nodes of the graphs = reachable markings
Arcs of the graphs = firings

Formally, $OG(N) = ([M_0], A)$ where $A \subseteq [M_0] \times T \times [M_0]$ s.t.

$$(M, t, M') \in A \quad \text{iff} \quad M \xrightarrow{t} M'$$

How to compute $OG(N)$

1. Initially $R = \{ M_0 \}$ and $A = \emptyset$

How to compute $OG(N)$

1. Initially $R = \{ M_0 \}$ and $A = \emptyset$
2. Take a marking $M \in R$ and a transition $t \in T$ such that
 1. M enables t and there is no arc labelled t leaving from M

How to compute $OG(N)$

1. Initially $R = \{ M_0 \}$ and $A = \emptyset$
2. Take a marking $M \in R$ and a transition $t \in T$ such that
 1. M enables t and there is no arc labelled t leaving from M
3. Let $M' = M - \cdot t + t \cdot$

How to compute $OG(N)$

1. Initially $R = \{ M_0 \}$ and $A = \emptyset$
2. Take a marking $M \in R$ and a transition $t \in T$ such that
 1. M enables t and there is no arc labelled t leaving from M
3. Let $M' = M - \cdot t + t \cdot$
4. Add M' to R and (M, t, M') to A

How to compute $OG(N)$

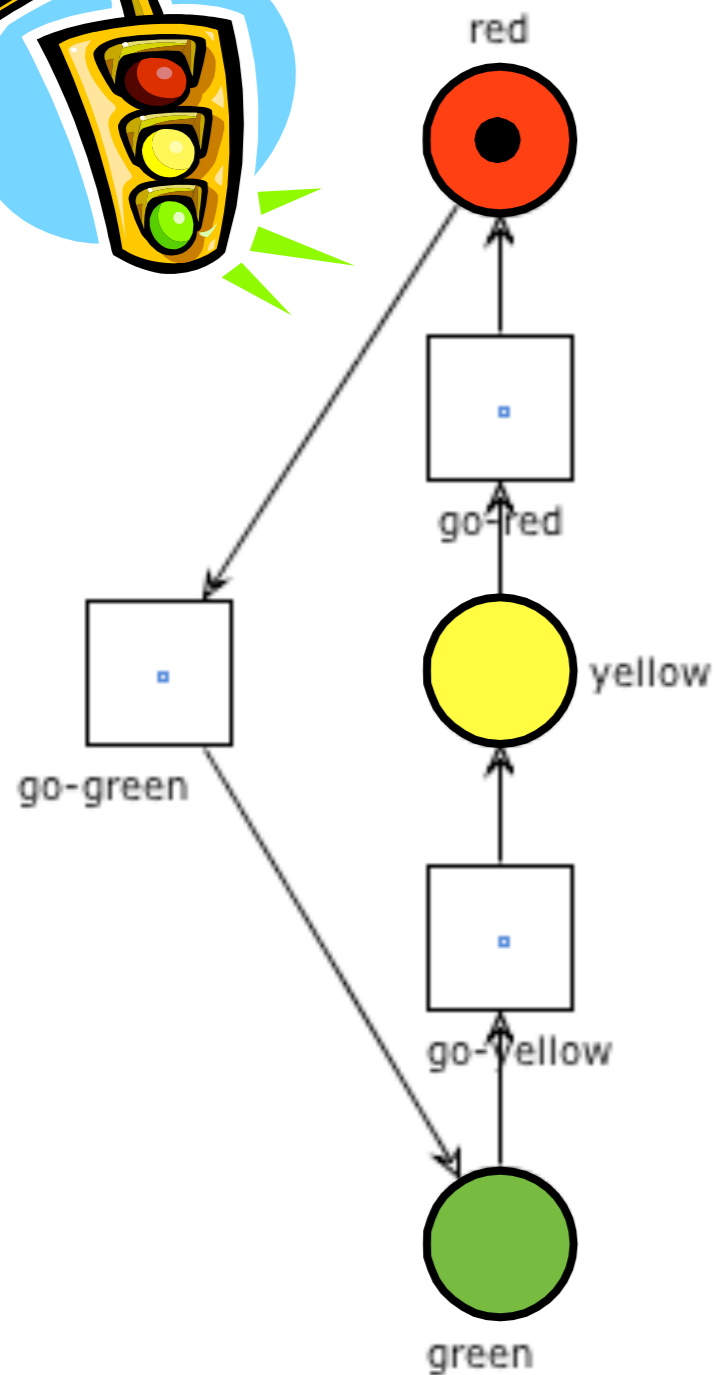
1. Initially $R = \{ M_0 \}$ and $A = \emptyset$
2. Take a marking $M \in R$ and a transition $t \in T$ such that
 1. M enables t and there is no arc labelled t leaving from M
3. Let $M' = M - \cdot t + t \cdot$
4. Add M' to R and (M, t, M') to A
5. Repeat steps 2,3,4 until no new arc can be added

How to compute $OG(N)$

The occurrence graph can be constructed as follows:

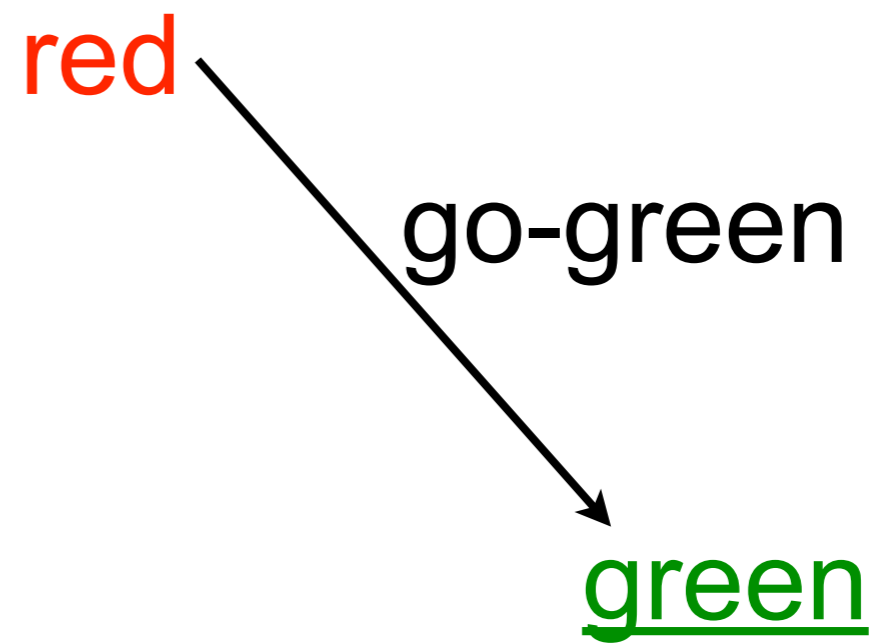
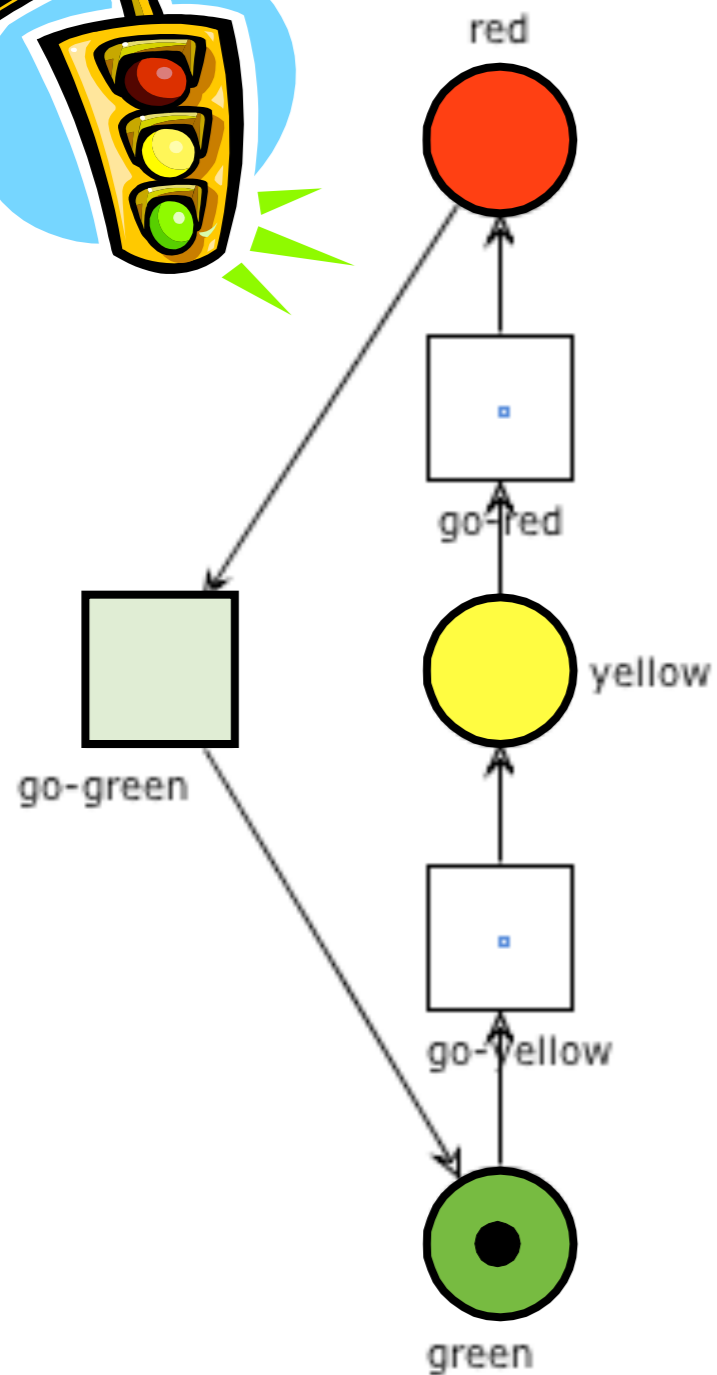
1. $Nodes = \{\}, Arcs = \{\}, Todo = \{M_0\}$
2. $M = next(Todo)$
3. $Nodes = Nodes \cup \{M\}, Todo = Todo \setminus \{M\}$
4. $Firings = \{(M, t, M') \mid \exists t \in T, \exists M' \in \mu(P), M \xrightarrow{t} M'\}$
5. $New = \{M' \mid (M, t, M') \in Firings\} \setminus (Nodes \cup Todo)$
6. $Todo = Todo \cup New, Arcs = Arcs \cup Firings$
7. $isEmpty(Todo) ? \text{stop} : \text{goto } 2$

Example: traffic light

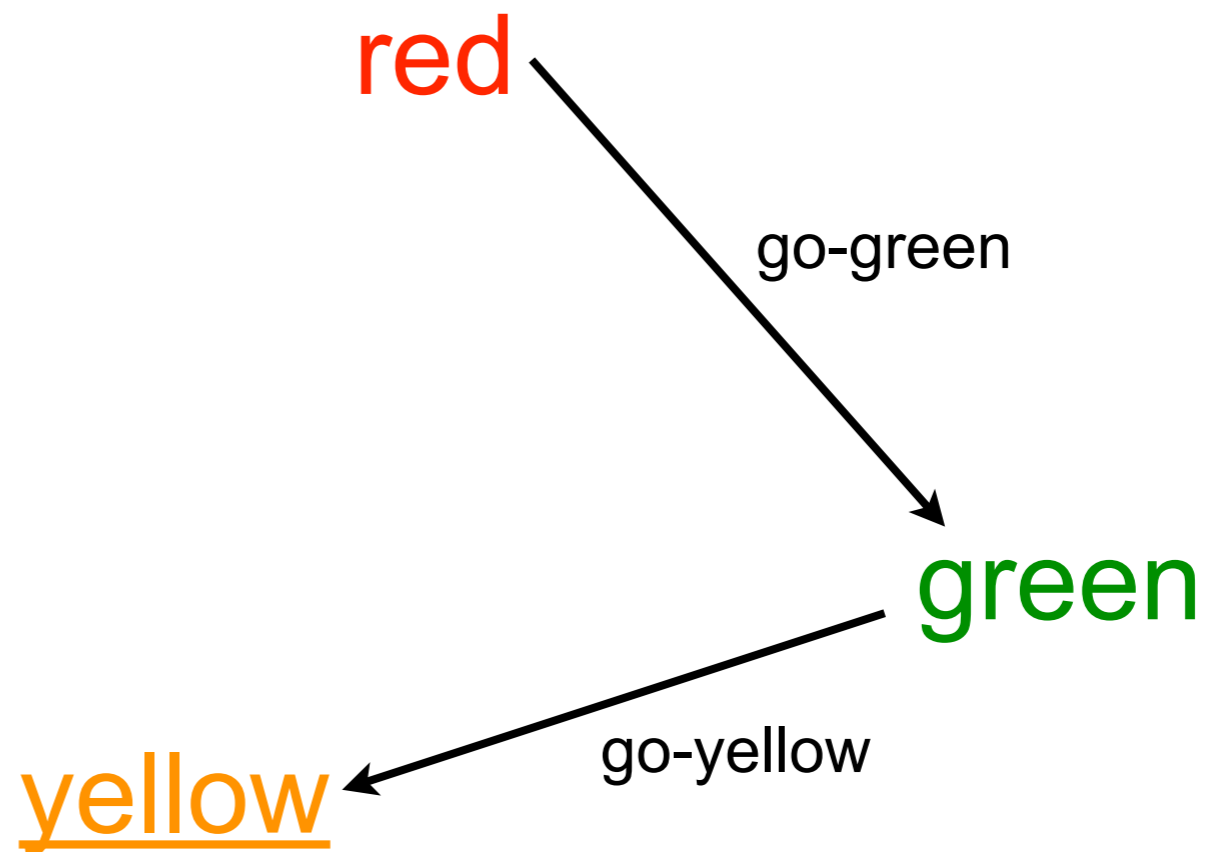
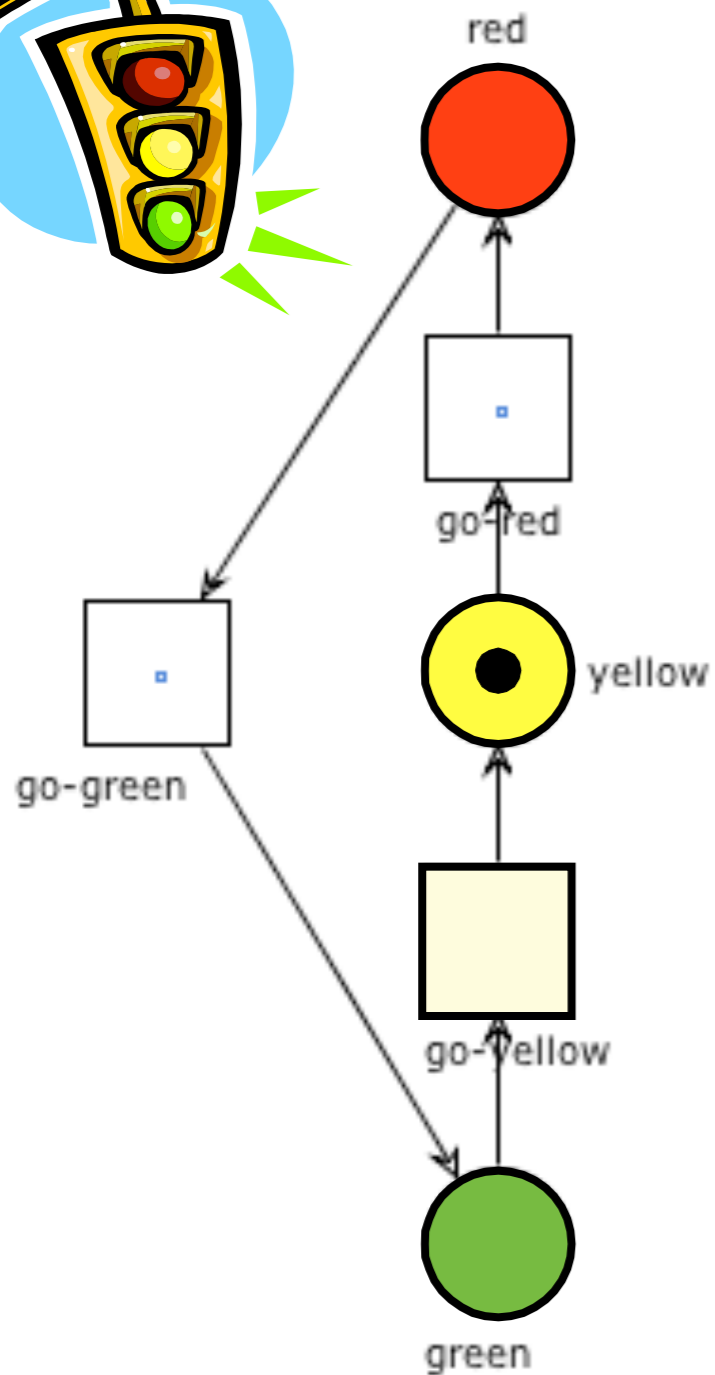


