

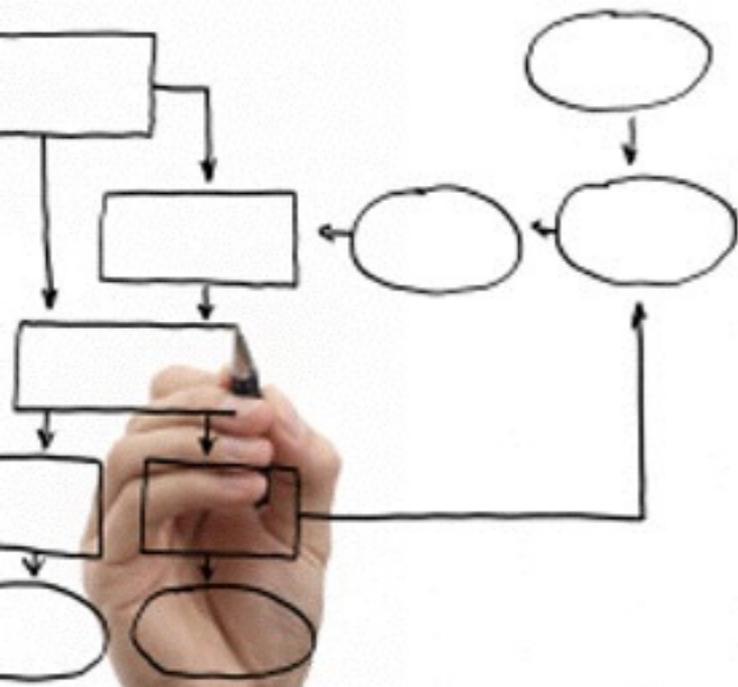
# Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni

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

08 - More on nets



# Object

To continue the overview of the basic concepts of  
Petri nets

Free Choice Nets (book, optional reading)

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

# Enabling $M[t \rangle$

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

# 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

# Some remarks

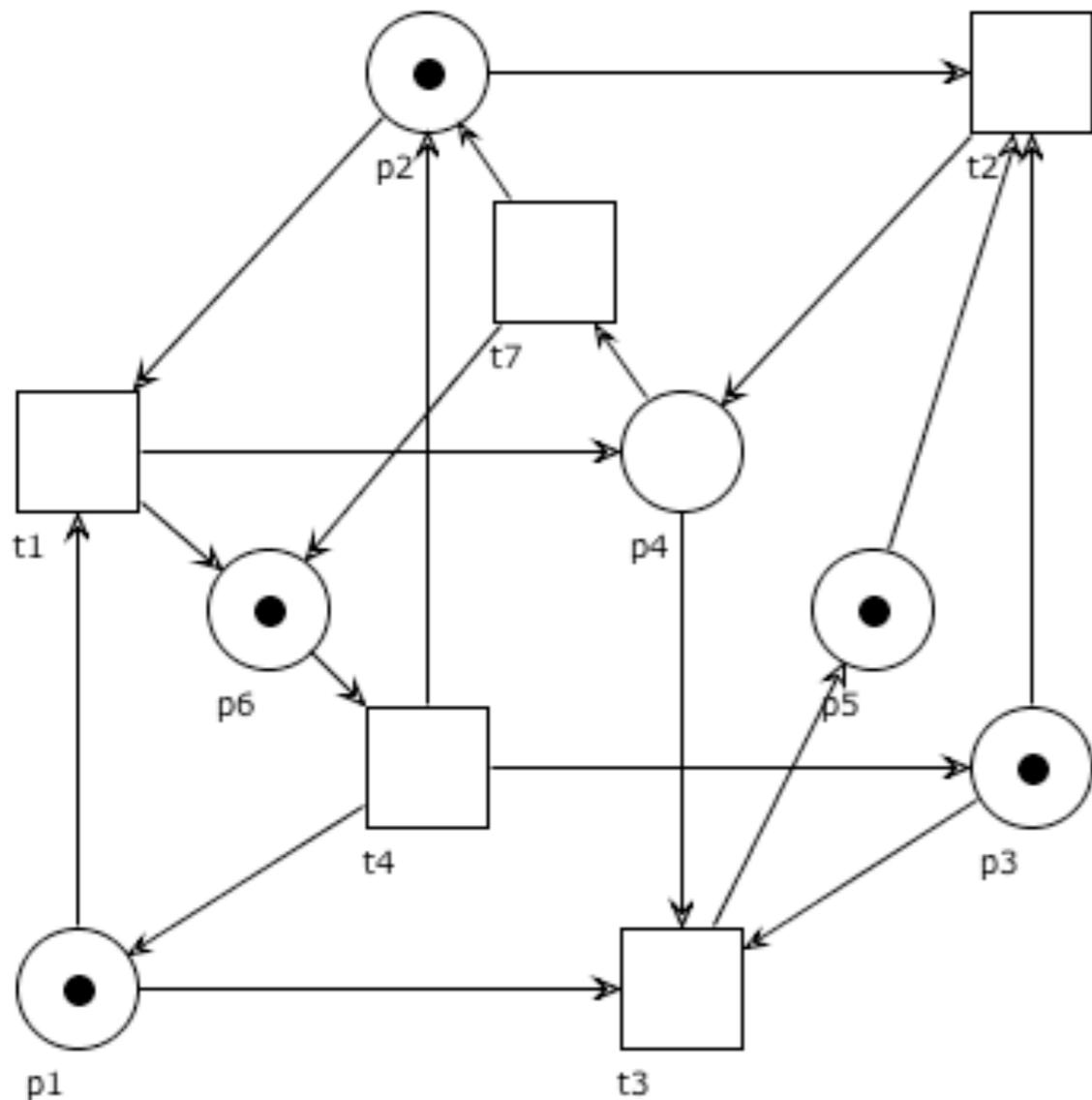
Firing is an atomic action

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)

# Question time

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

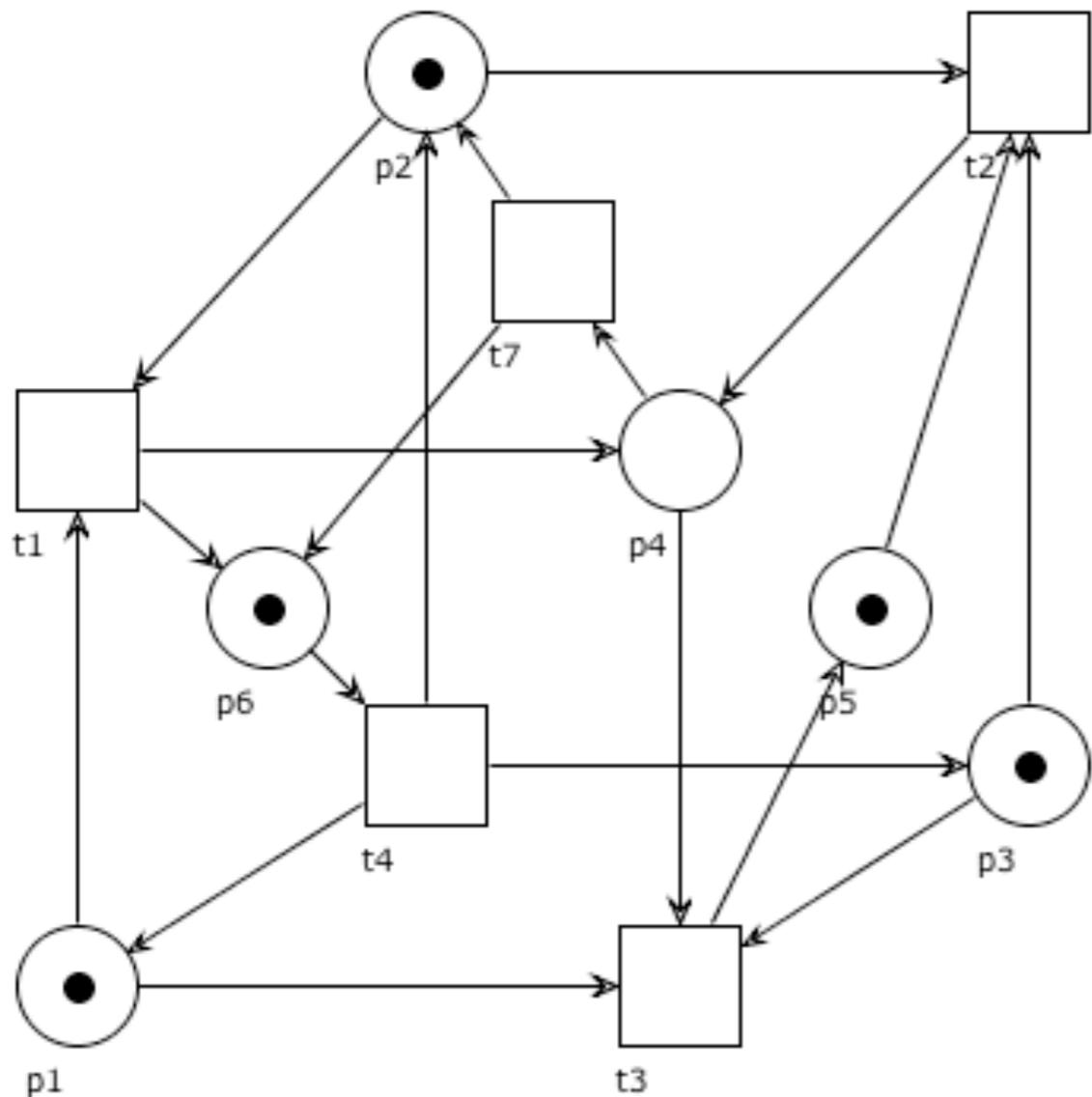


Which of the following holds true?

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

# Question time

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



Which of the following holds true?

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

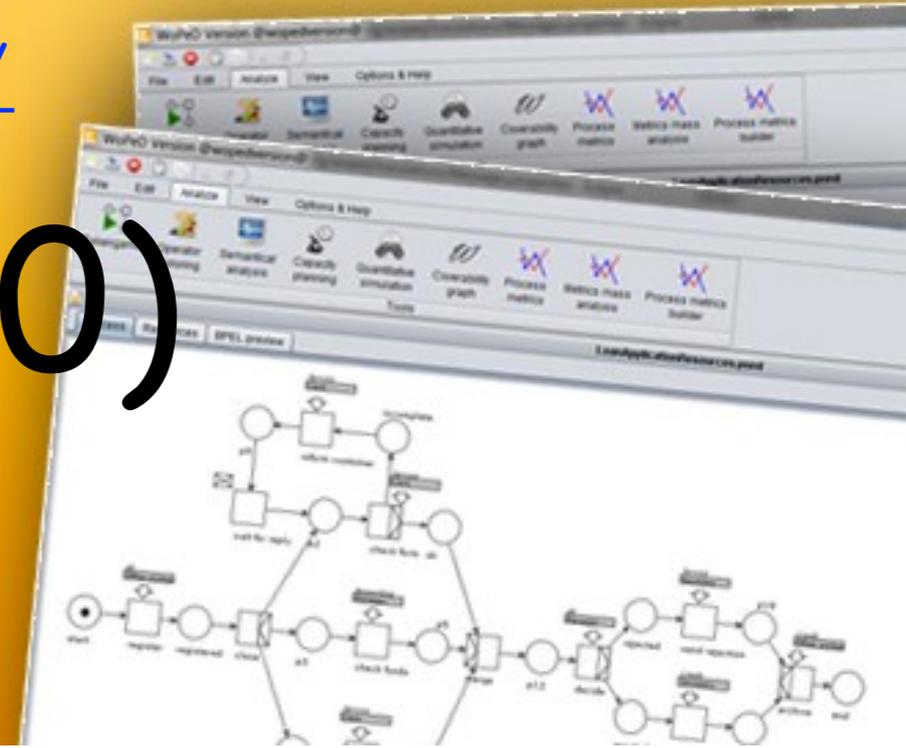
<http://woped.dhbw-karlsruhe.de/woped/>

# WoPeD (3.2.0)



Workflow Petri Net Designer

*Download WoPeD at sourceforge!*



WoPeD 3.2.0

Analyze View Options & Help Community

Token game Operator coloring Semantical analysis Capacity planning Quantitative simulation Coverability graph Show metrics Metrics mass analysis Metrics builder