red

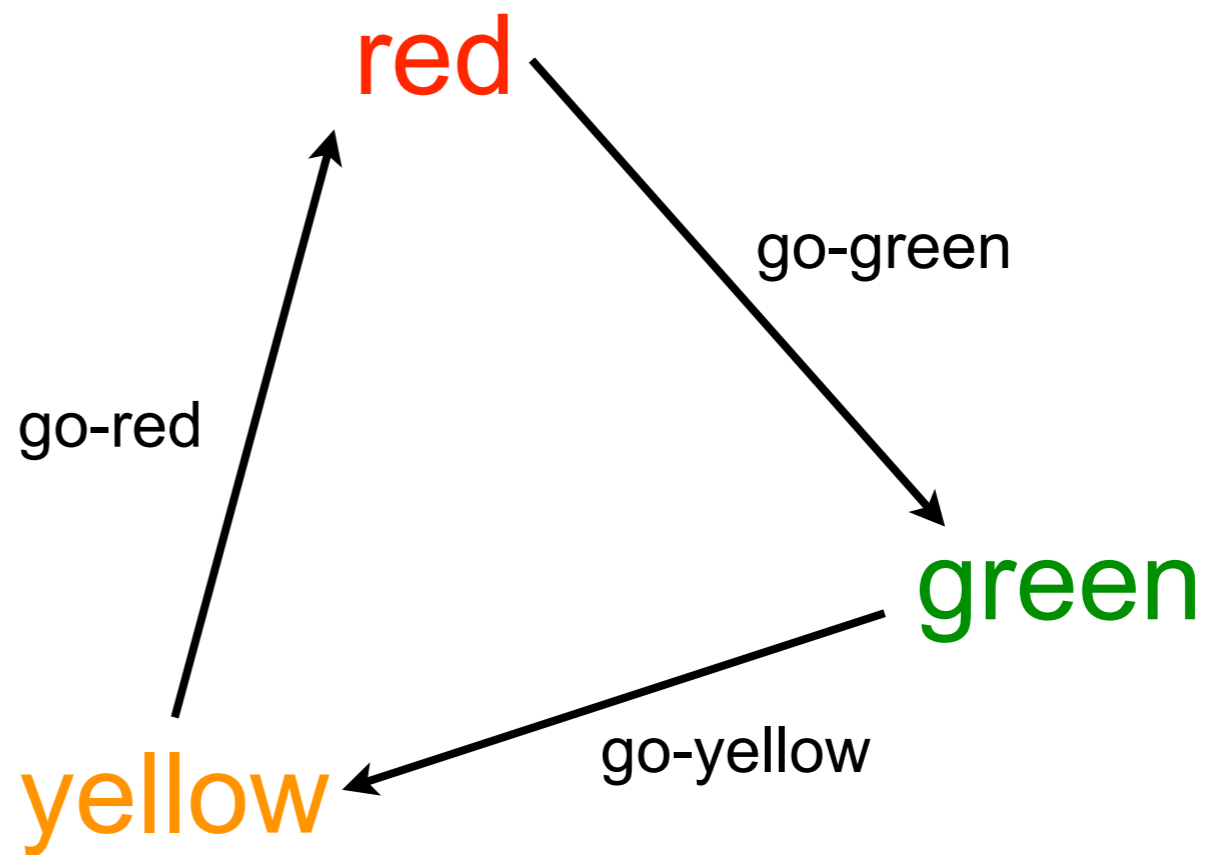
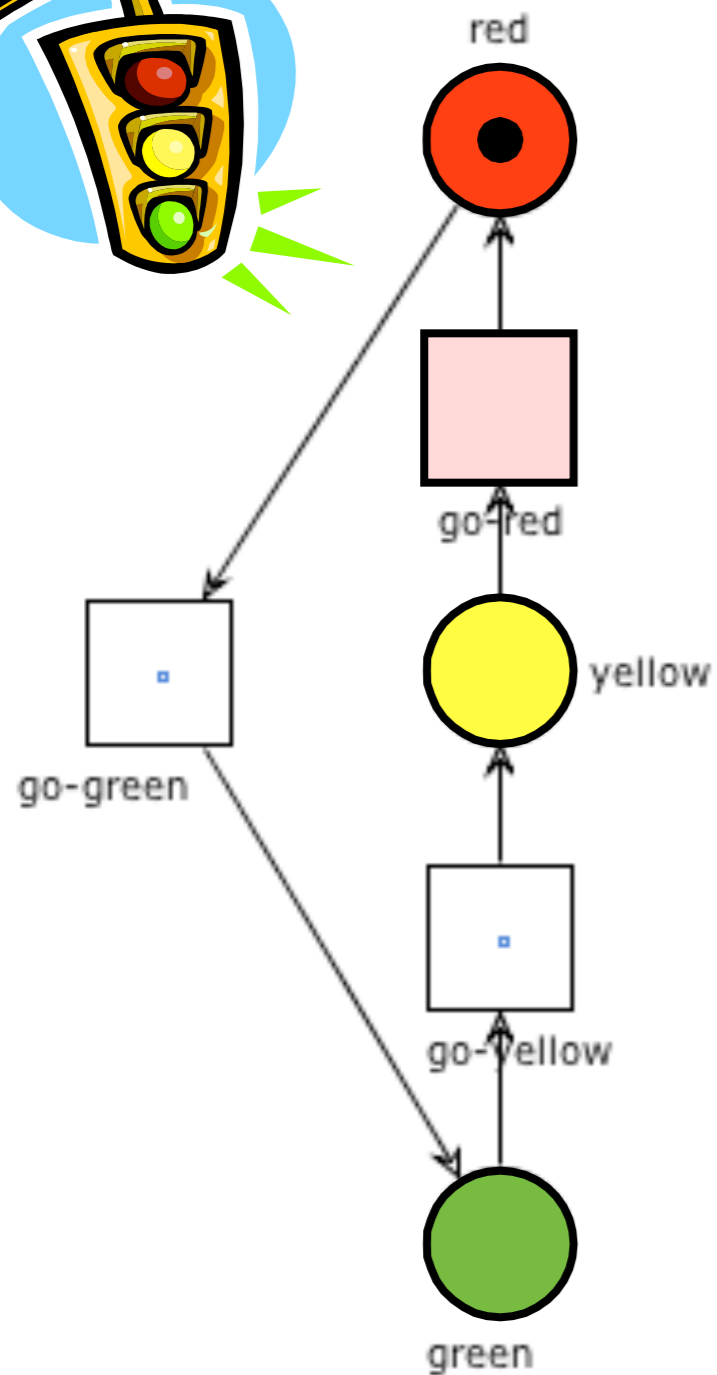
Example: traffic light



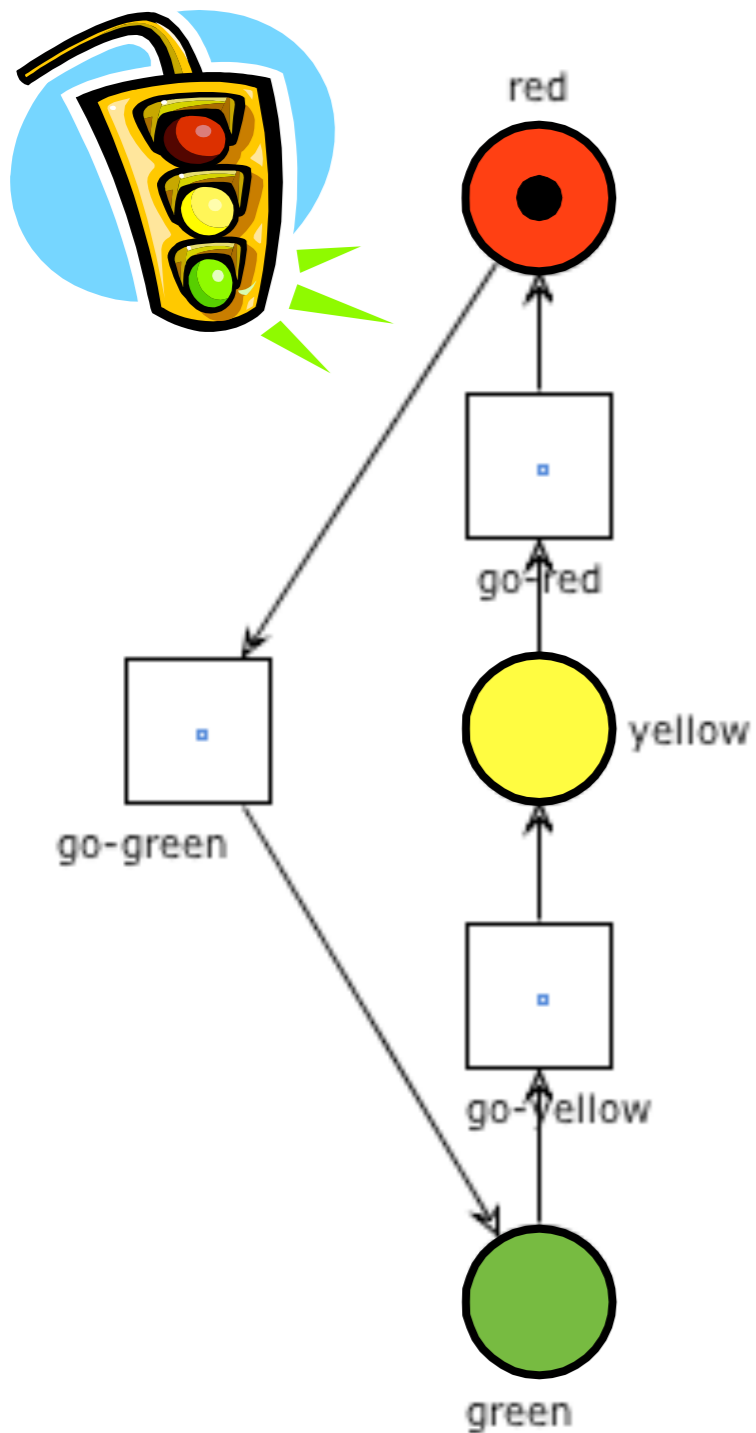
Example: traffic light



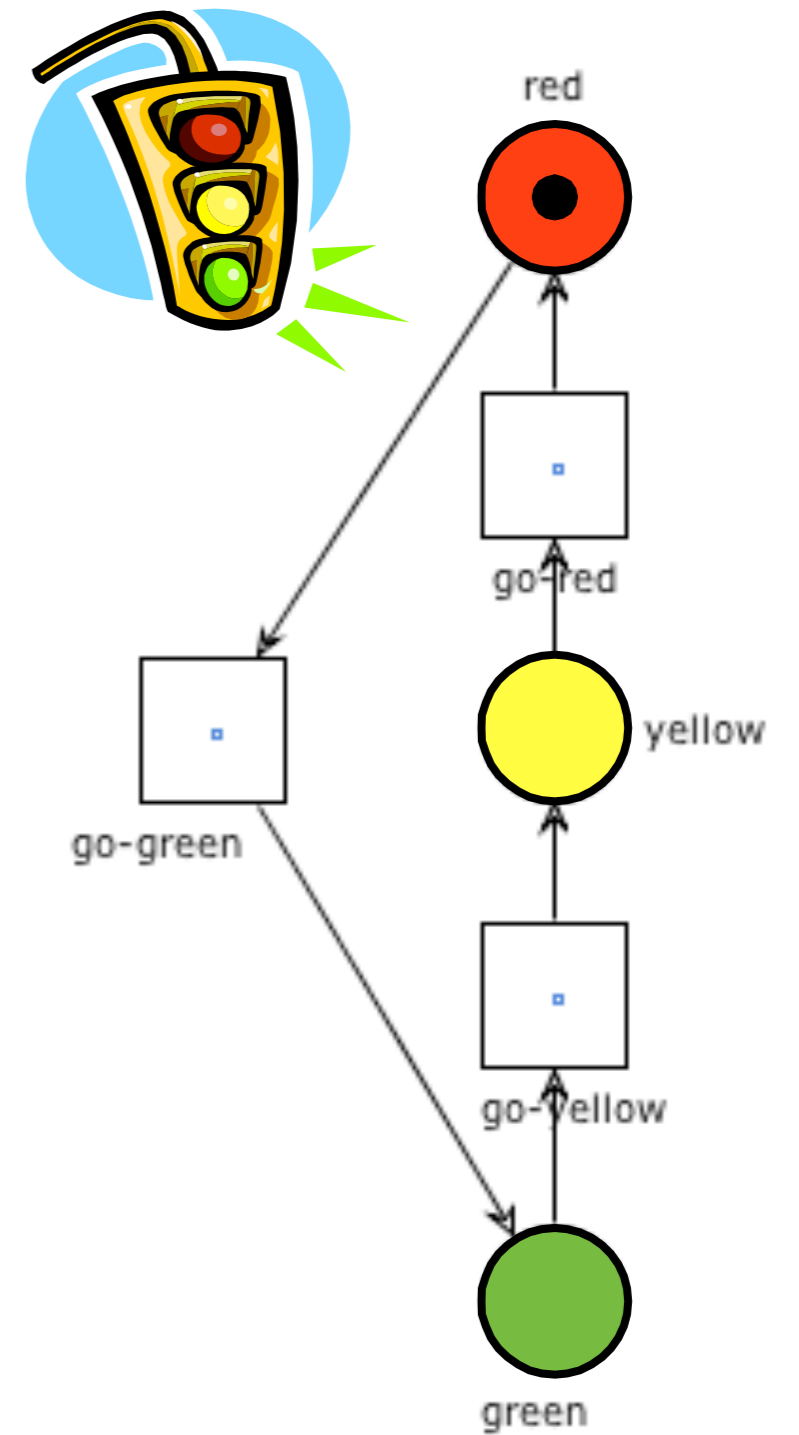
Example: traffic light



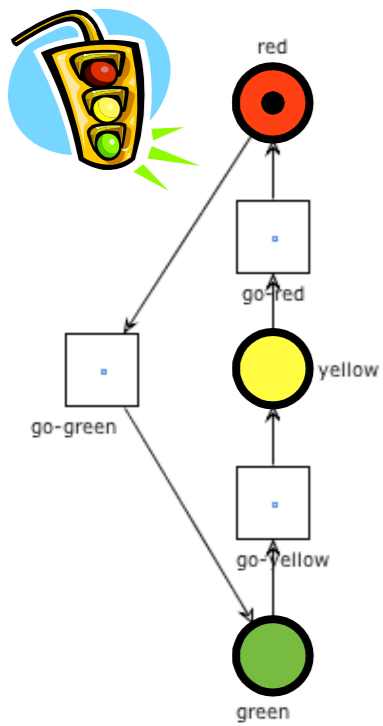
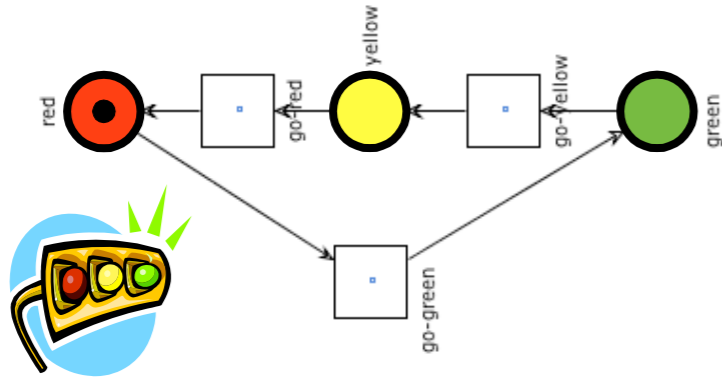
Example: two traffic lights



red + red'



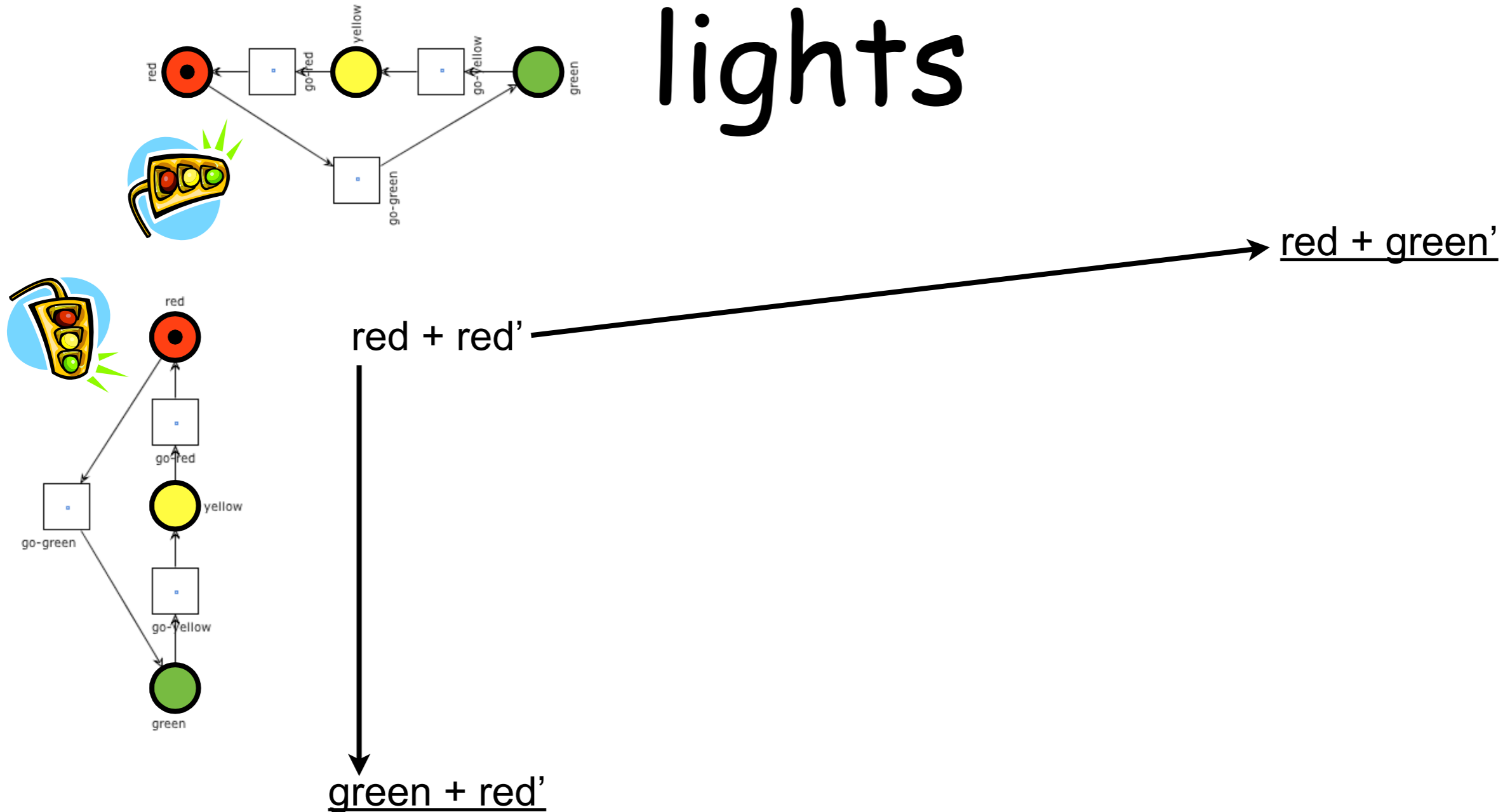
Example: two traffic lights



red + red'

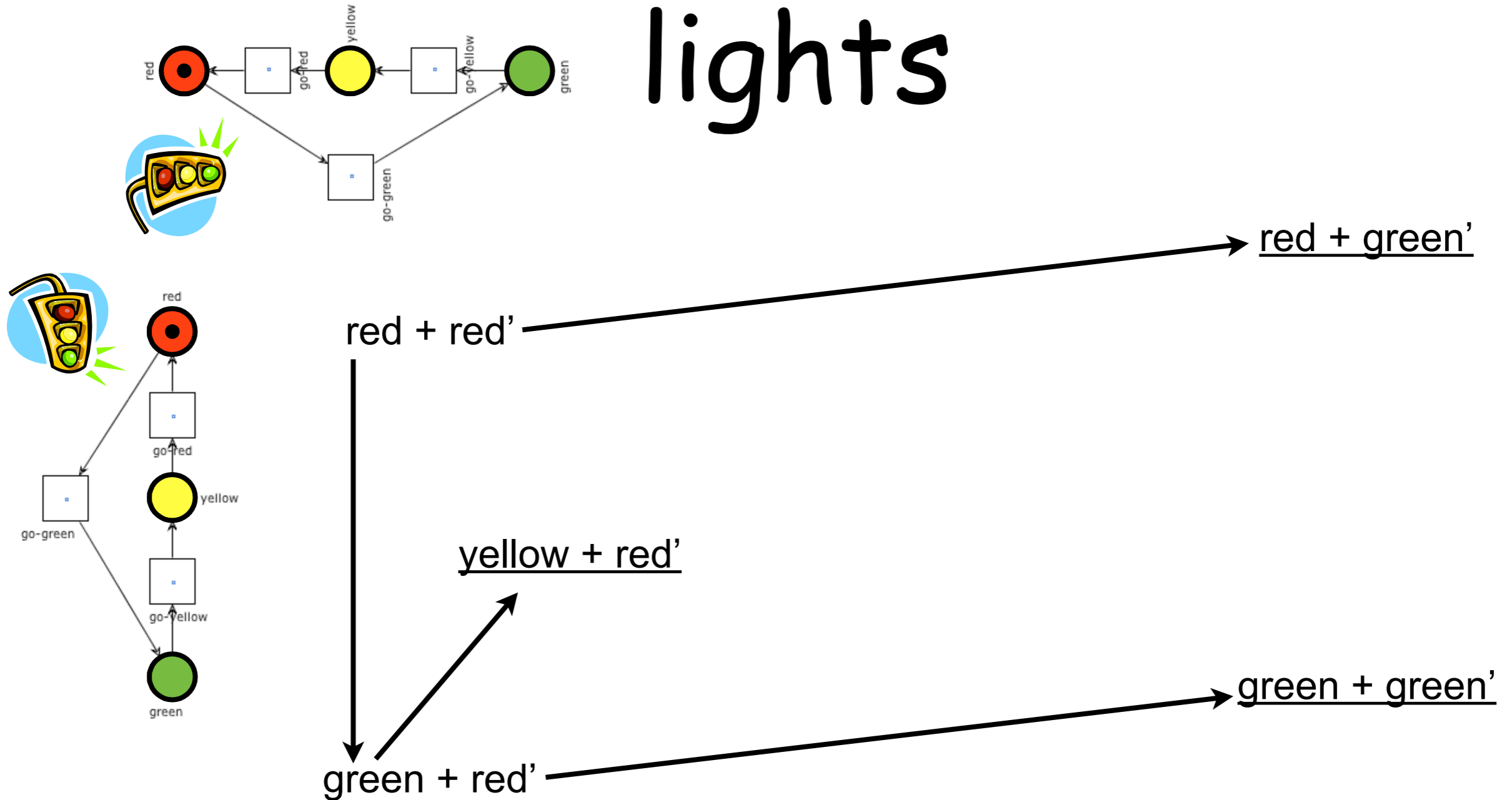
(we omit arc labels
for readability issues)

Example: two traffic lights



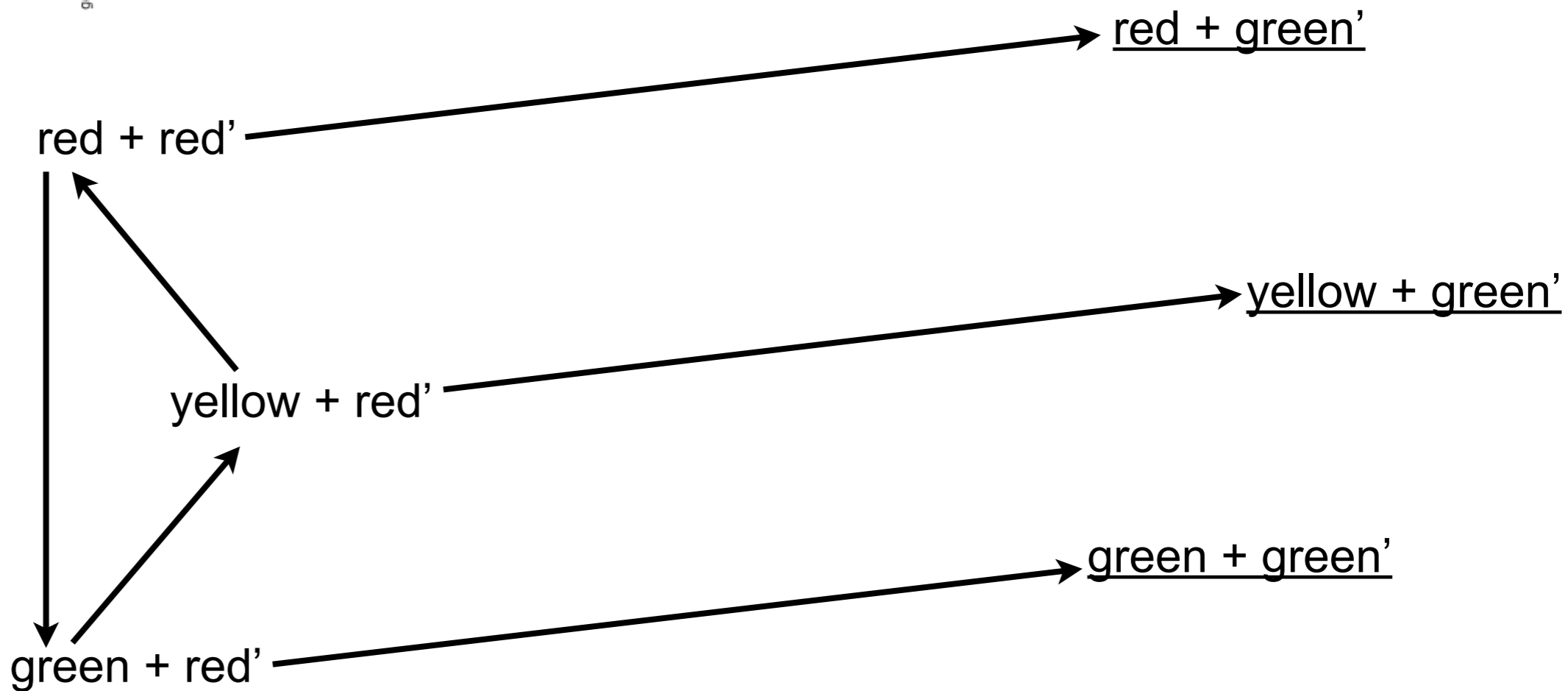
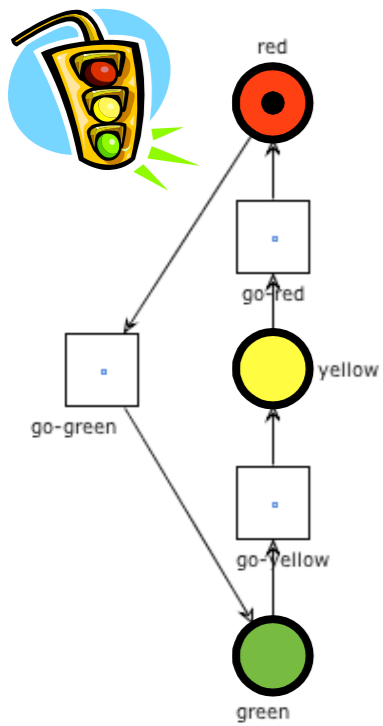
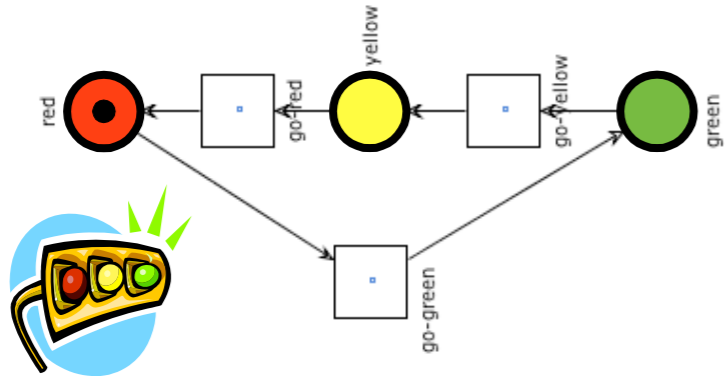
(we omit arc labels
for readability issues)

Example: two traffic lights



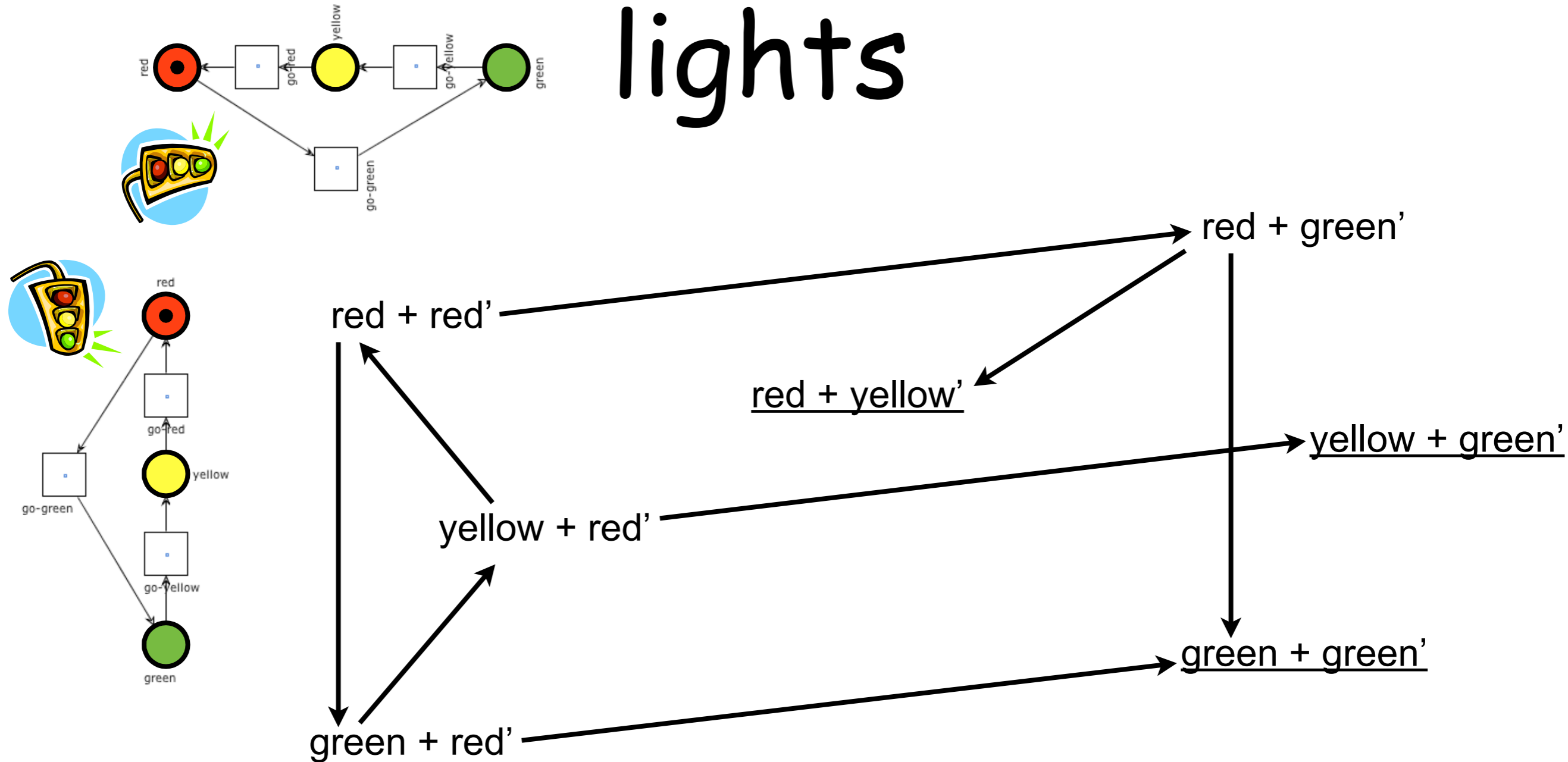
(we omit arc labels for readability issues)