Analysis tools Process metrics

08-cube.pnml

Process Resources BPEL preview

Horizontal Zoom: 100% Saved

08-cube.pnml

# Notation

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

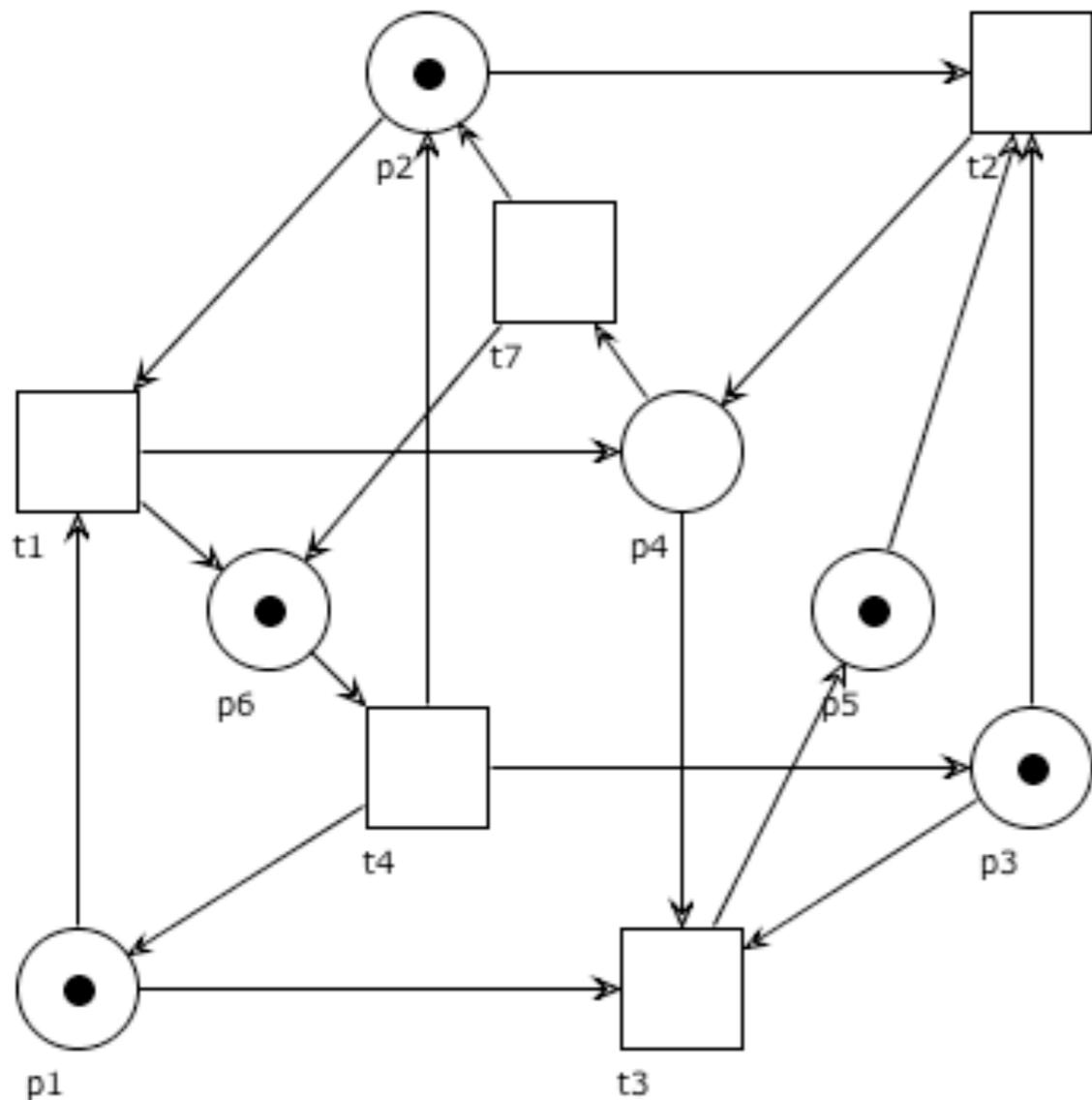
We write  $M \rightarrow M'$  if  $M \xrightarrow{t} M'$  for some transition  $t$

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

- $M_0 \longrightarrow$
- $M_0 \longrightarrow p_1 + p_4 + p_6$
- $M_0 \not\stackrel{t_7}{\longrightarrow}$
- $p_1 + p_5 \not\longrightarrow$

# 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  $M \xrightarrow{*} M'$

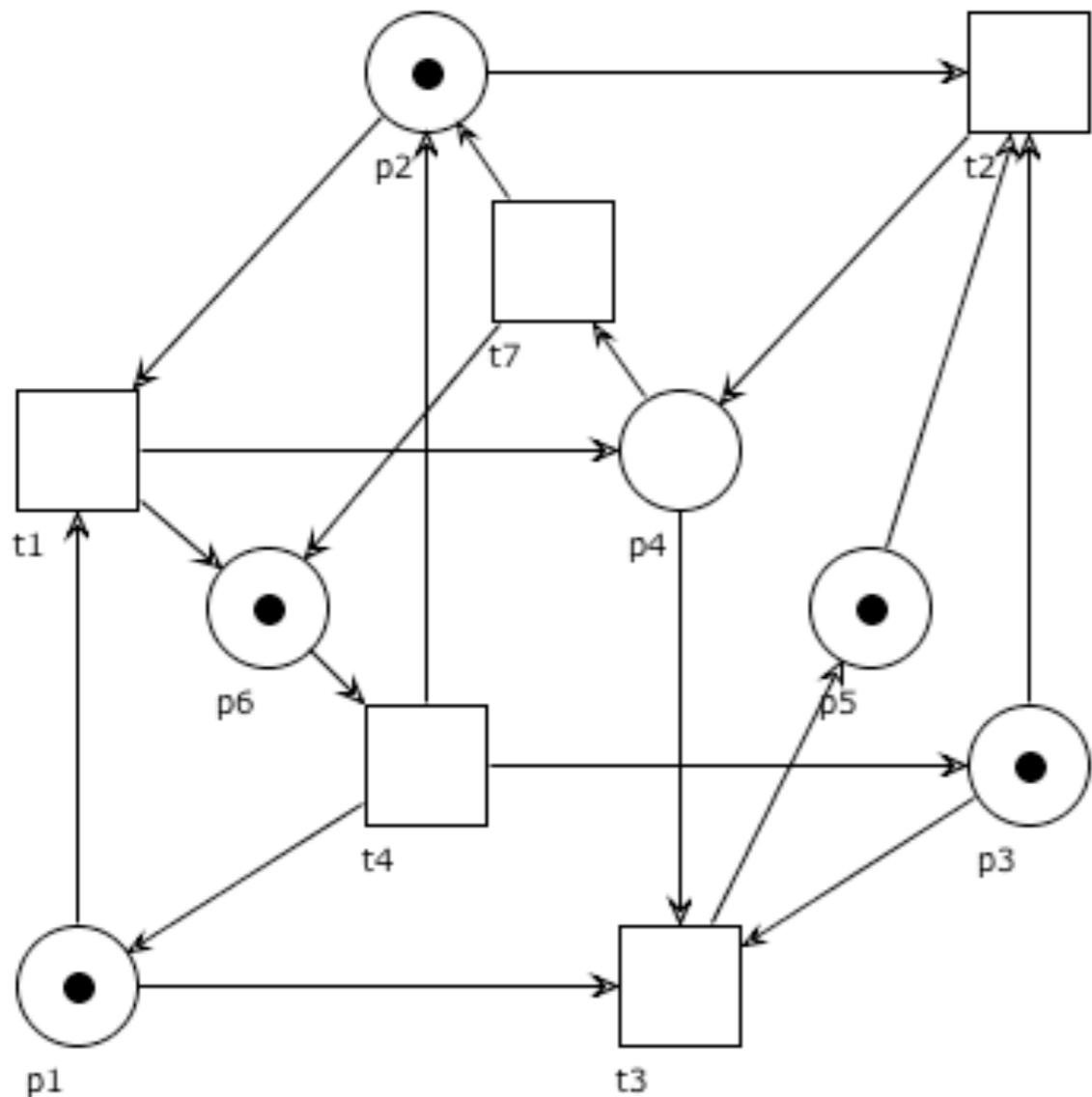
Note that  $M \xrightarrow{\epsilon} M$  for  $\epsilon$  the empty sequence

The set of markings reachable from  $M$  is often denoted:

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

# Question time

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

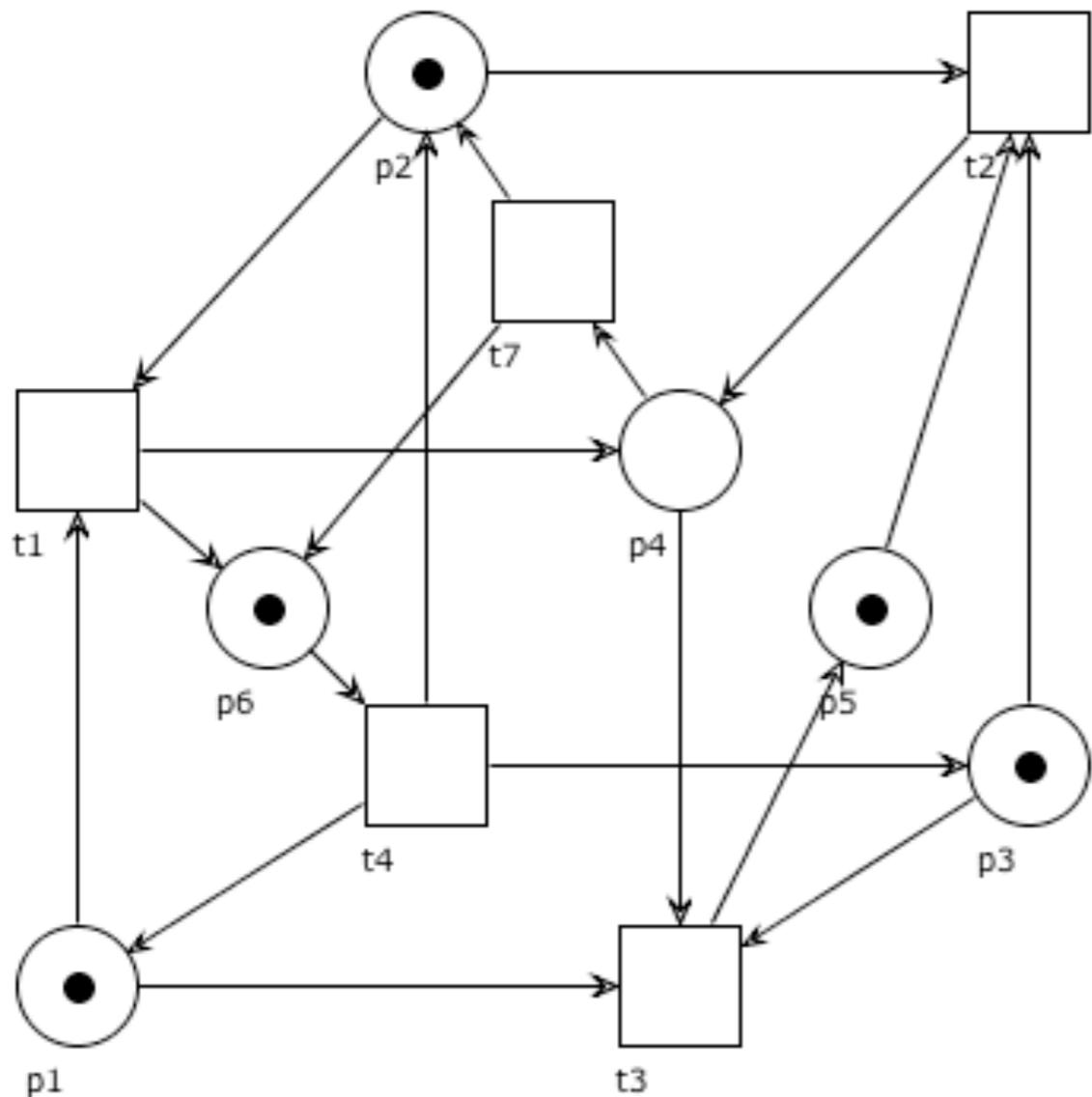


Which of the following holds true?