Example: two traffic lights



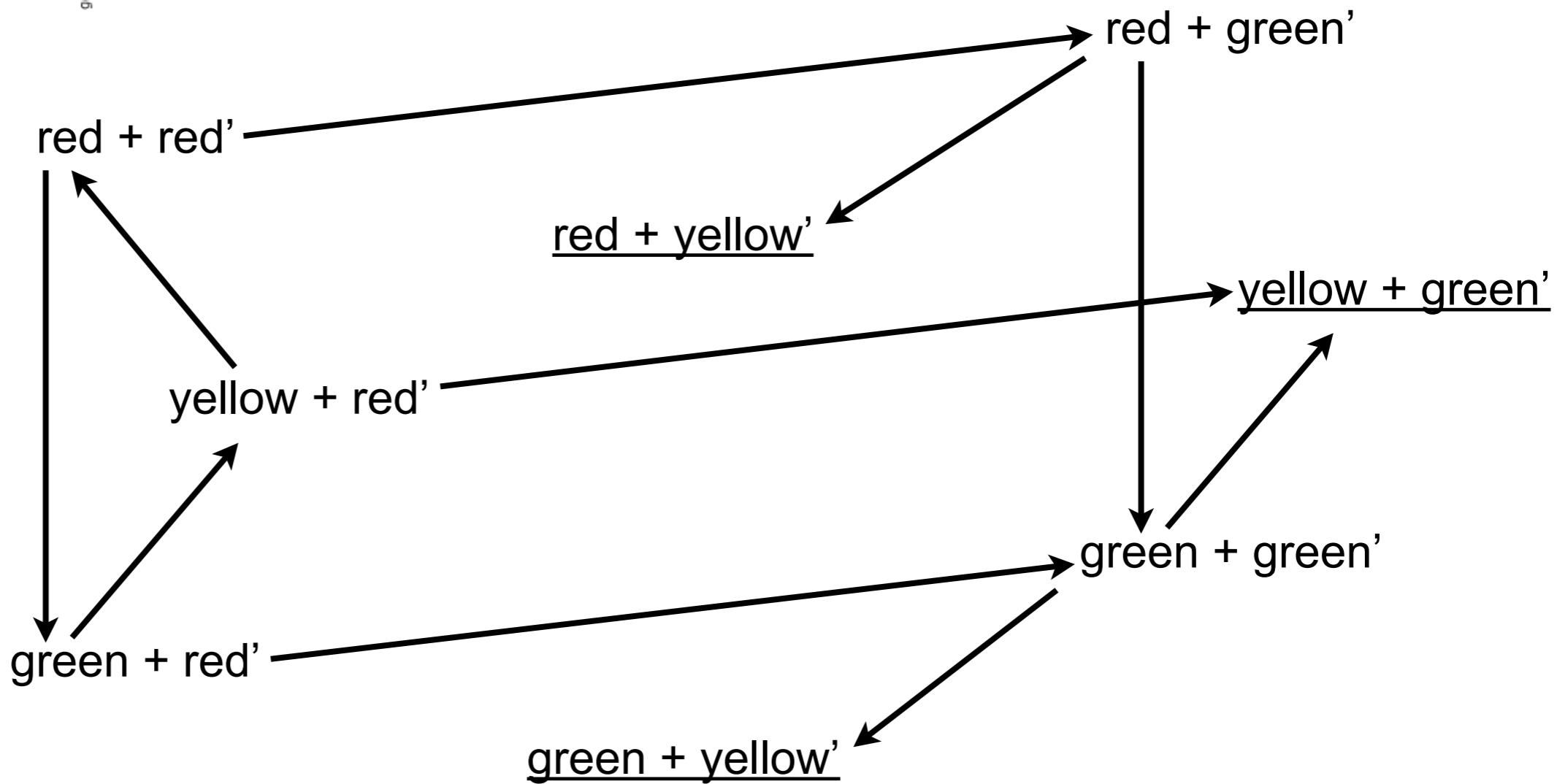
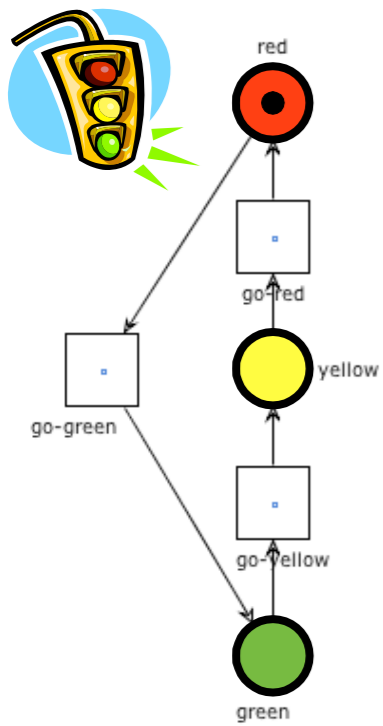
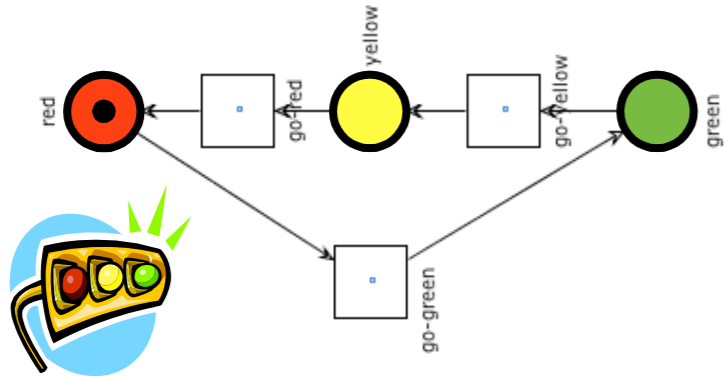
(we omit arc labels for readability issues)

Example: two traffic lights



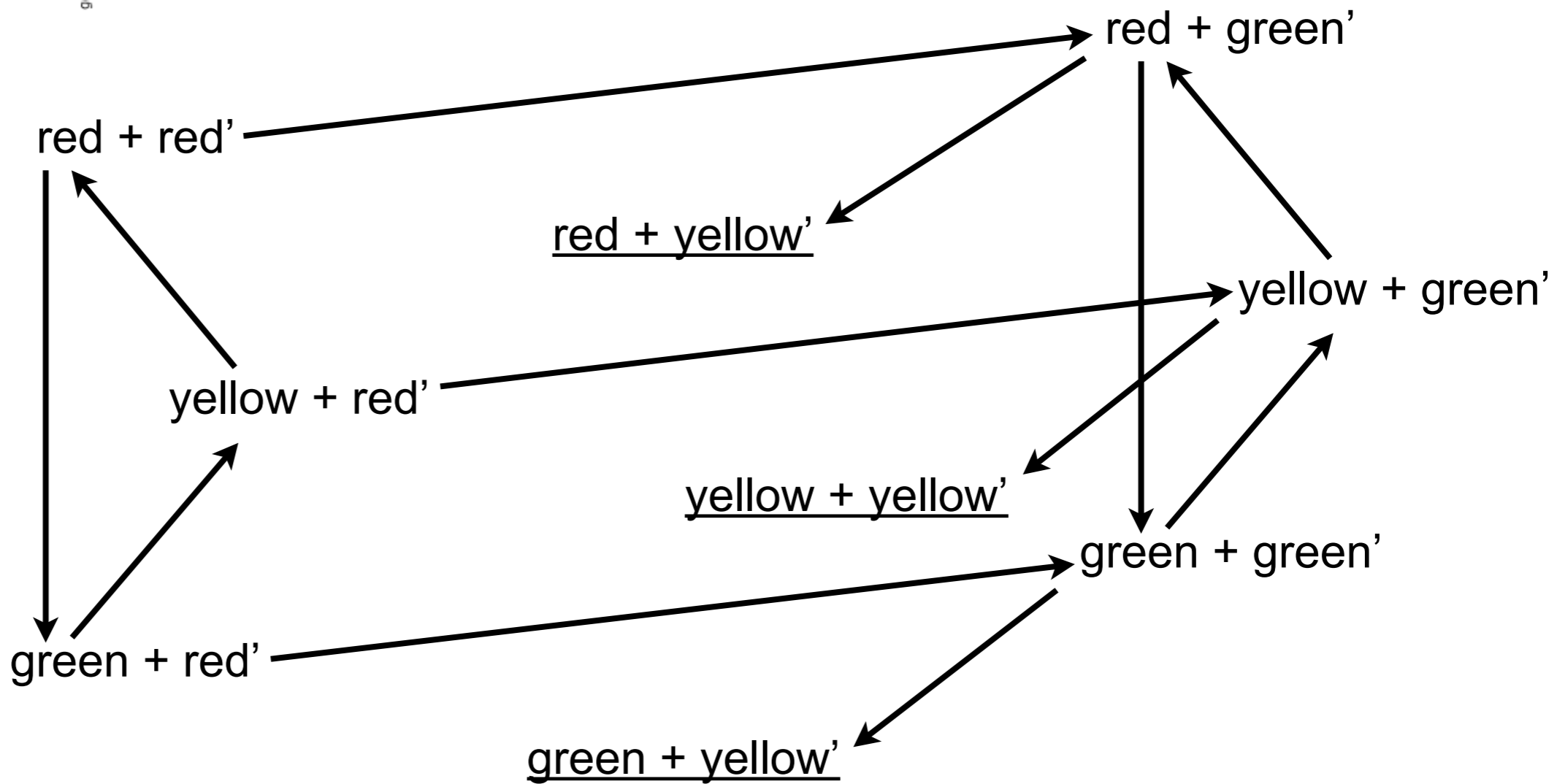
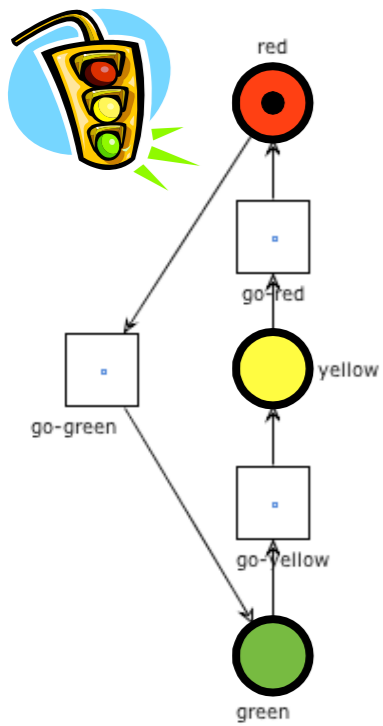
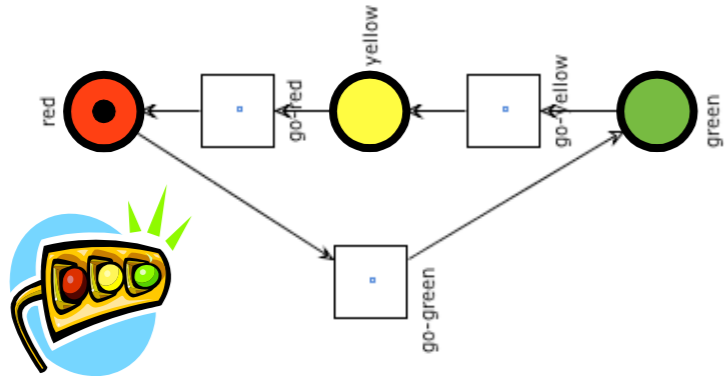
(we omit arc labels for readability issues)

Example: two traffic lights



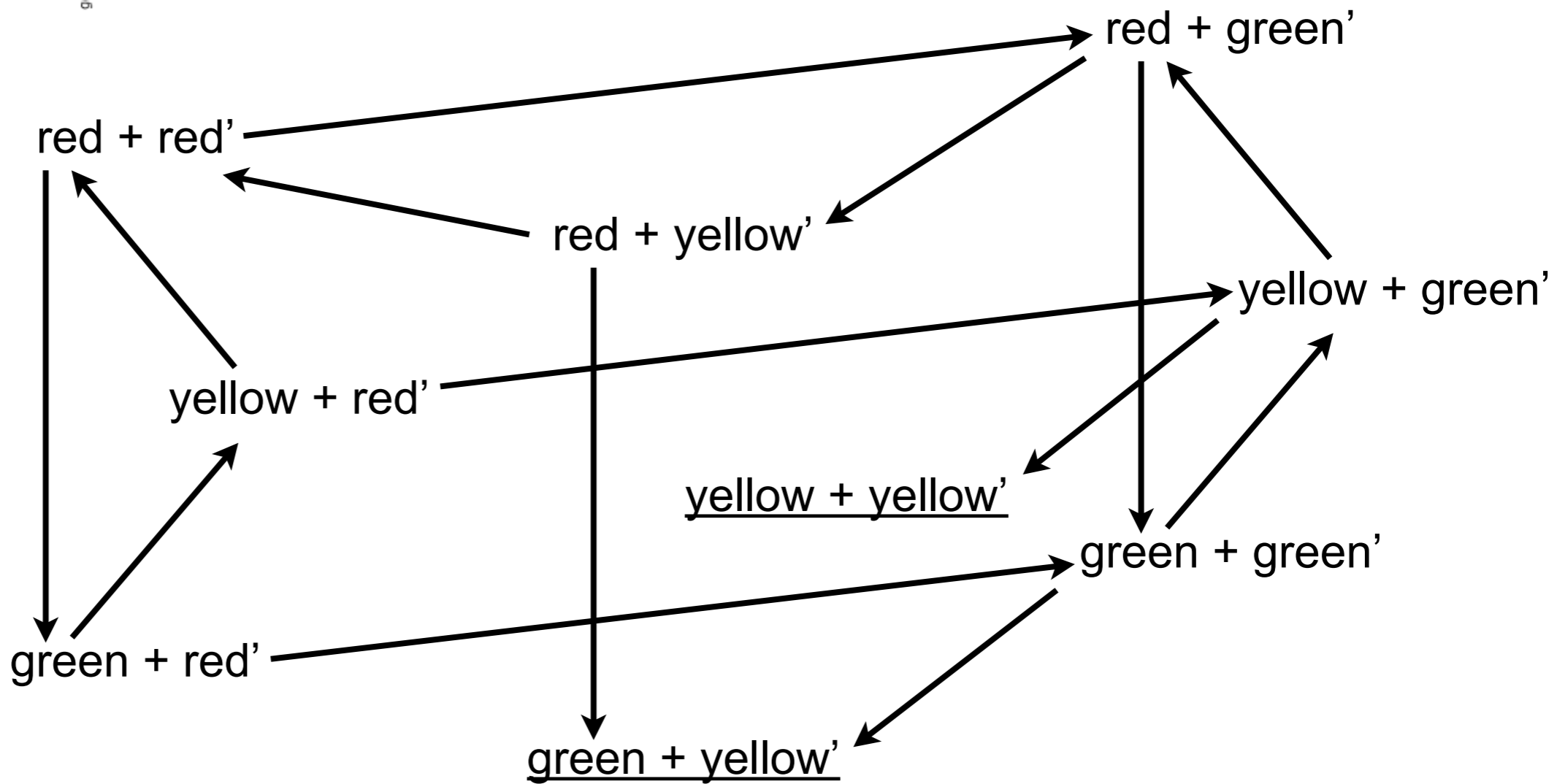
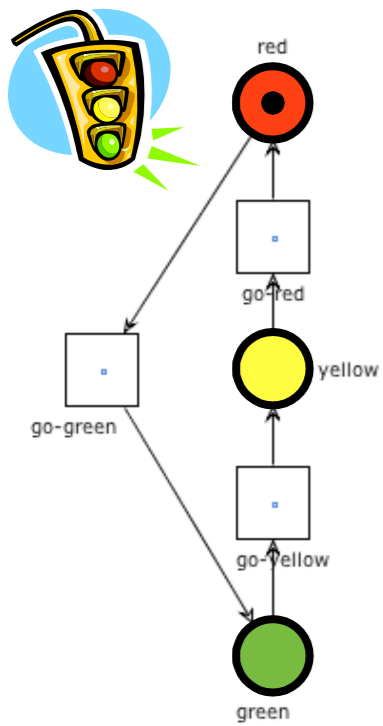
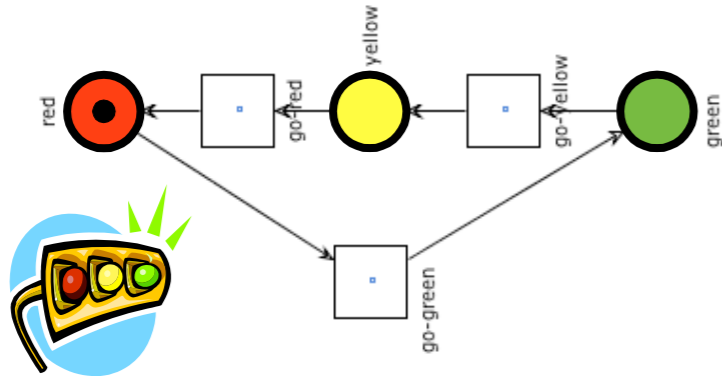
(we omit arc labels for readability issues)

Example: two traffic lights



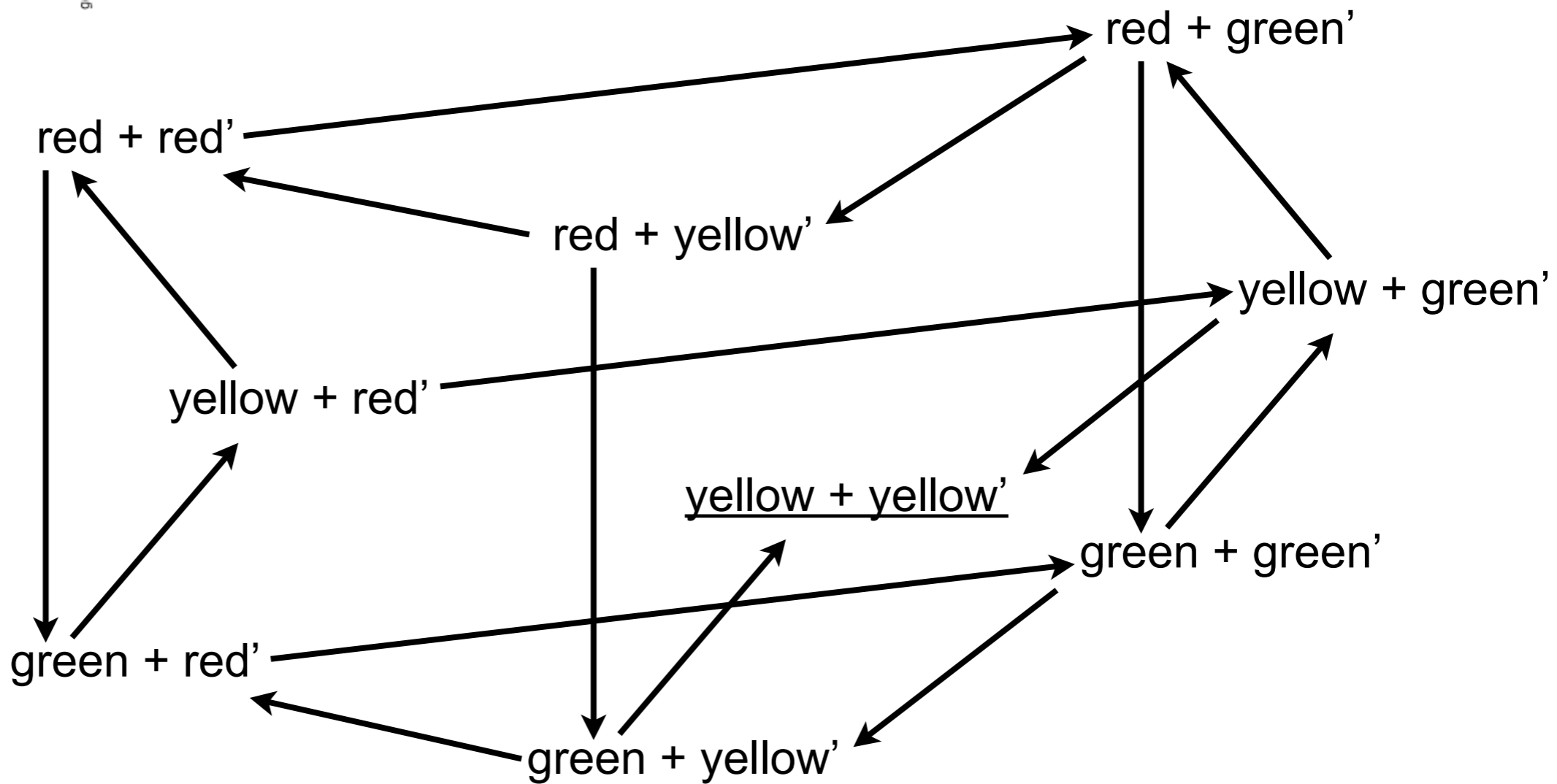
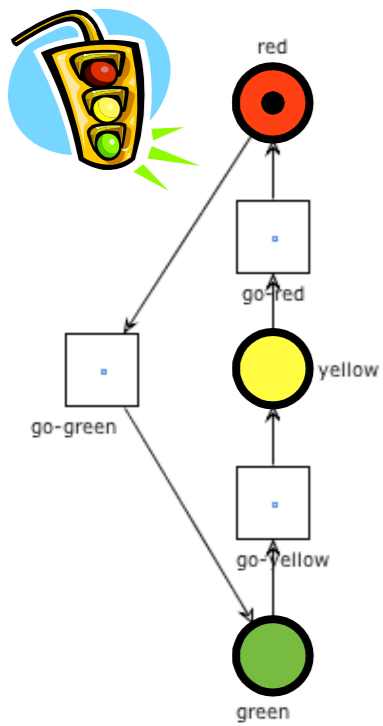
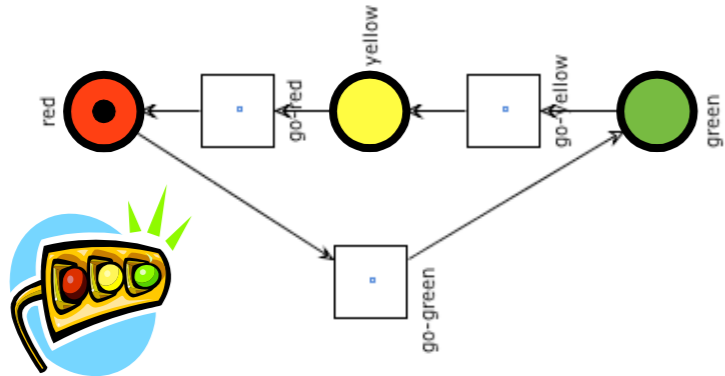
(we omit arc labels for readability issues)