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

# 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:

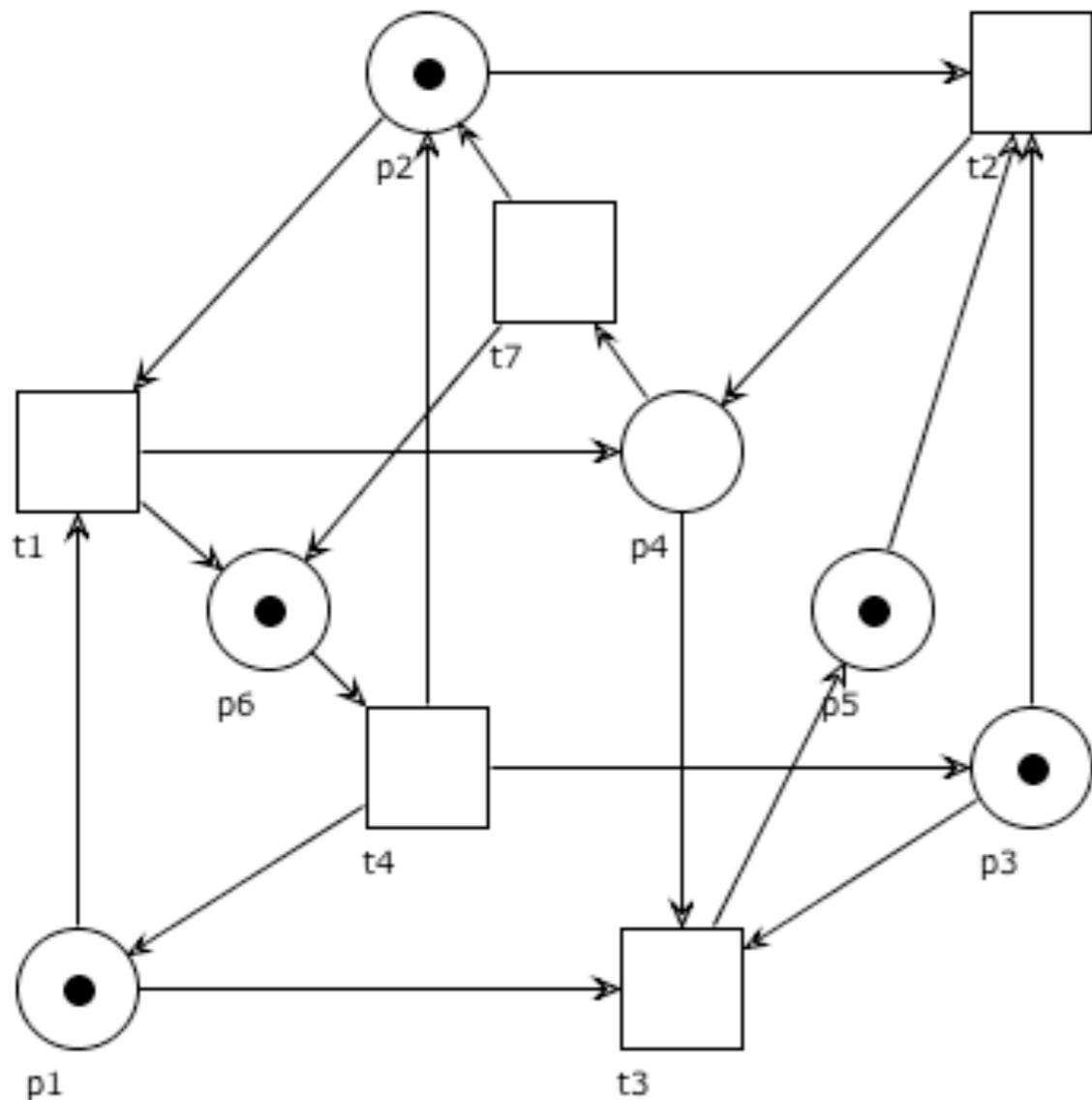
there is an infinite sequence of markings  $M_1, M_2, \dots$

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  $\sigma$  is **enabled** if  $M \xrightarrow{\sigma}$

( $\sigma$  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

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

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

# Enabledness

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

( $\Rightarrow$ ) immediate from definition

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

When  $\sigma$  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  $\sigma$ .

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  $\sigma$  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

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

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

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

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

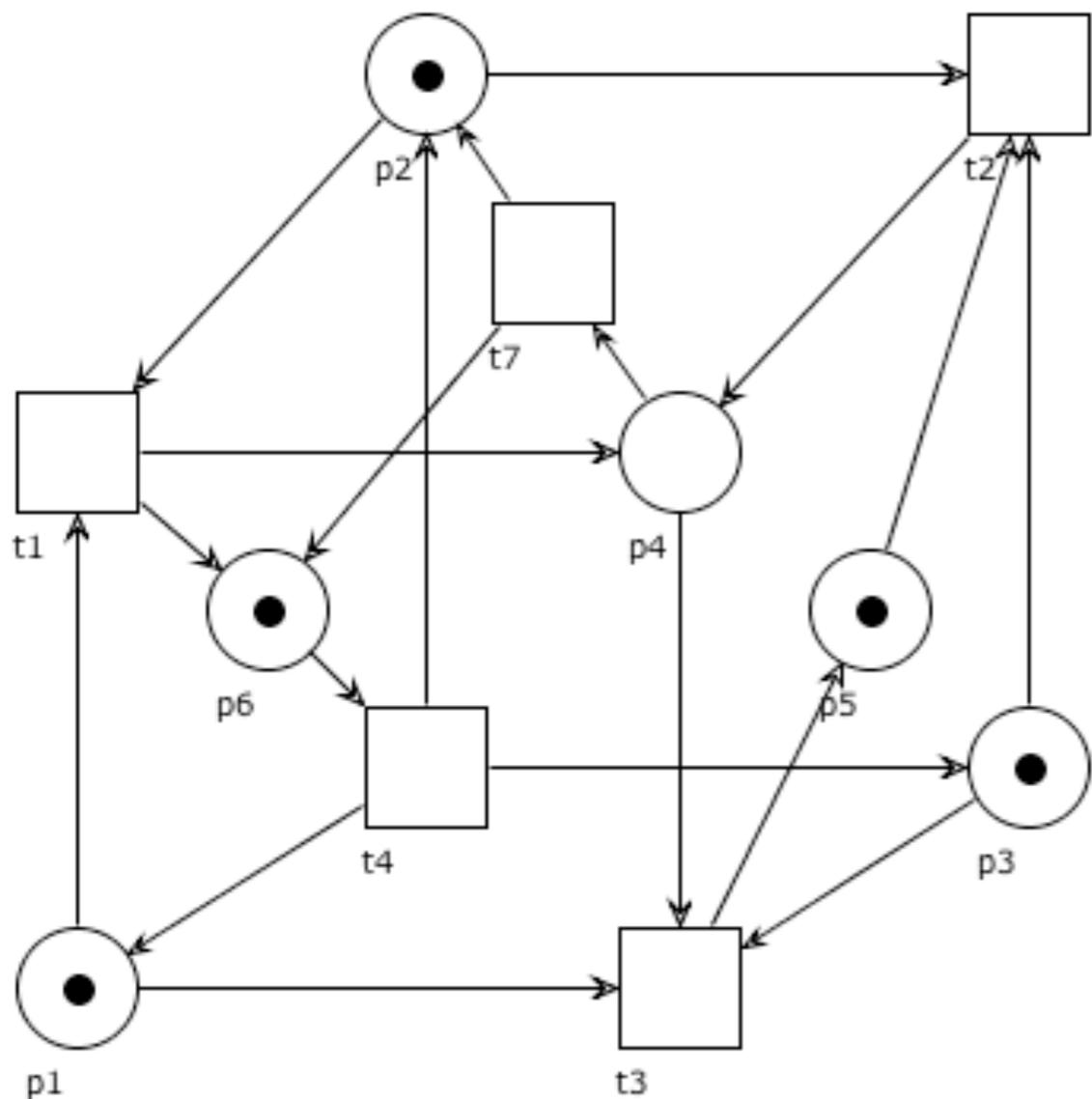
$$= t_1 t_4 t_1 t_4 (t_7) |_{\{t_1, t_4\}}$$

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

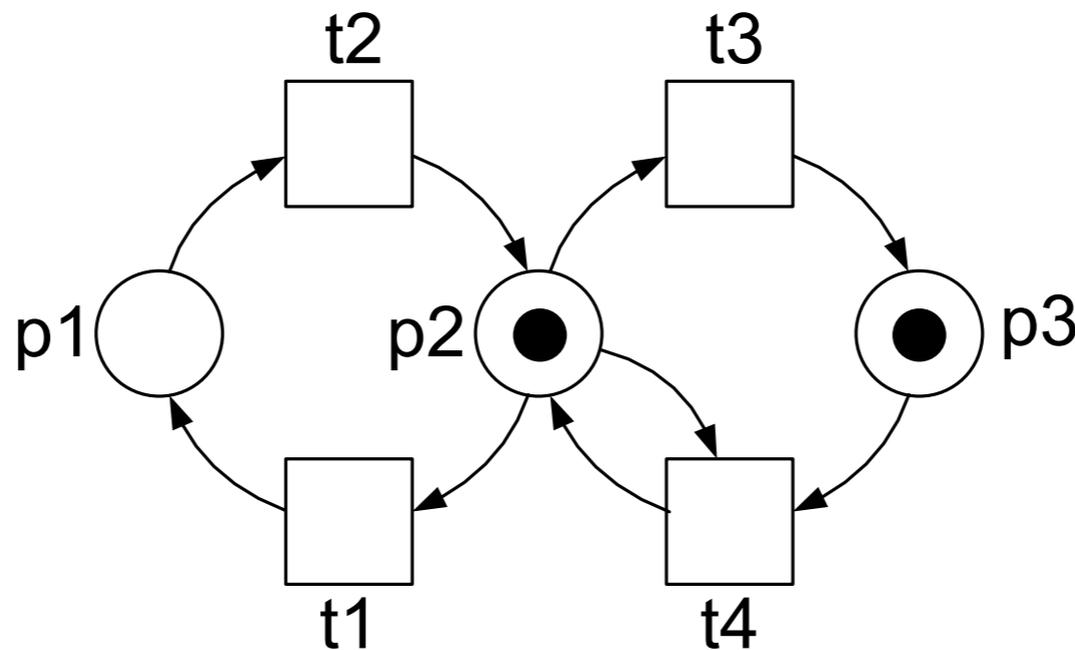
$$= t_1 t_4 t_1 t_4 (\epsilon) |_{\{t_1, t_4\}}$$

$$= t_1 t_4 t_1 t_4 \epsilon$$

$$= t_1 t_4 t_1 t_4$$



# Exercises



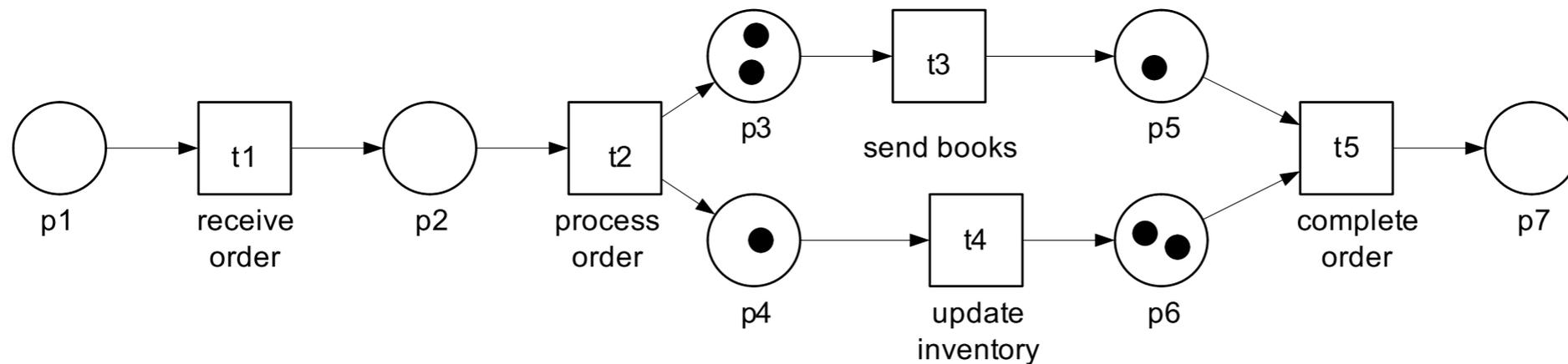
Determine the pre- and post-set of each element

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?

# 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 $OG(N)$

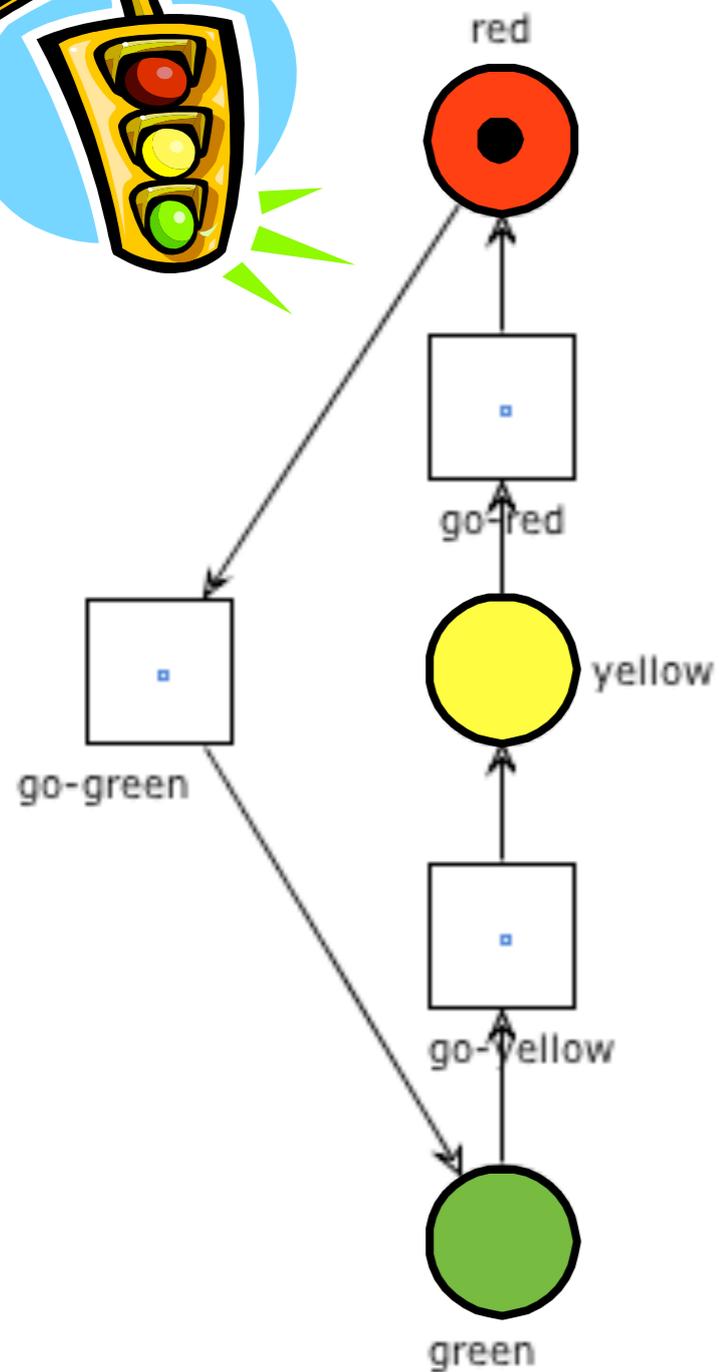
1. Initially  $R = \{ M_0 \}$  and  $A = \emptyset$
2. Take a marking  $M \in R$  and a transition  $t \in T$  such that
  1.  $M$  enables  $t$  and there is no arc labelled  $t$  leaving from  $M$
3. Let  $M' = M - \bullet t + t \bullet$
4. Add  $M'$  to  $R$  and  $(M, t, M')$  to  $A$
5. Repeat steps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

# How to compute $OG(N)$

The occurrence graph can be constructed as follows:

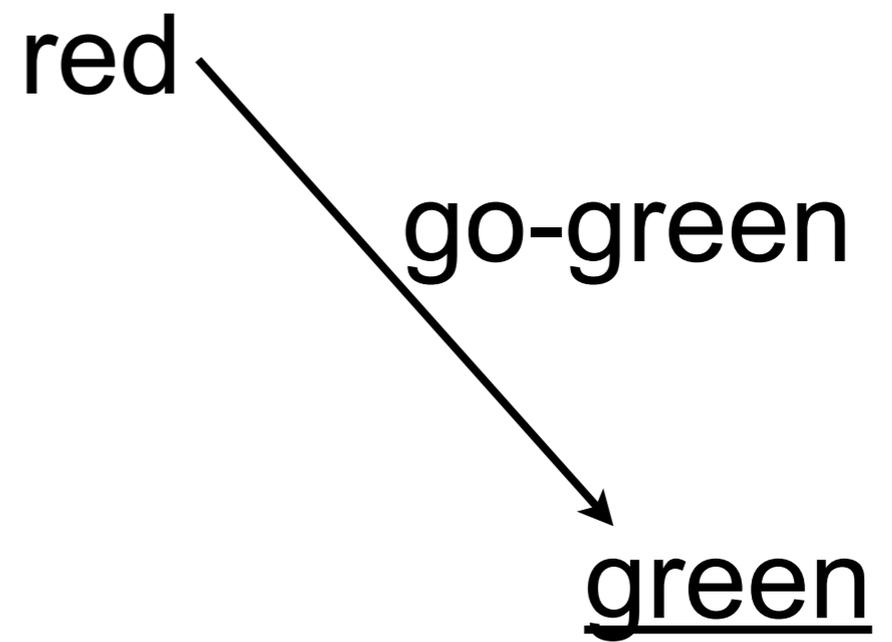
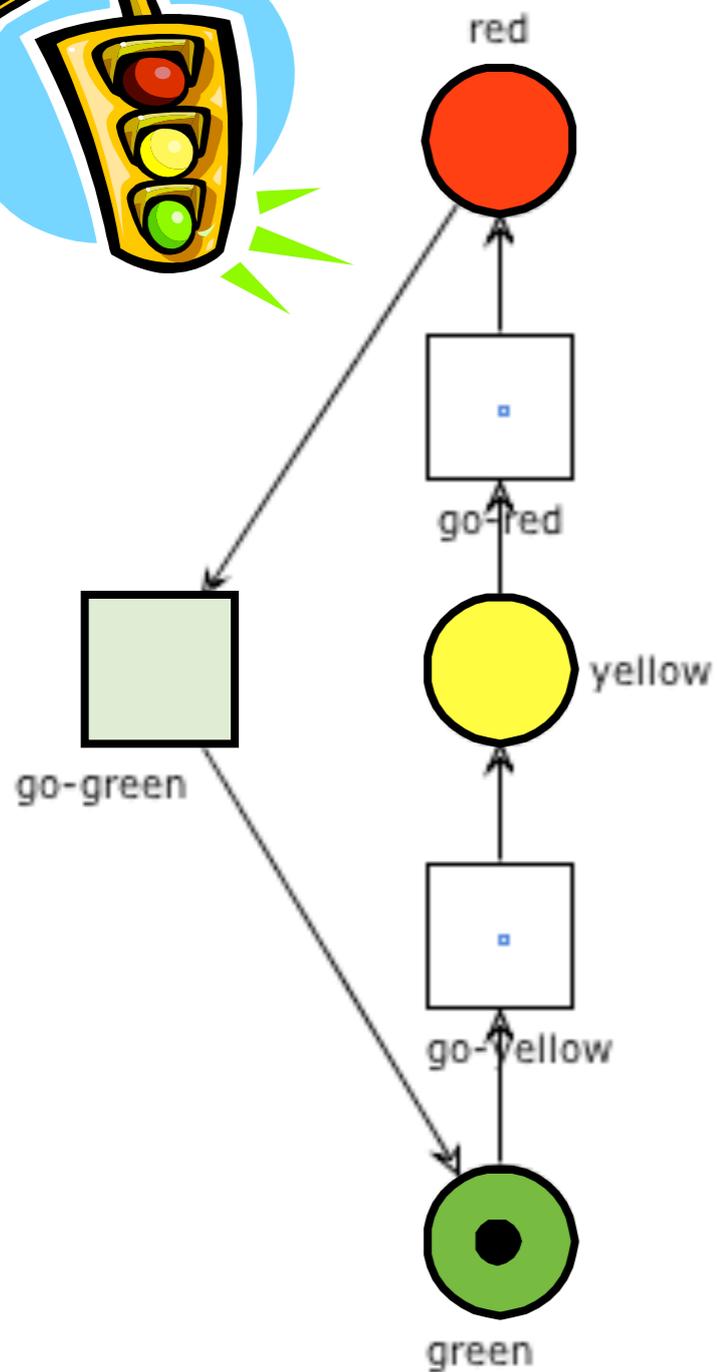
1.  $Nodes = \{\}, Arcs = \{\}, Todo = \{M_0\}$
2.  $M = next(Todo)$
3.  $Nodes = Nodes \cup \{M\}, Todo = Todo \setminus \{M\}$
4.  $Firings = \{(M, t, M') \mid \exists t \in T, \exists M' \in \mu(P), M \xrightarrow{t} M'\}$
5.  $New = \{M' \mid (M, t, M') \in Firings\} \setminus (Nodes \cup Todo)$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

# Example: traffic light

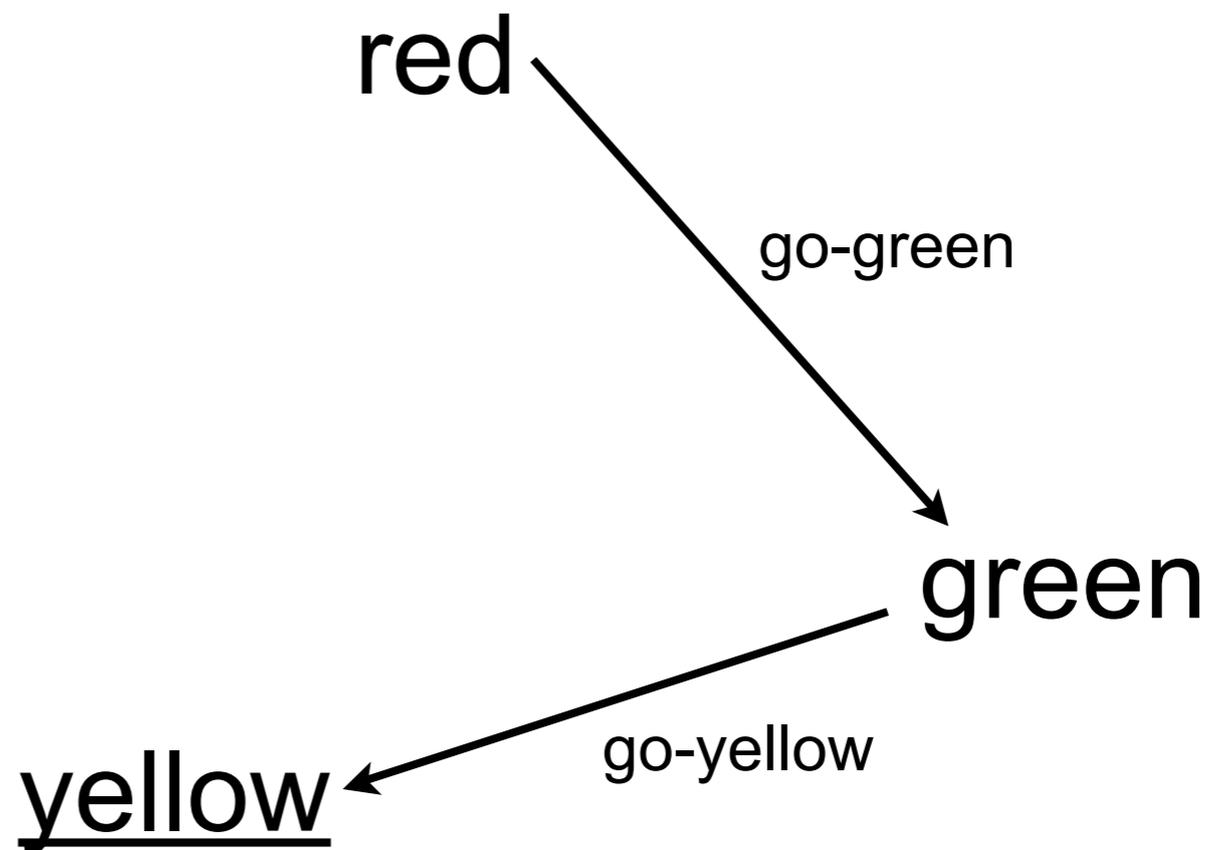
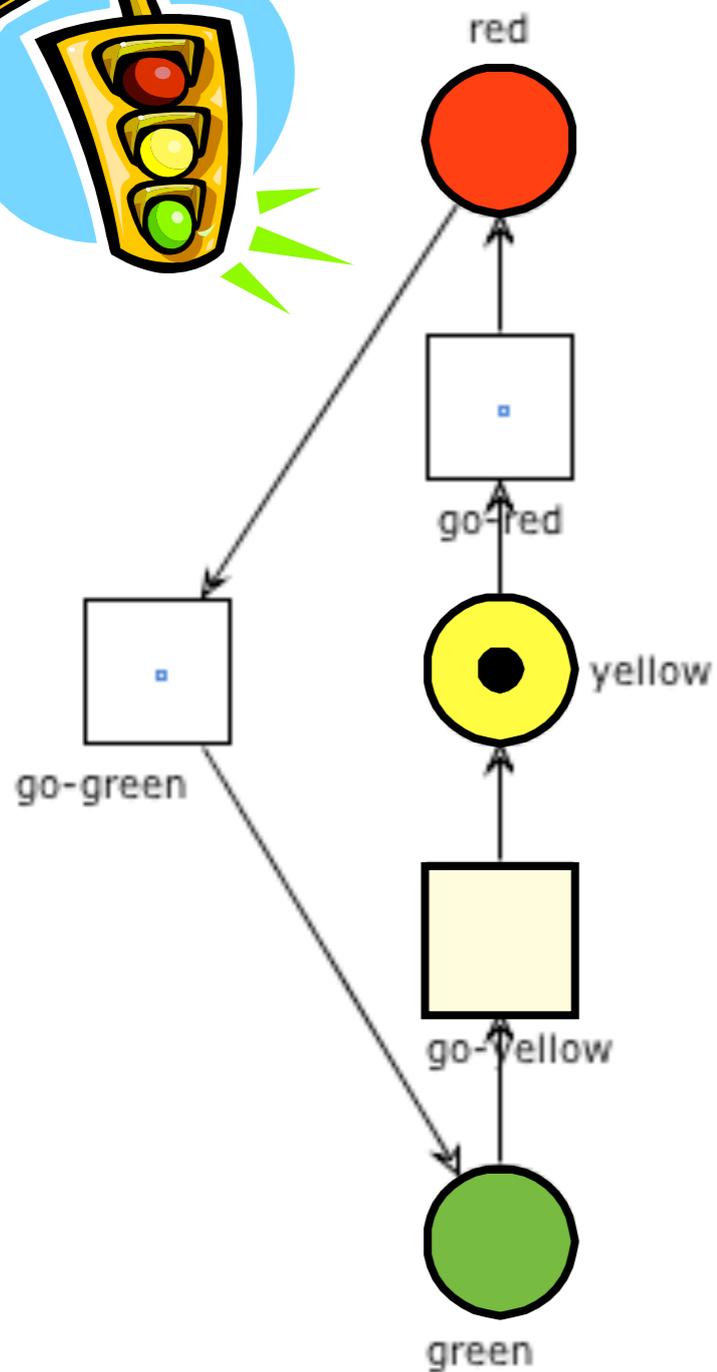