Example: two traffic lights



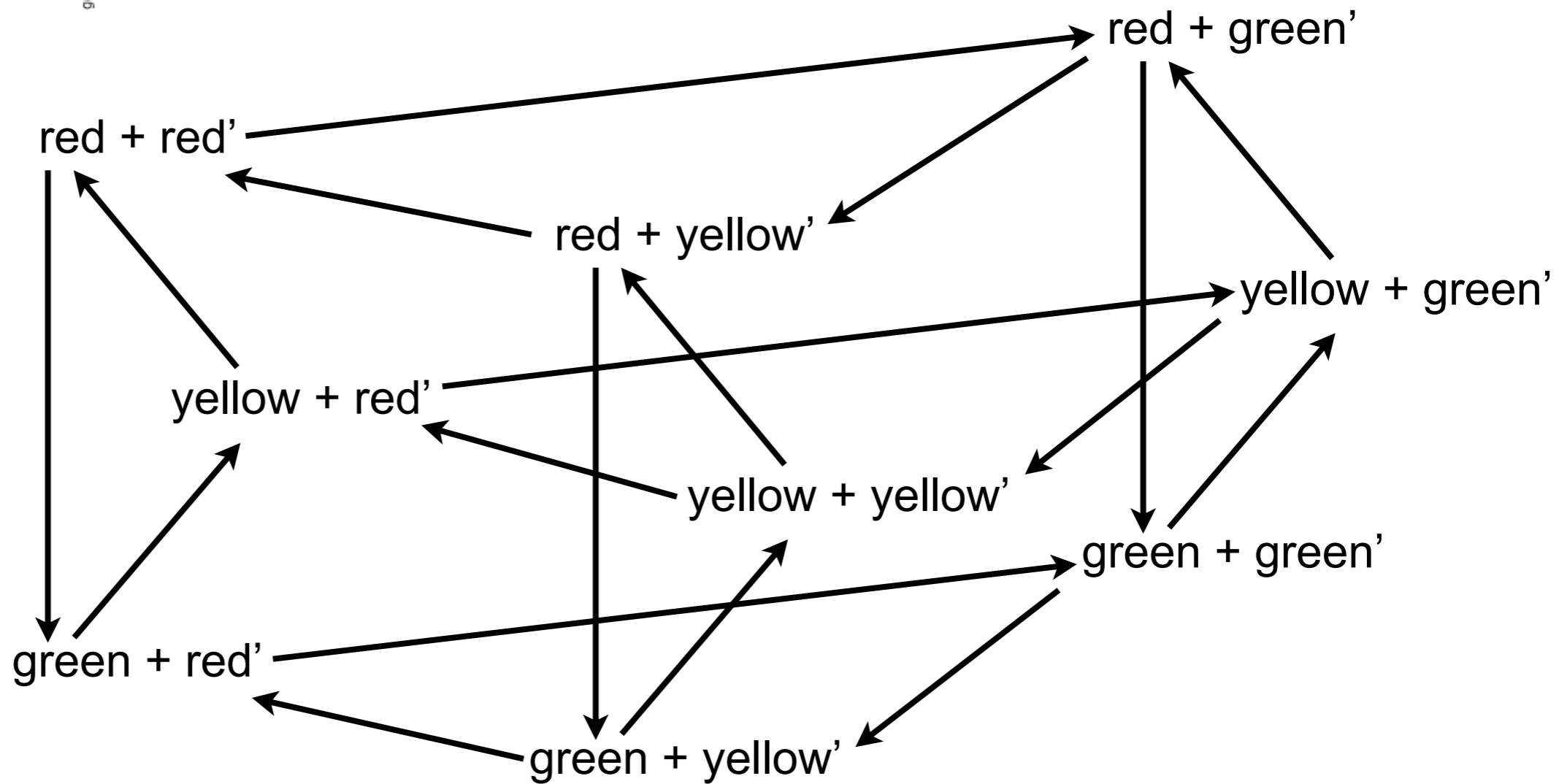
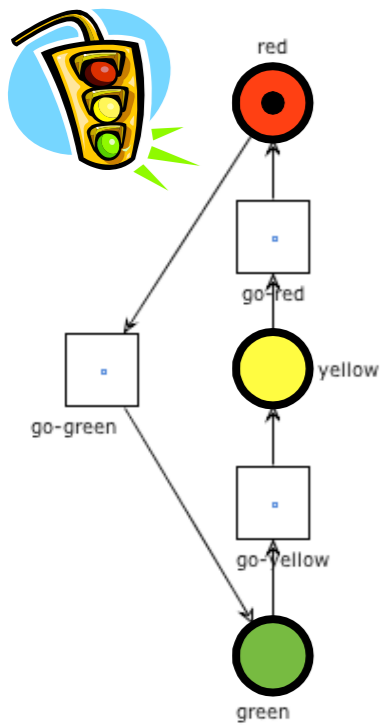
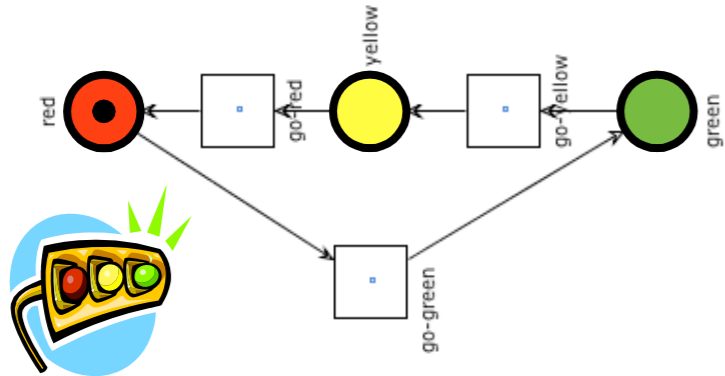
(we omit arc labels for readability issues)

Example: two traffic lights



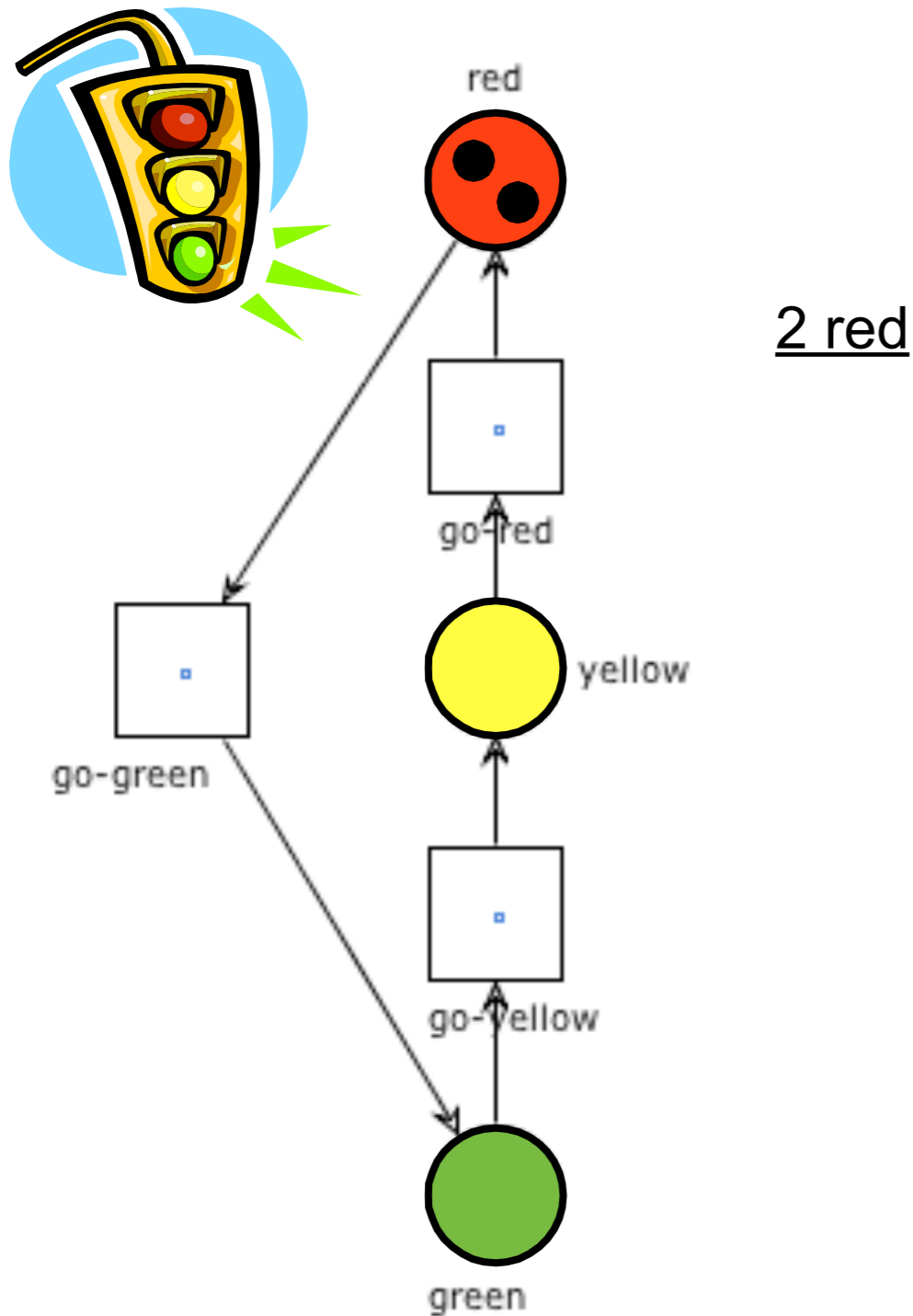
(we omit arc labels for readability issues)

Example: two traffic lights

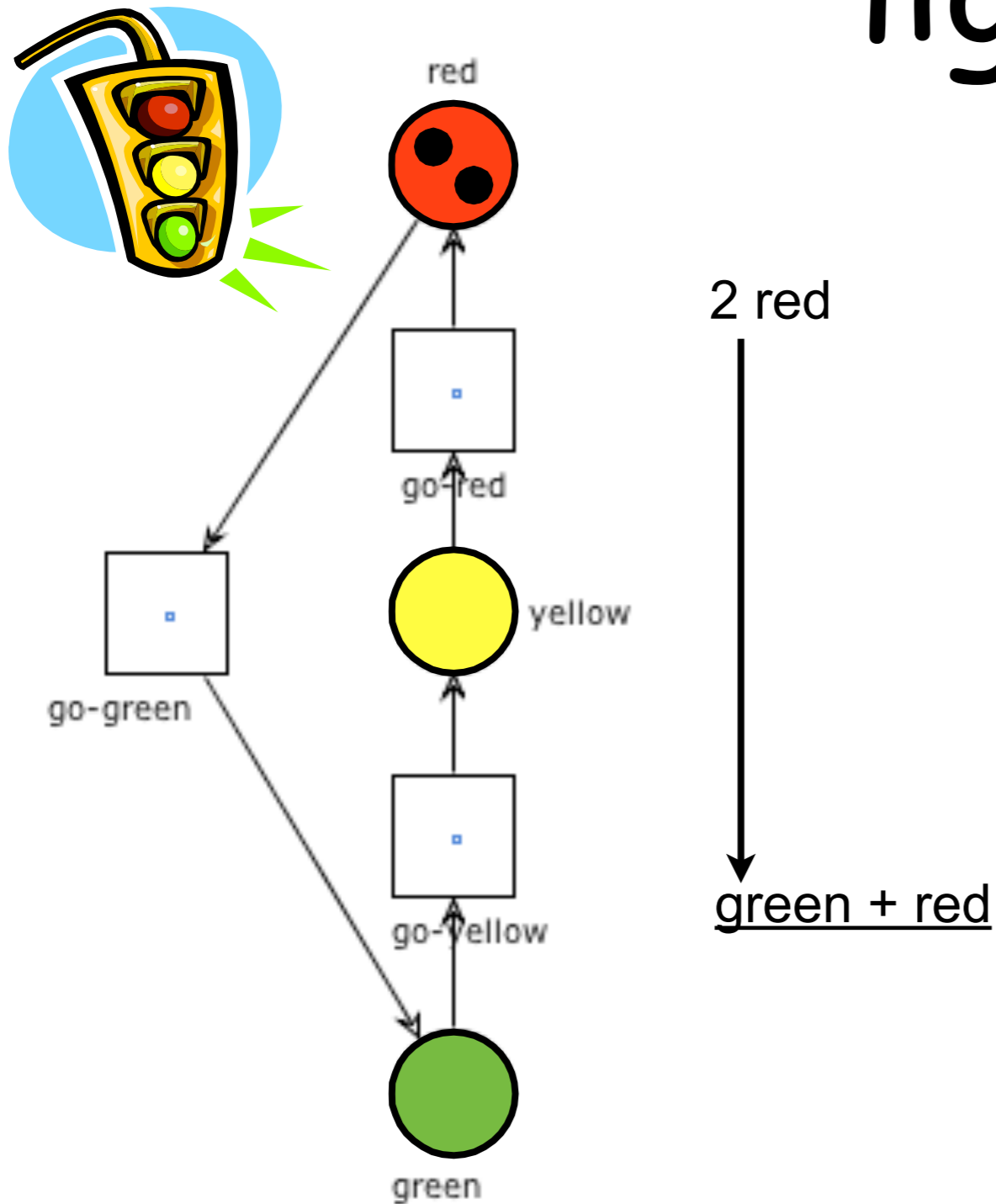


(we omit arc labels for readability issues)

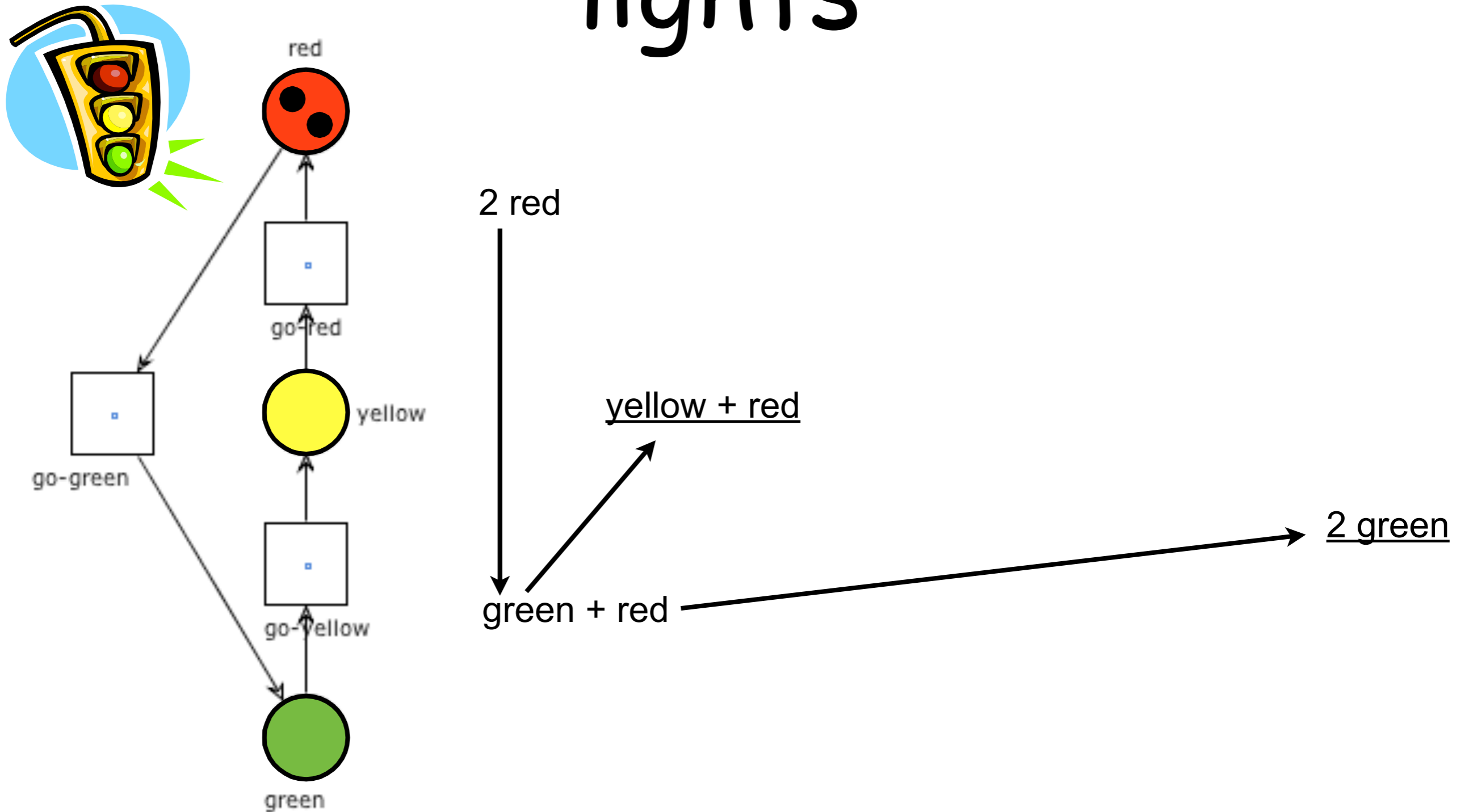
Example: two traffic lights



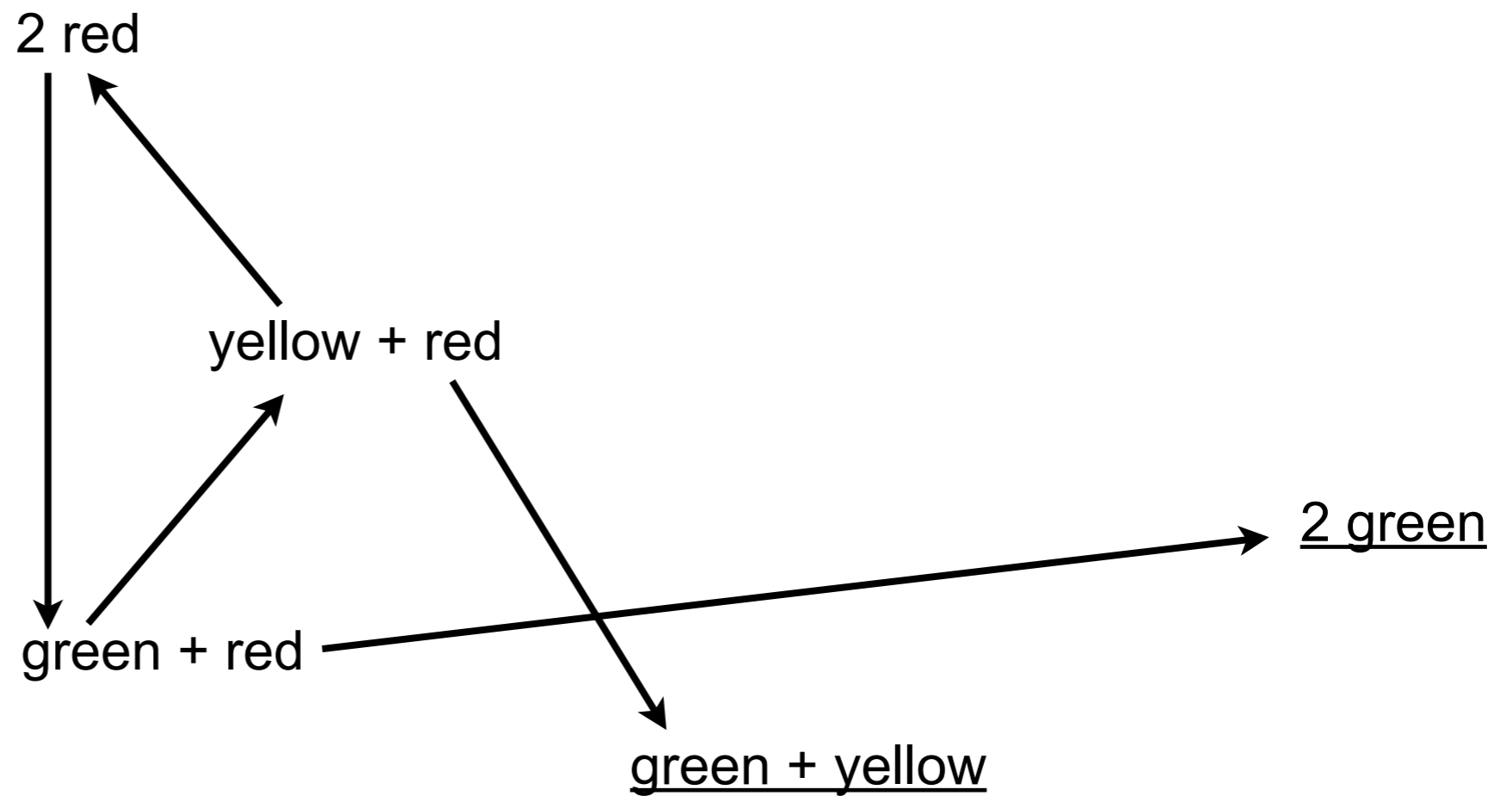
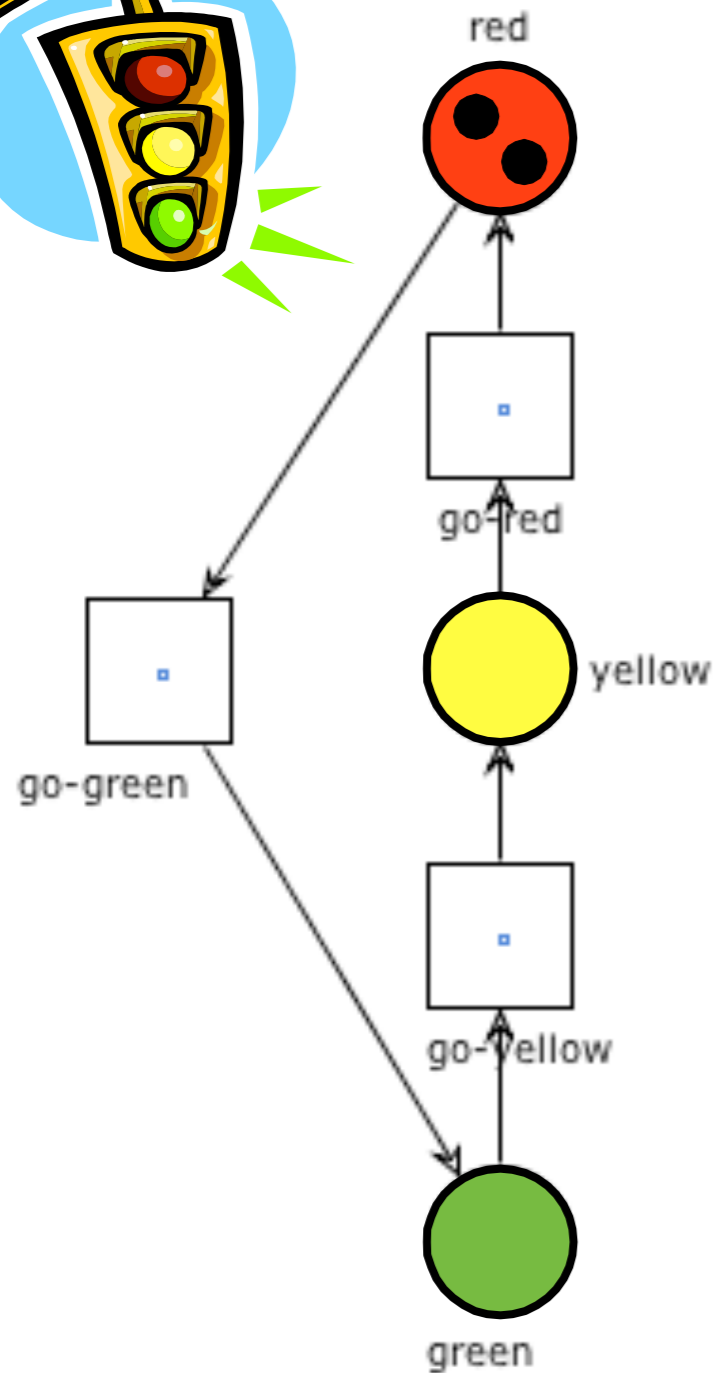
Example: two traffic lights



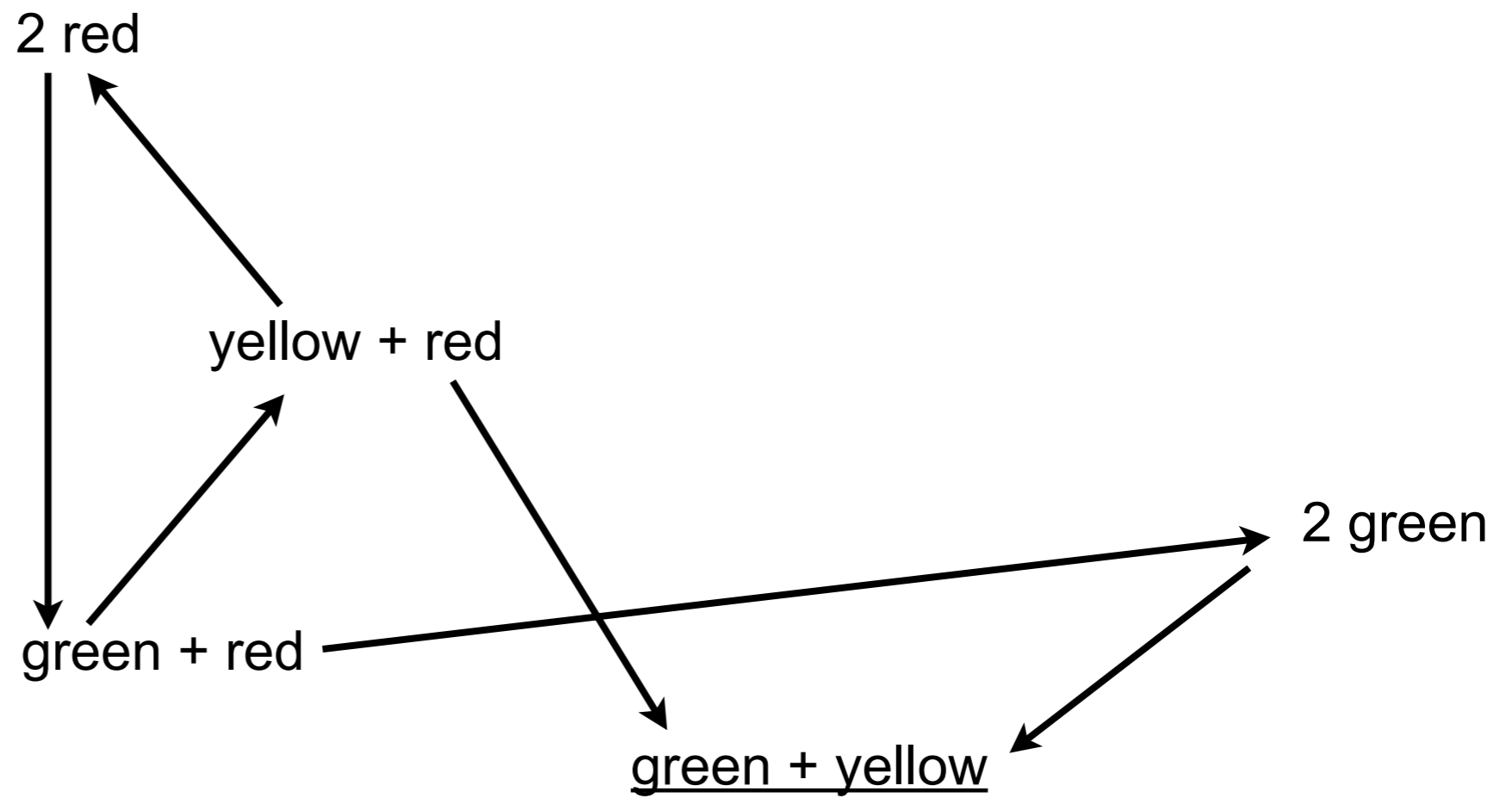
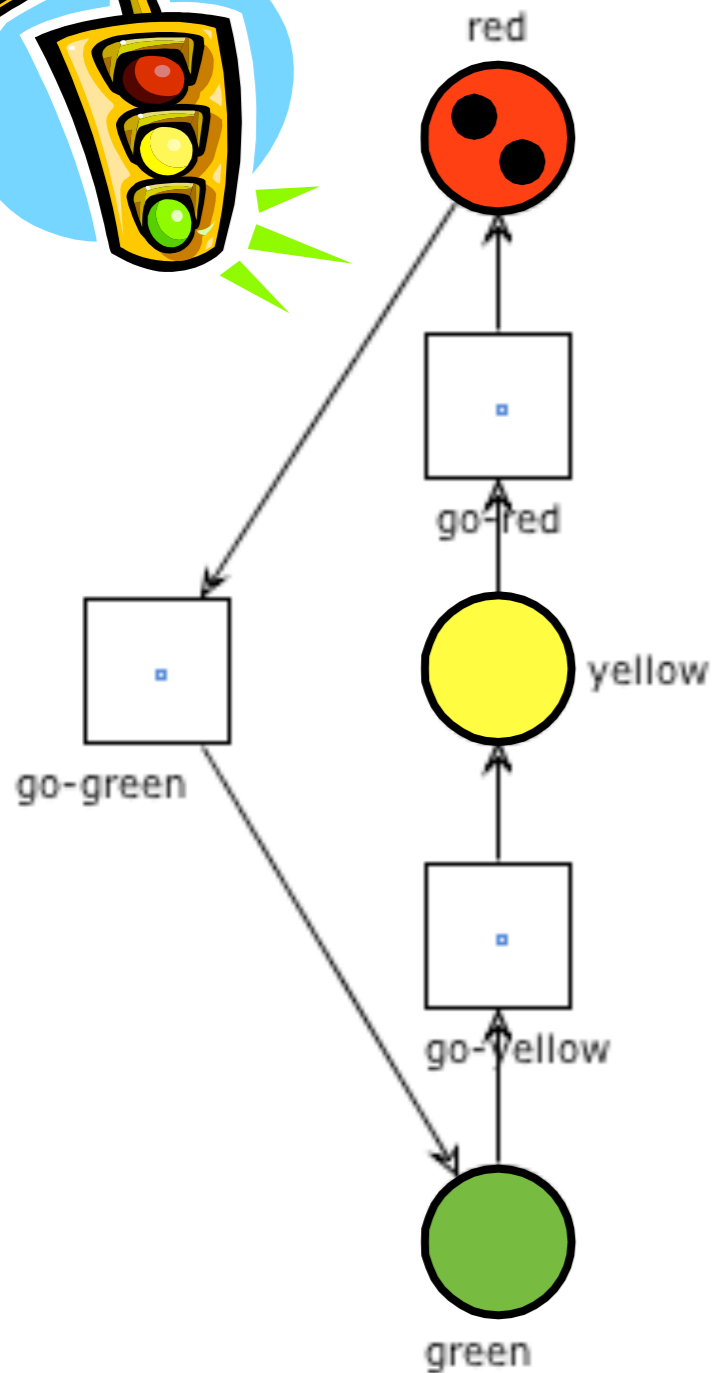
Example: two traffic lights