red

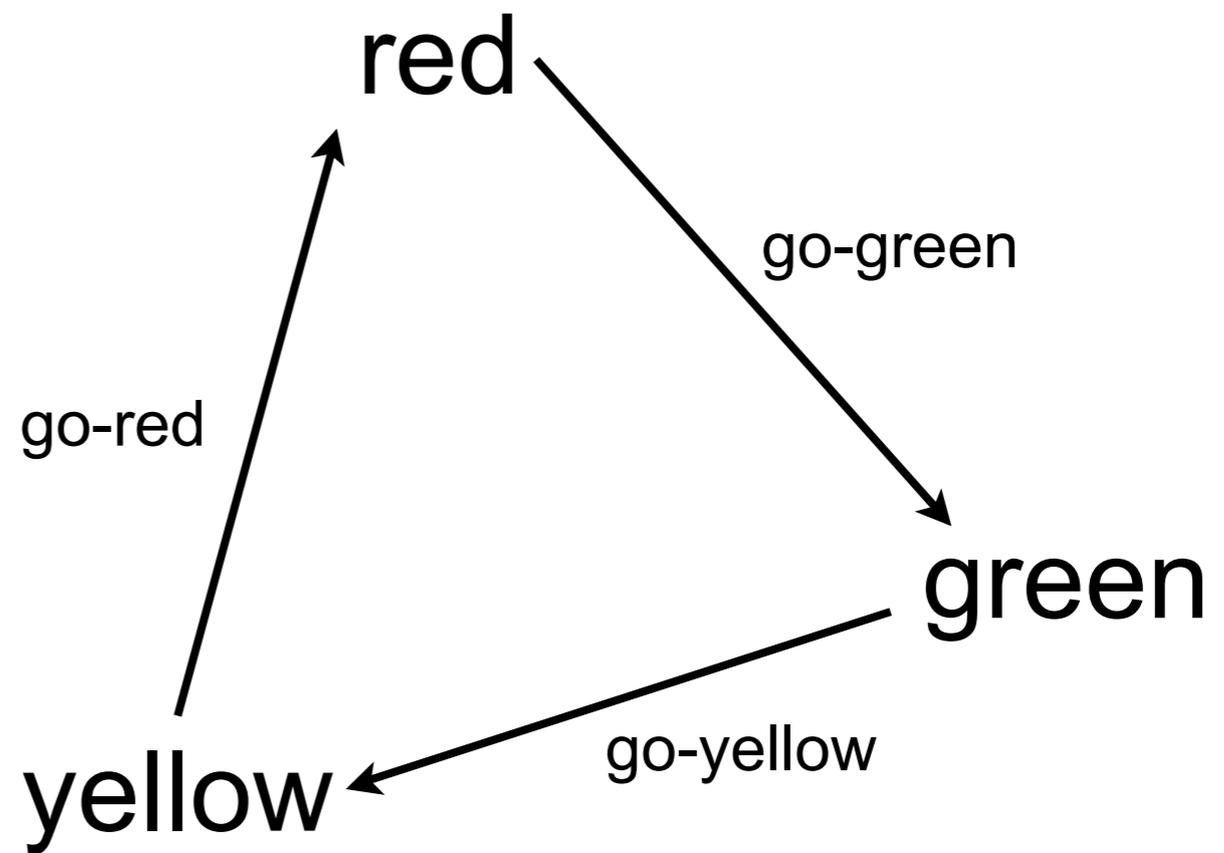
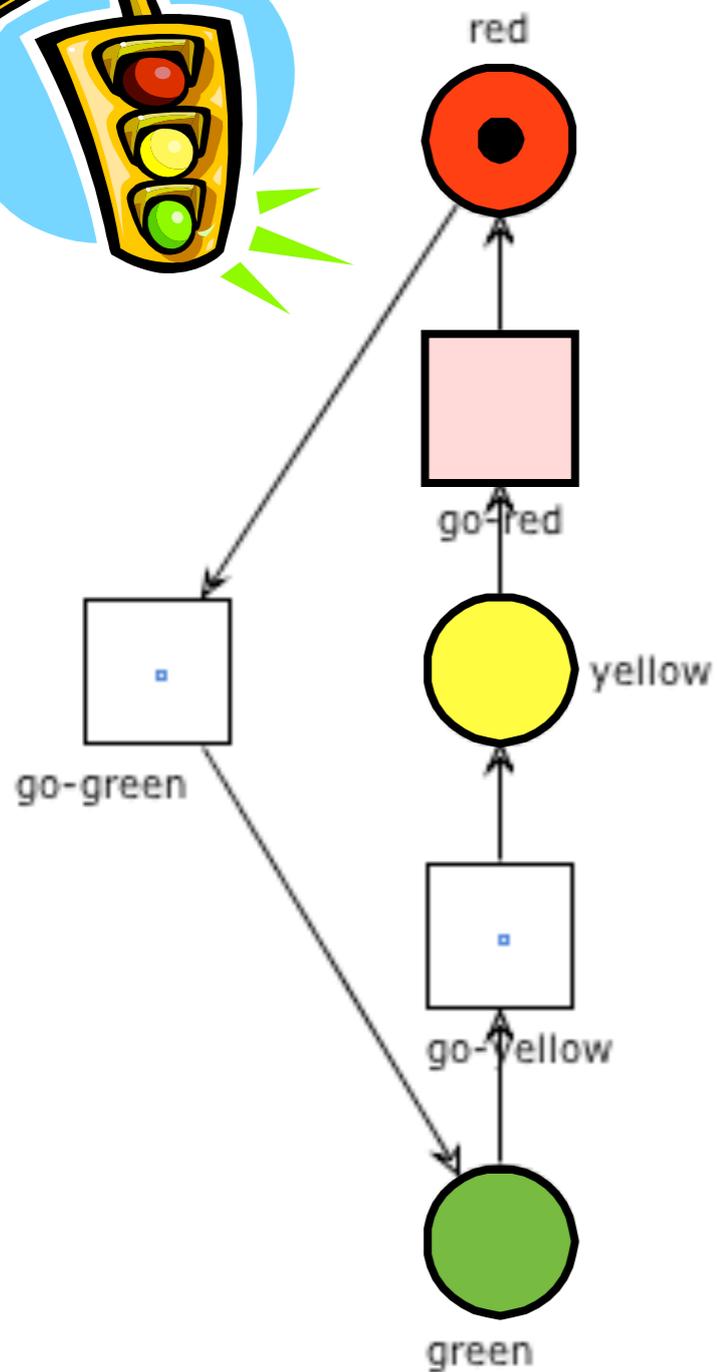
# Example: traffic light