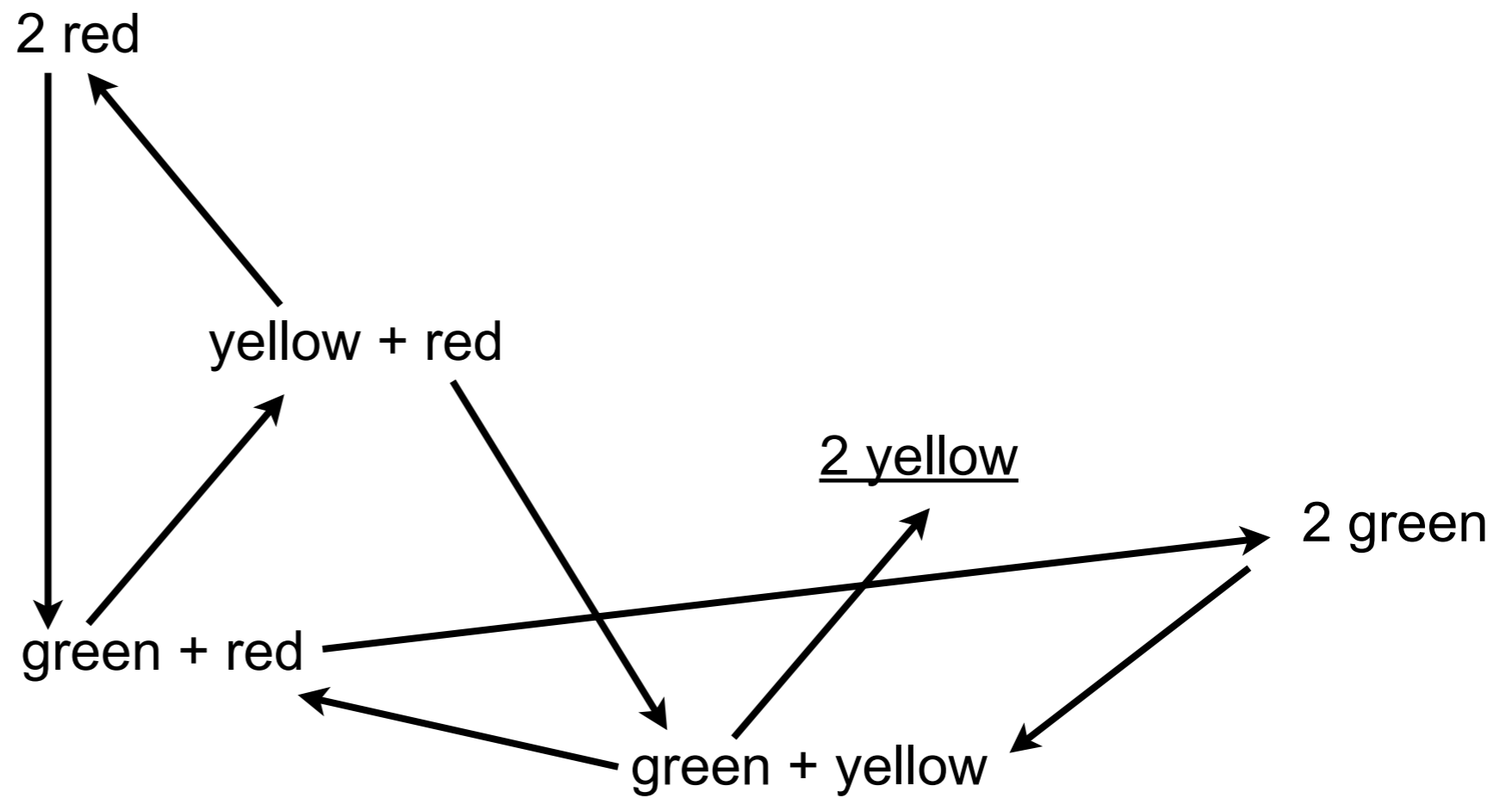
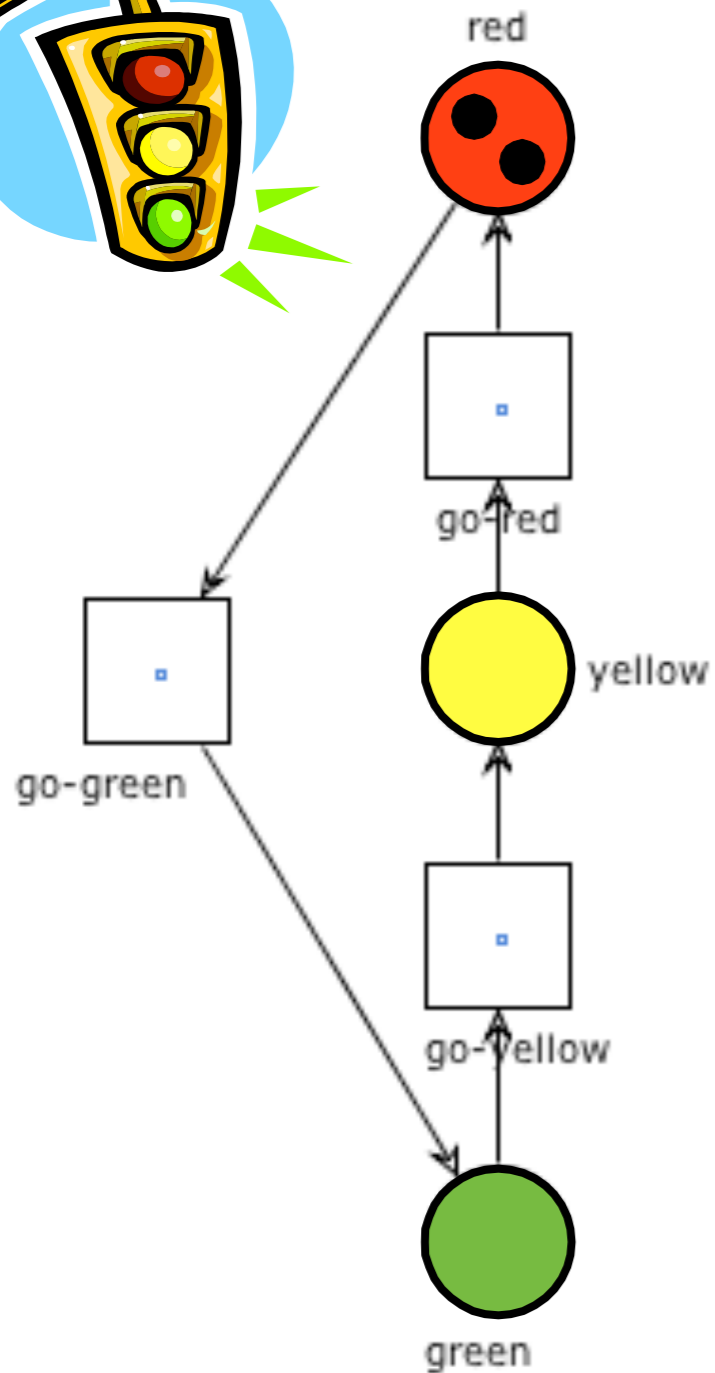
Example: two traffic lights



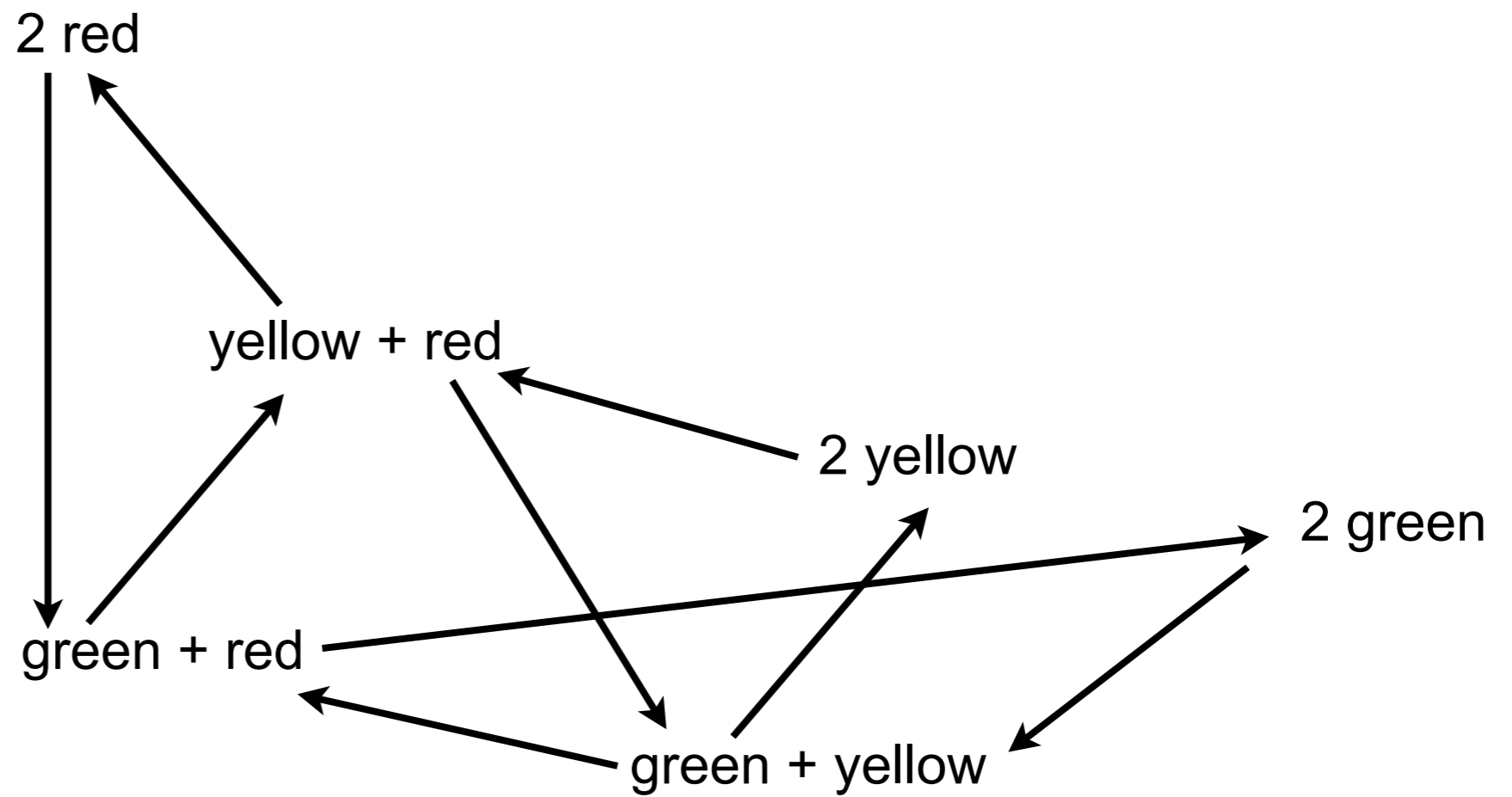
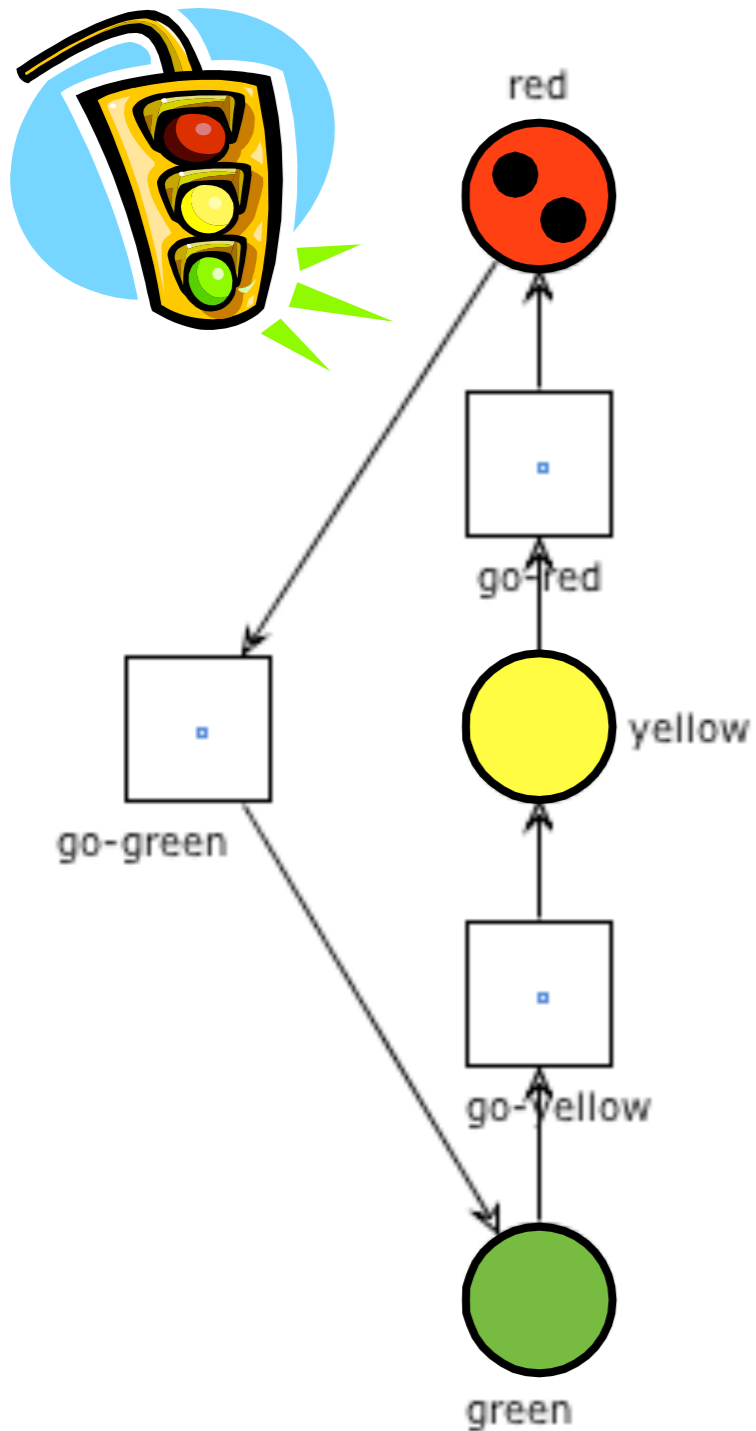
Example: two traffic lights



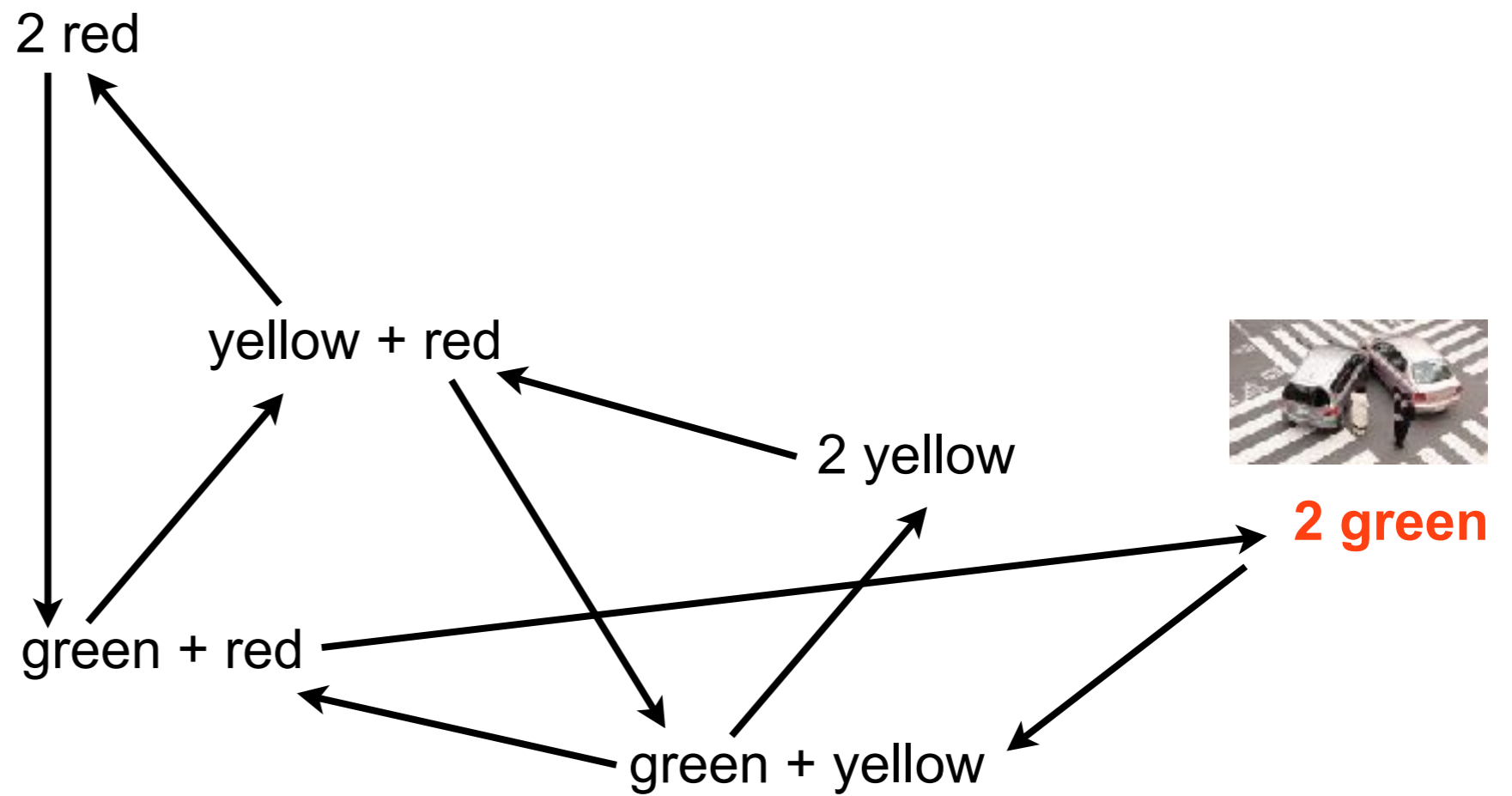
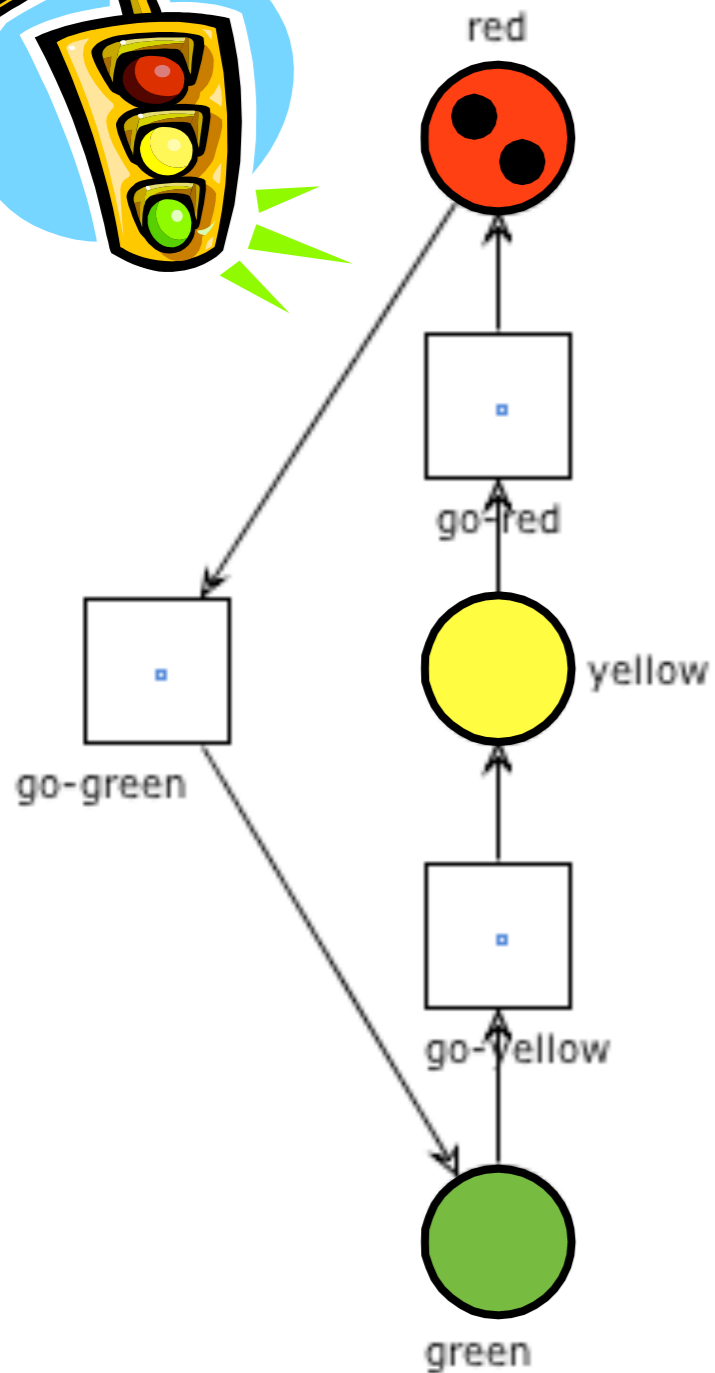
Example: two traffic lights



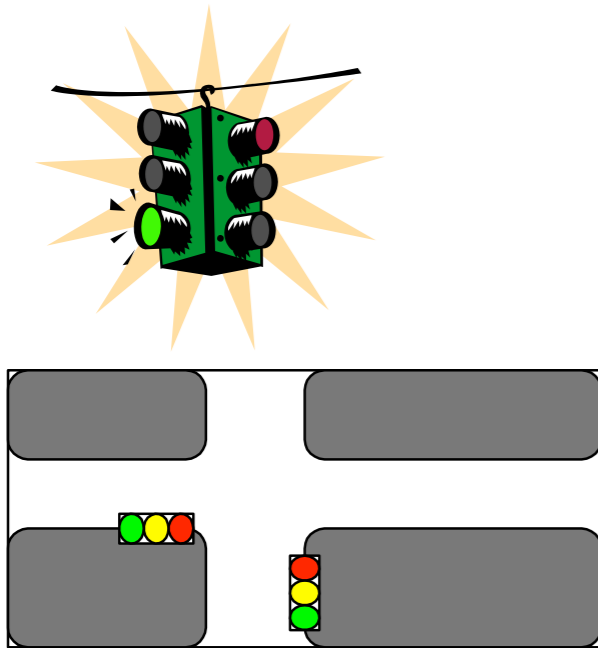
Example: two traffic lights



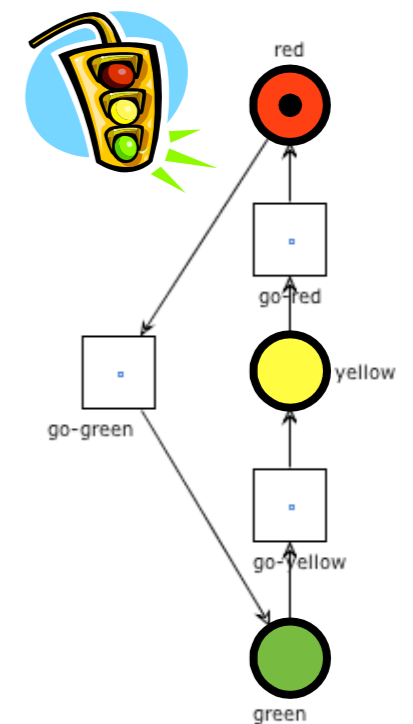
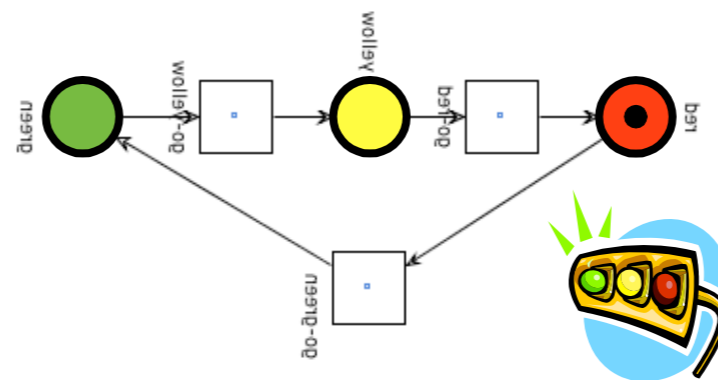
Example: two traffic lights



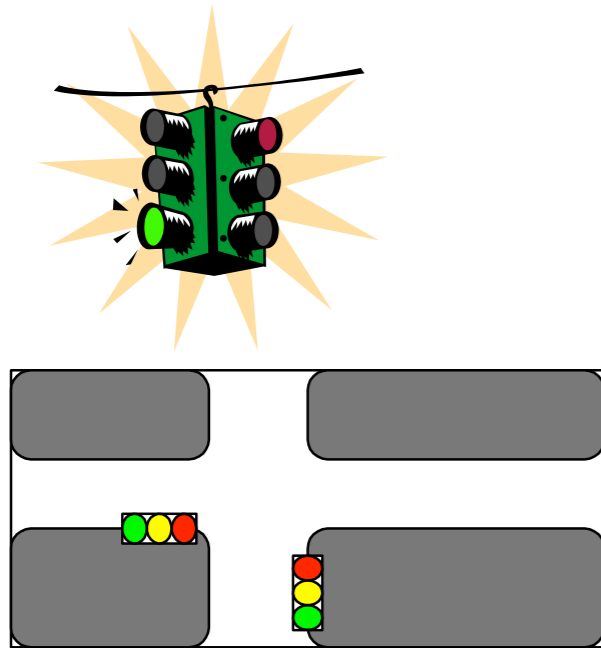
Question time



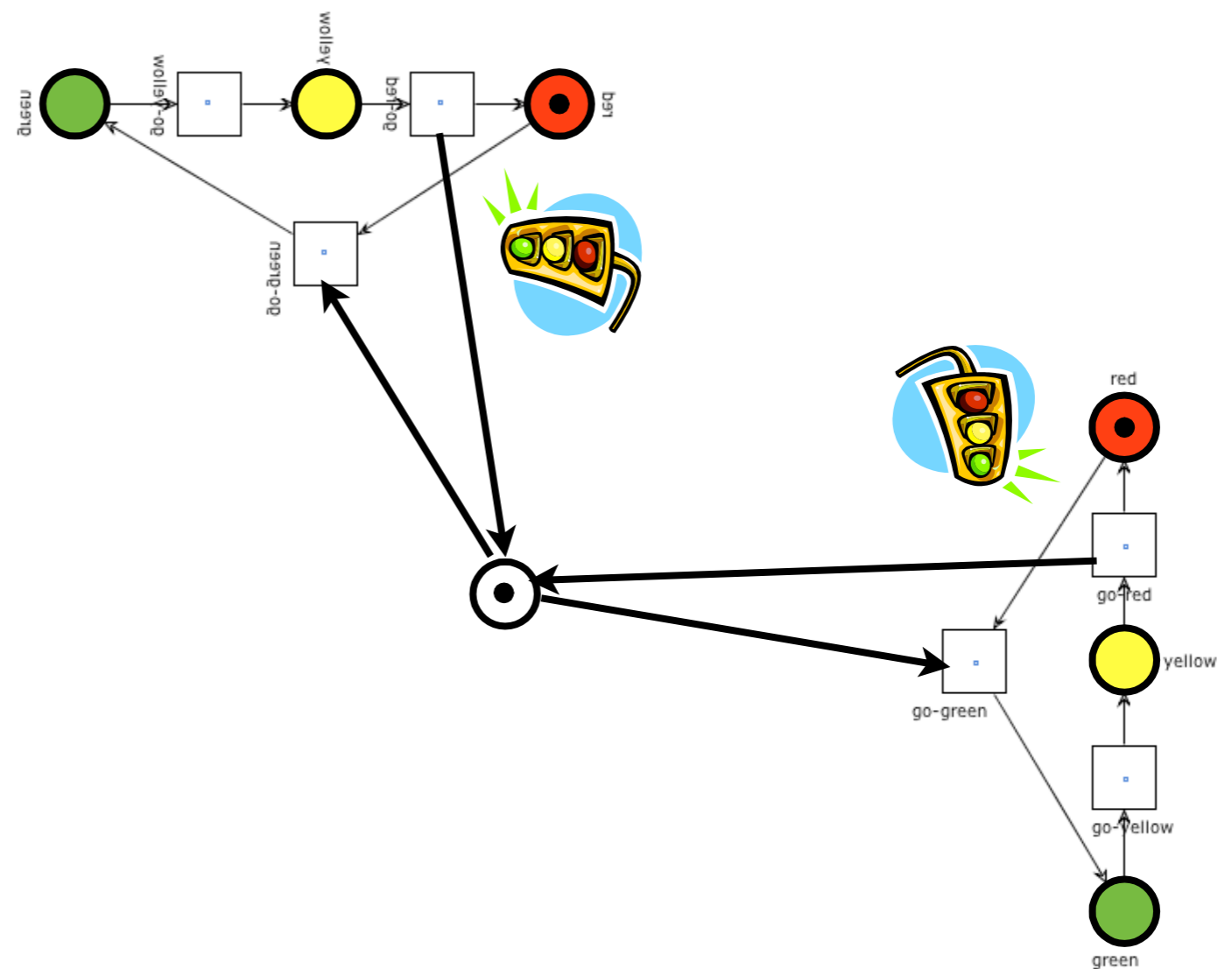
Complete the net in such a way that the two lights can never be green at the same time



Question time



Complete the net in such a way that the two lights can never be green at the same time



Exercises

Draw the reachability graph of the last net

Modify the net so to guarantee that green alternate on the two traffic lights and then draw the reachability graph

Play the “token games” on the above nets using Workflow Petri net Designer:
<http://www.woped.org>

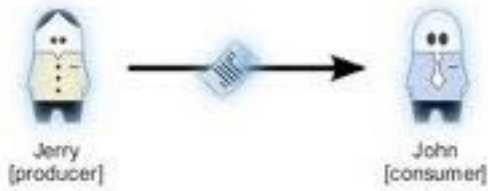
Exercise:

German traffic lights

German traffic lights have an extra phase: traffic lights turn not suddenly from red to green but give a red light together with a yellow light before turning to green.

Identify the possible states and model the transition system that lists all possible states and state transitions.

Provide a Petri net that is able to behave exactly like a German traffic light. There should be three places indicating the state of each light and make sure that the Petri net does not allow state transitions which should not be possible.



Exercise:

Producer and consumer

Model a process with one **producer** and one **consumer**:

Each one is either **busy** or **free**.

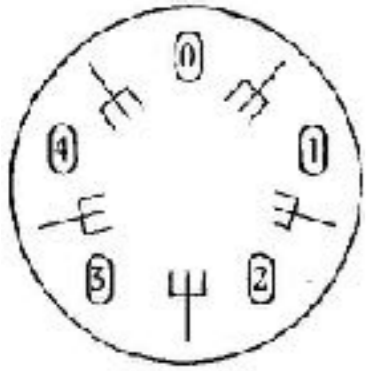
Each one alternates between these two states

After every production cycle the producer puts a product in a **buffer** and the consumer consumes one product from this buffer (when available) per cycle.

Draw the reachability graph

How to model 4 producers and 3 consumers connected through a single buffer?

How to limit the size of the buffer to 2 items?



Exercise:

Dining philosophers

The problem is originally due to E.W. Dijkstra (and soon elaborated by T. Hoare) as an examination question on a synchronization problem where five computers competed for access to five shared tape drive peripherals.

It can be used to illustrate several important concepts in concurrency (mutual exclusion, deadlock, starvation)

Exercise: Dining philosophers

The life of a philosopher consists of an alternation of **thinking** and **eating**

Five philosophers are living in a house where a table is laid for them, each philosopher having his own place at the table

Their only problem (besides those of philosophy) is that the dish served is a very difficult kind of spaghetti, that has to be eaten with two forks. There are **two forks next to each plate**, so that presents no difficulty: as a consequence, however, no two neighbours may be eating simultaneously.

Exercise: Dining philosophers

Design a net for representing the dining philosophers problem, then use WoPeD to compute the reachability graph

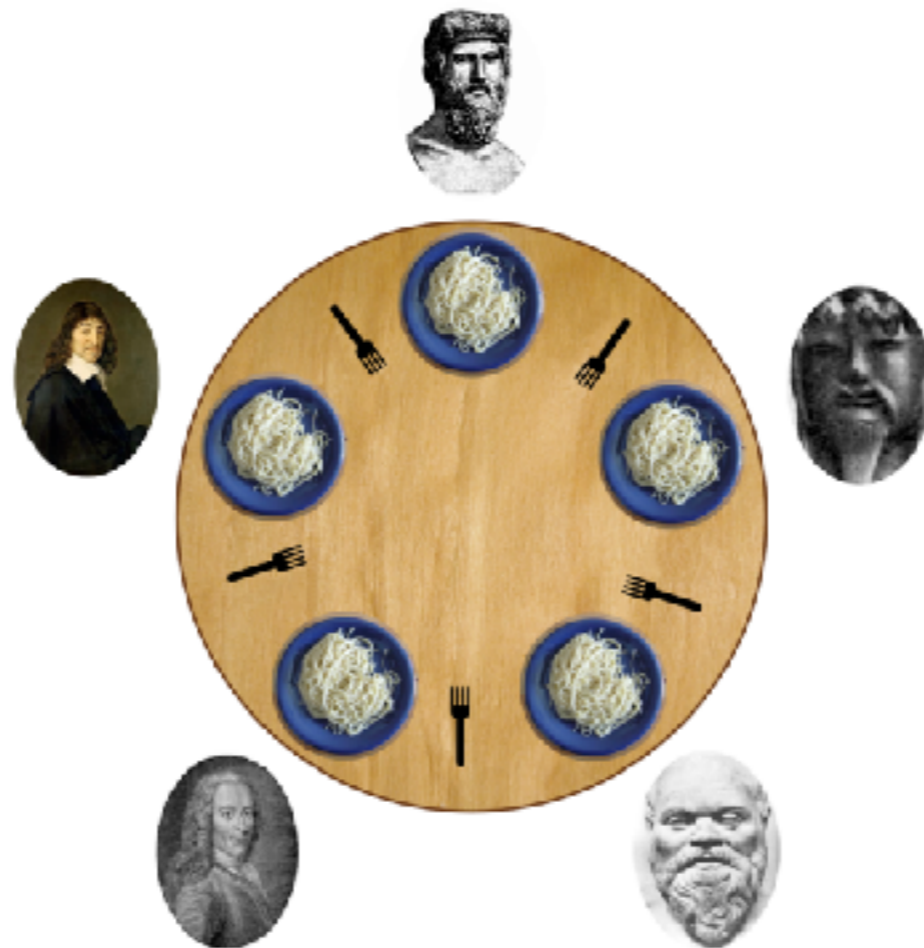


image taken from wikipedia
philosophers clockwise from top:
Plato, Konfuzius, Socrates,
Voltaire and Descartes

Exercise

Use a Petri net to model a circular railway system with **four stations** (st_1, st_2, st_3, st_4) and **one train**

At each station passengers may

"hop on" or **"hop off"**

(this is impossible when the train is moving)

The train has a **capacity of 50 persons**

(if the train is full no passenger can hop on,
if the train is empty no passenger can hop off)

What is the number of reachable states?