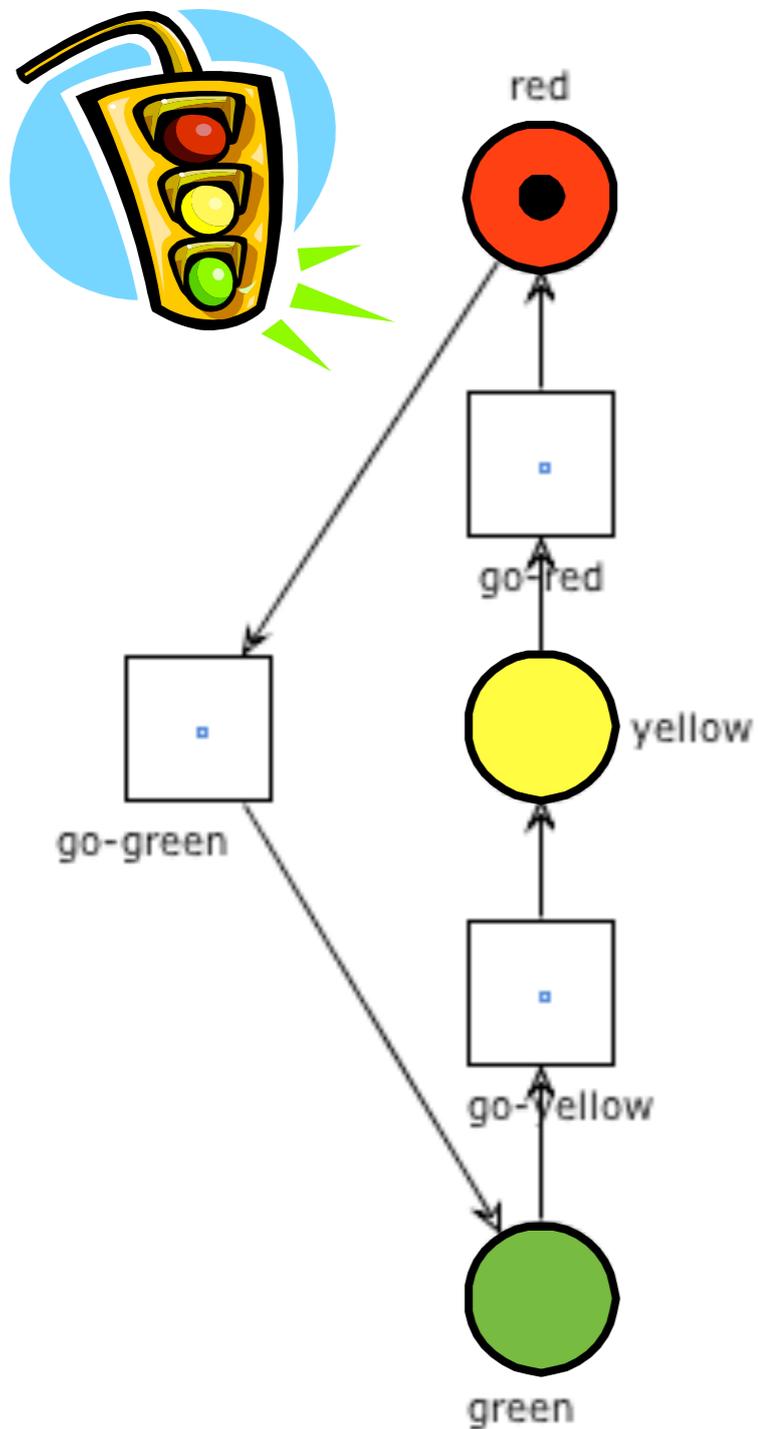
# Example: traffic light



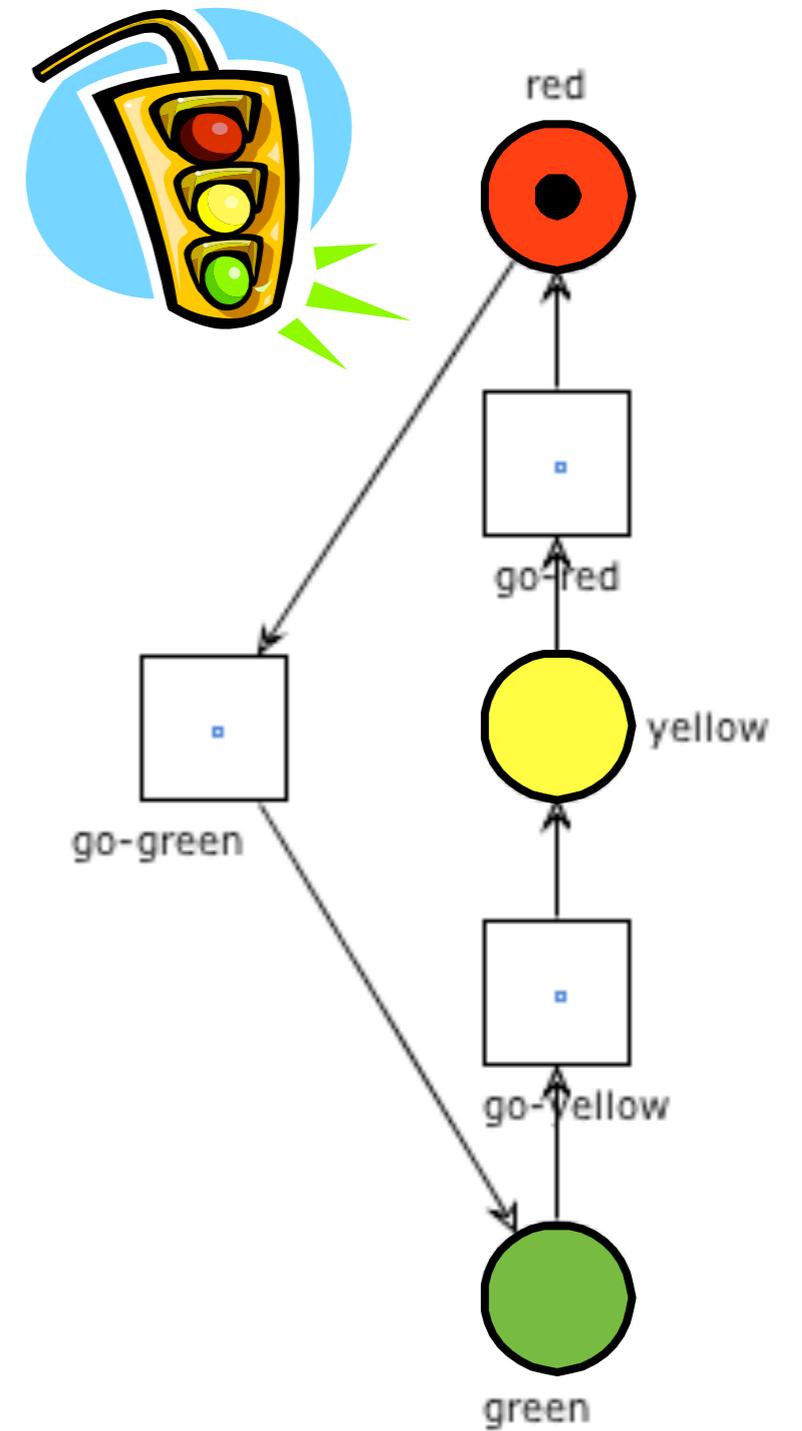
# Example: traffic light



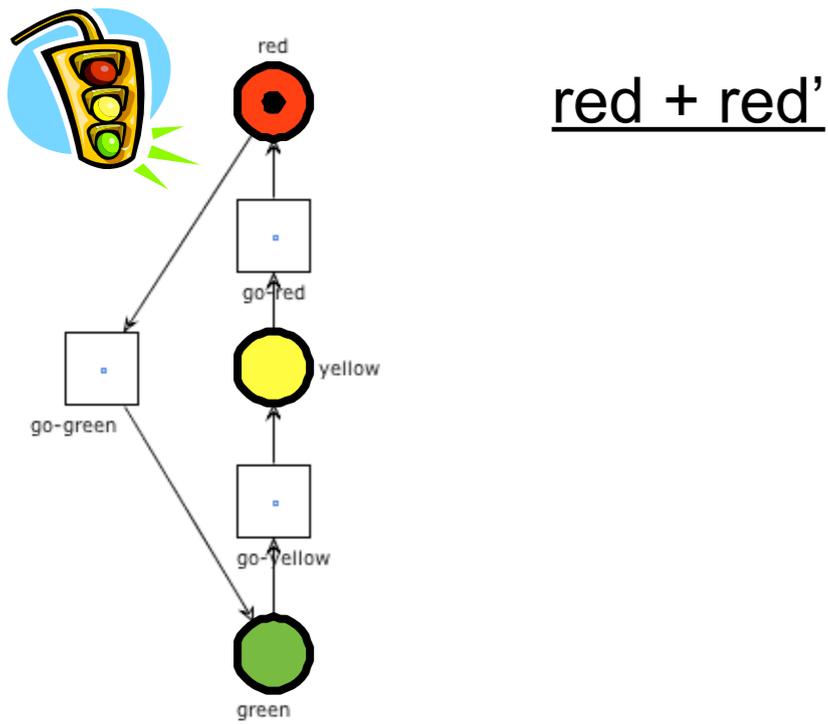
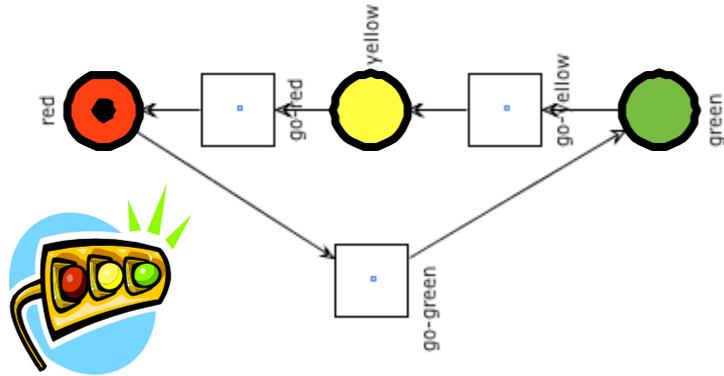
# Example: two traffic lights



red + red'

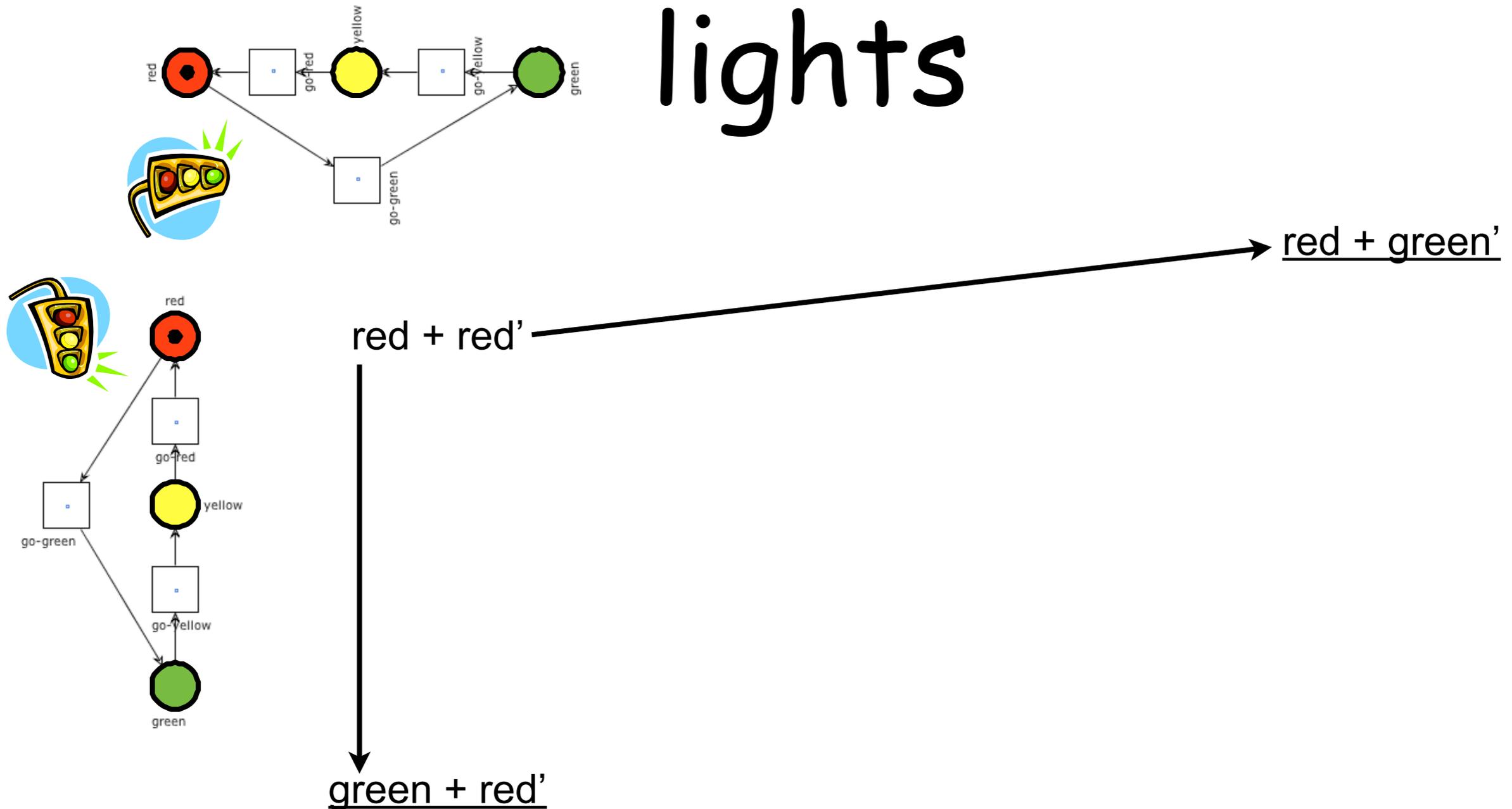


# Example: two traffic lights



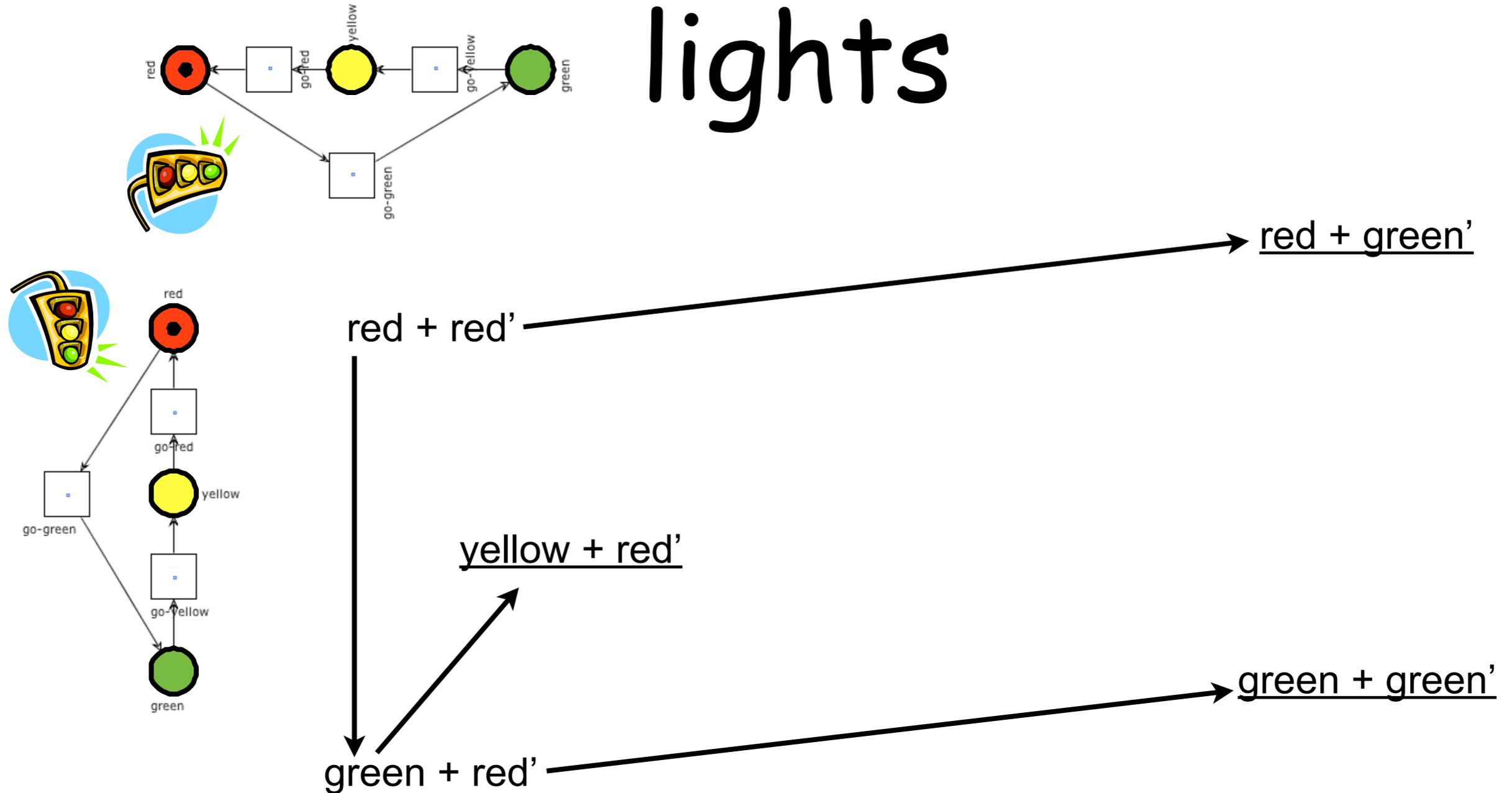
(we omit arc labels  
for readability issues)

# Example: two traffic lights



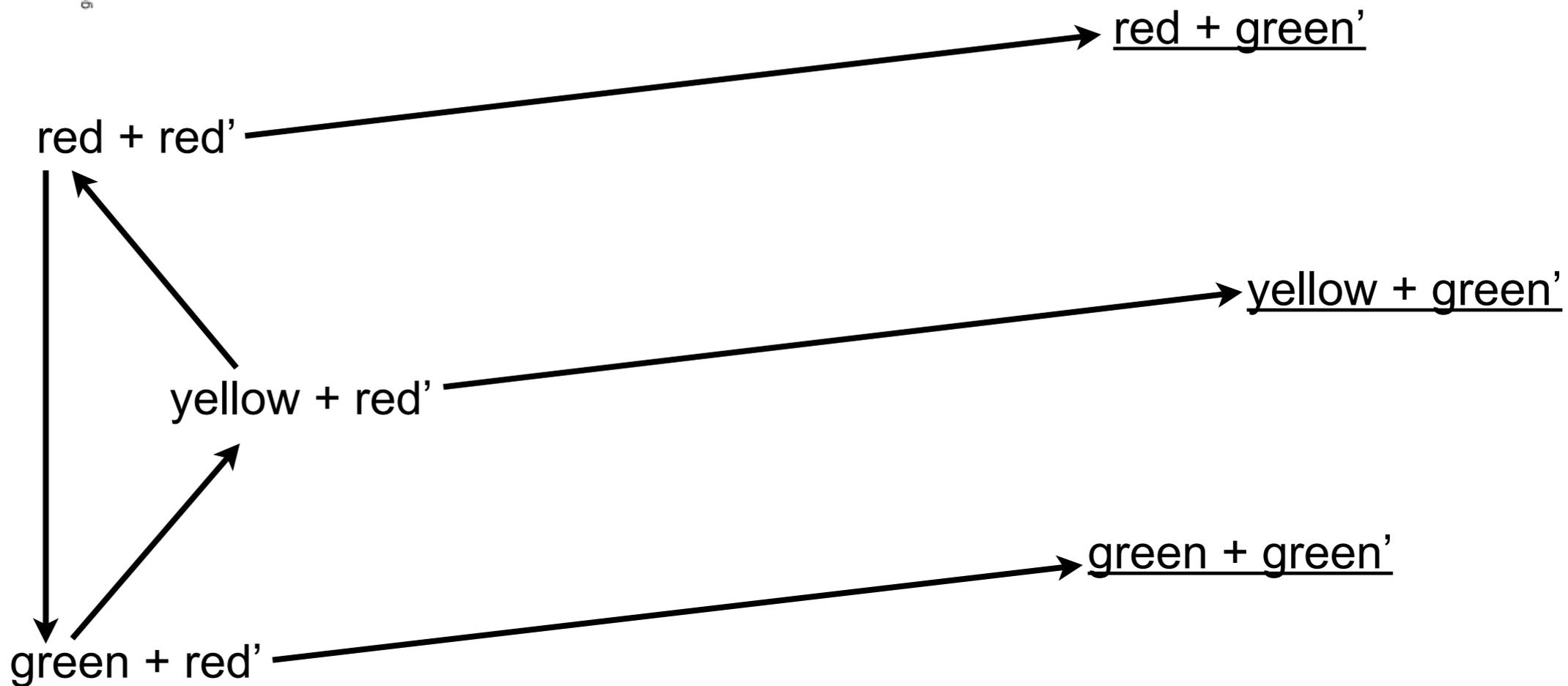
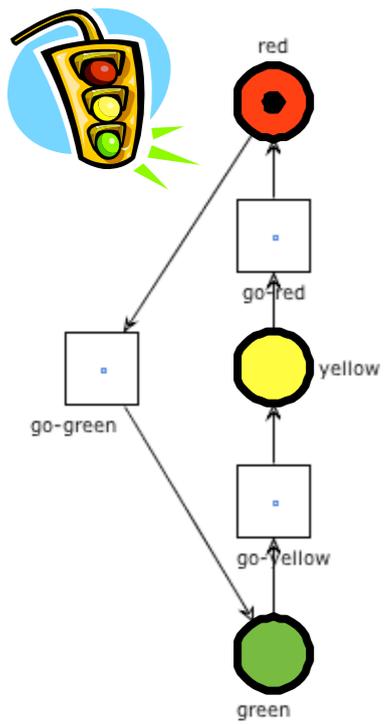
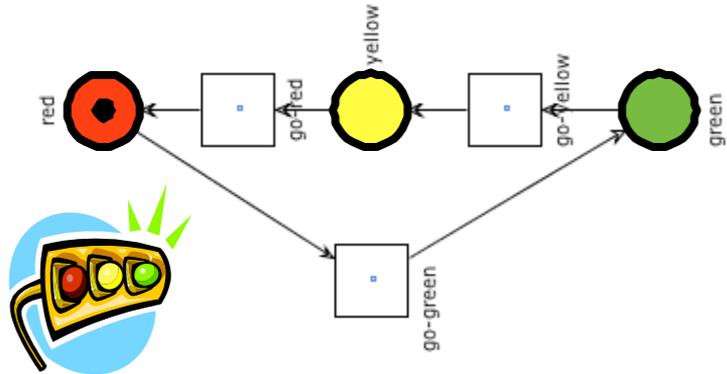
(we omit arc labels for readability issues)

# Example: two traffic lights



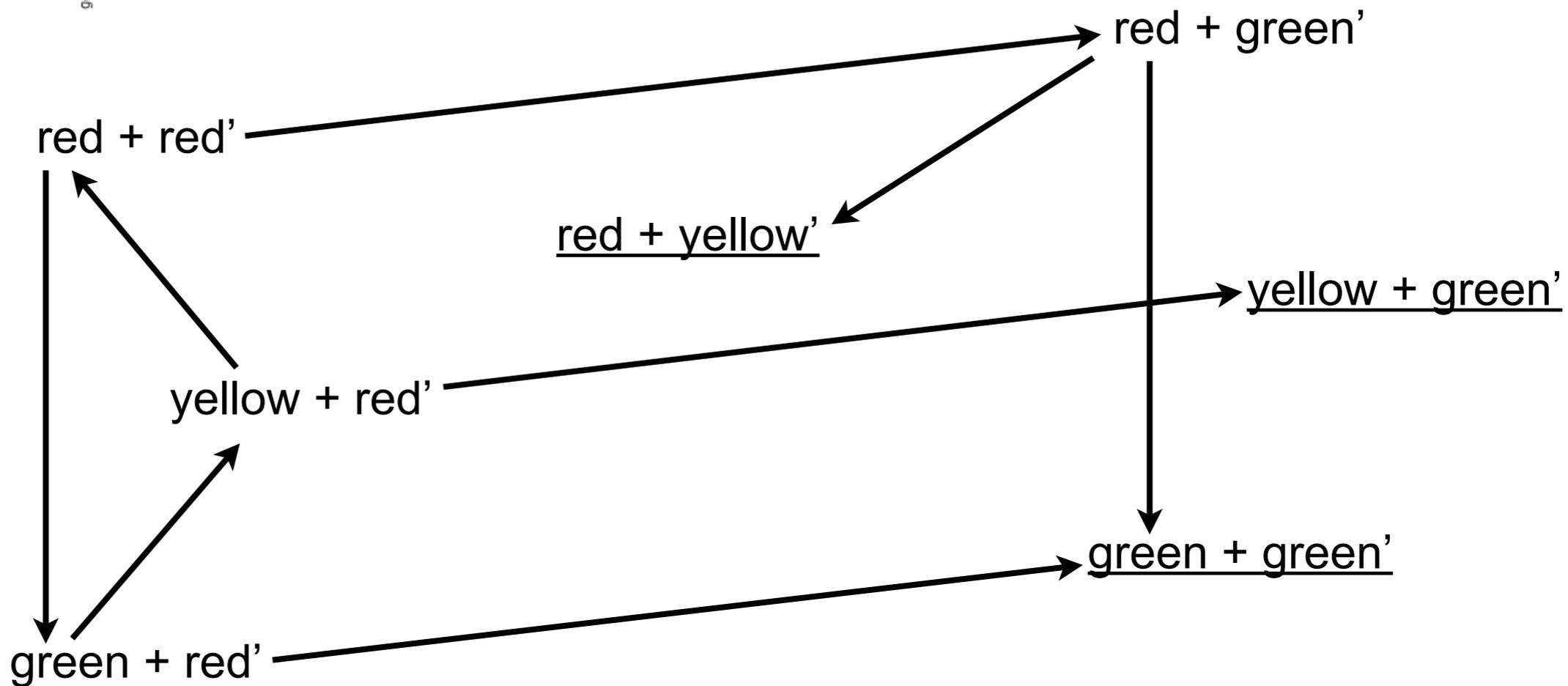
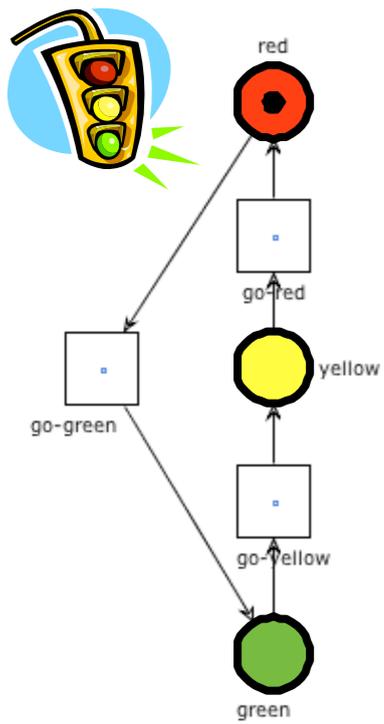
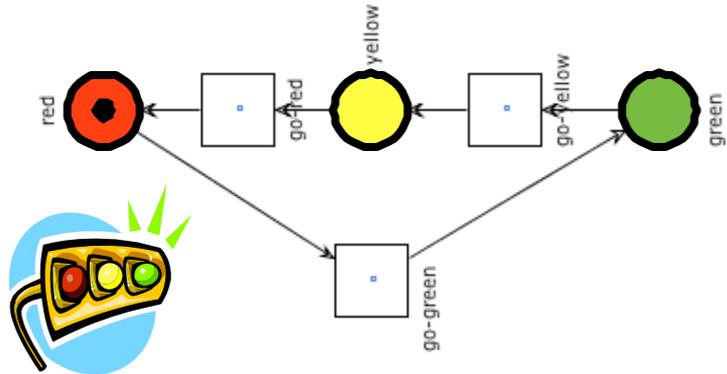
(we omit arc labels for readability issues)

# Example: two traffic lights



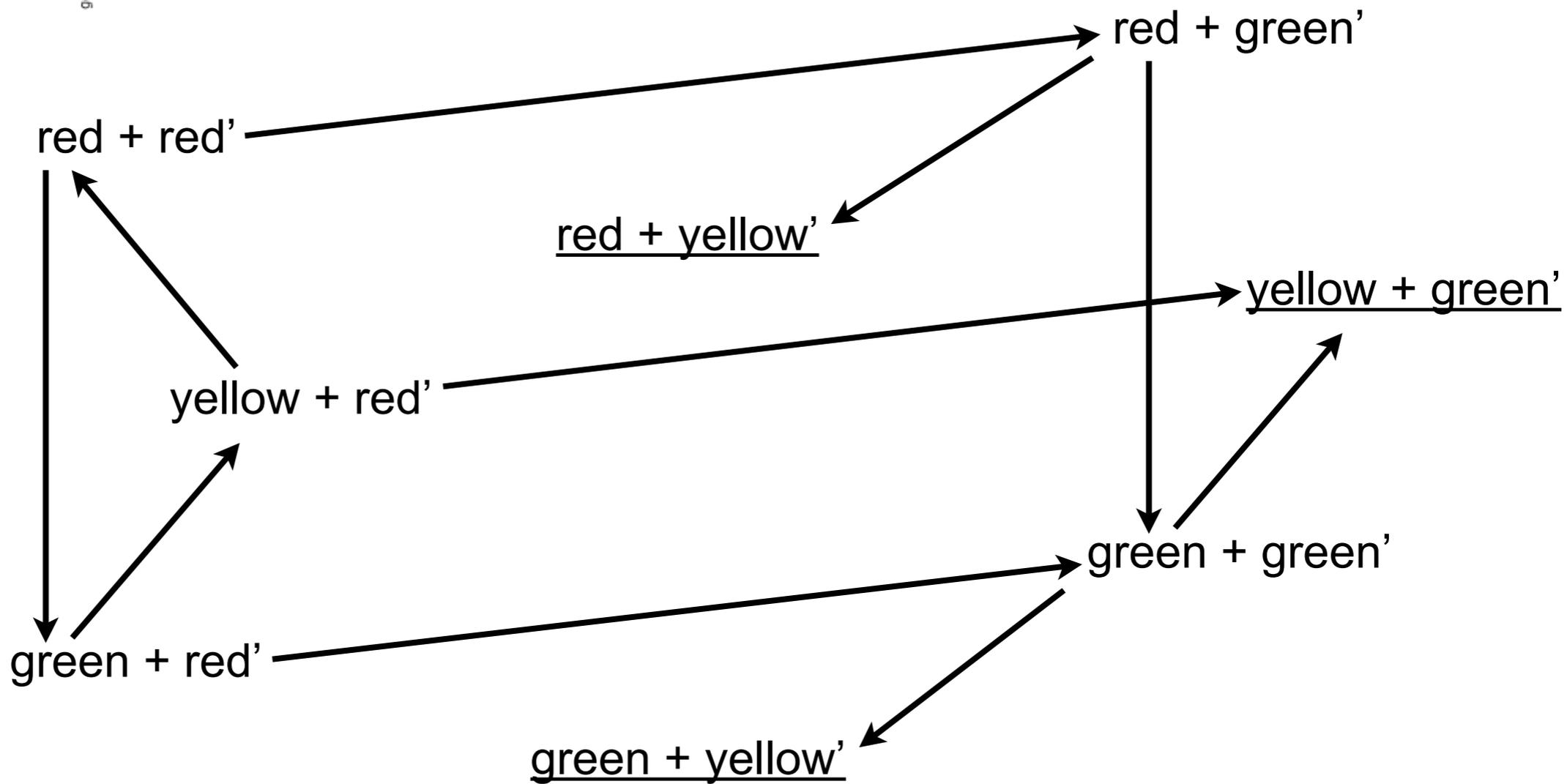
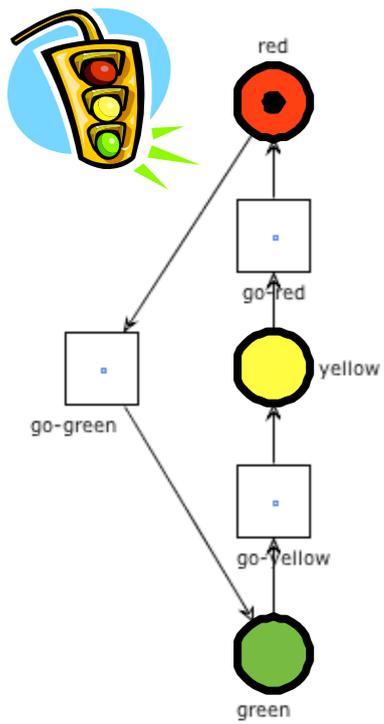
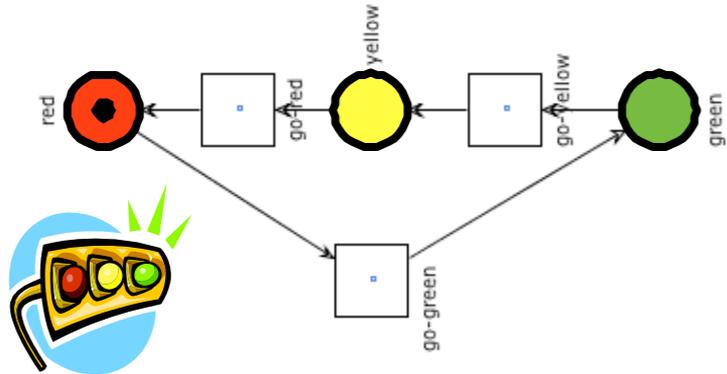
(we omit arc labels for readability issues)

# Example: two traffic lights



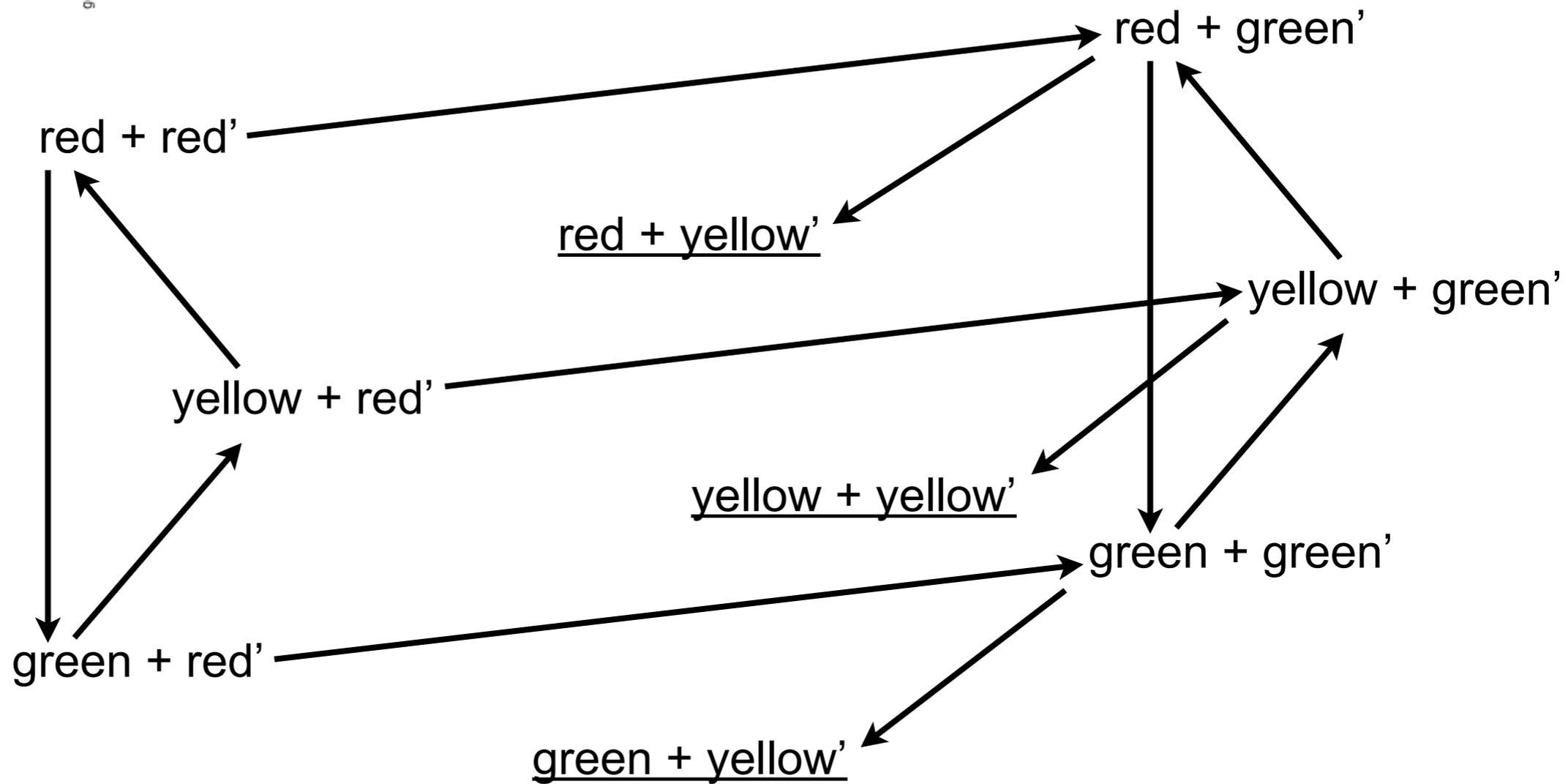
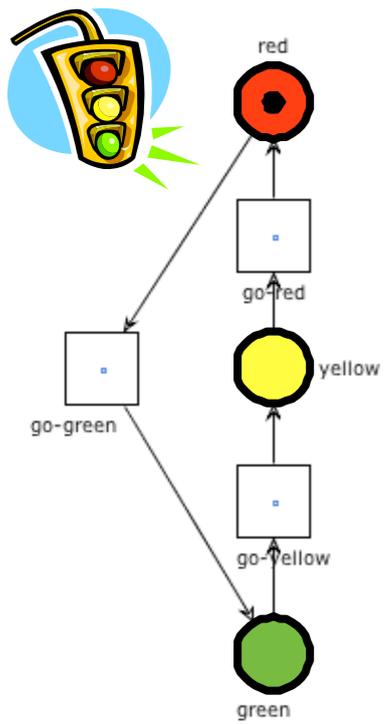
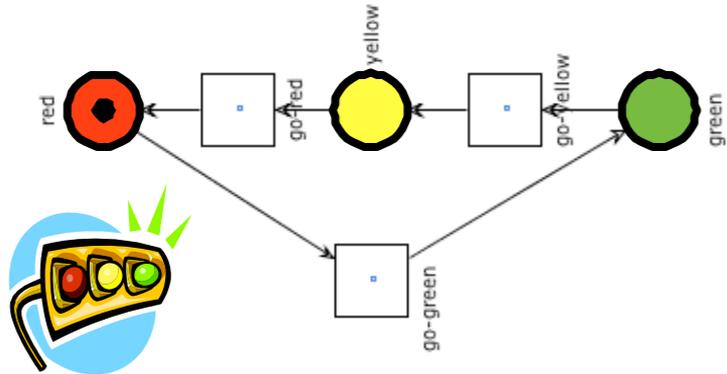
(we omit arc labels for readability issues)

# Example: two traffic lights



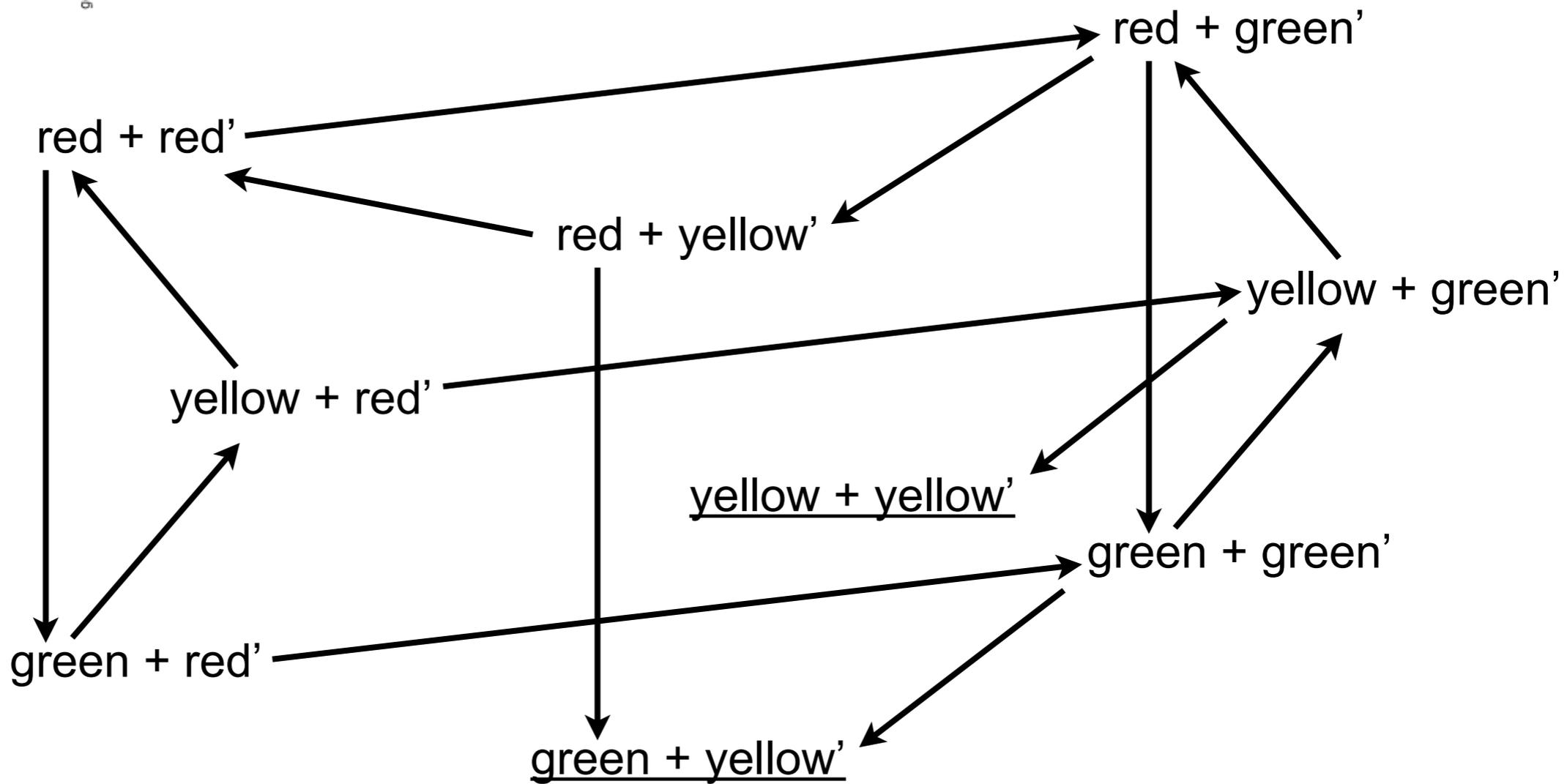
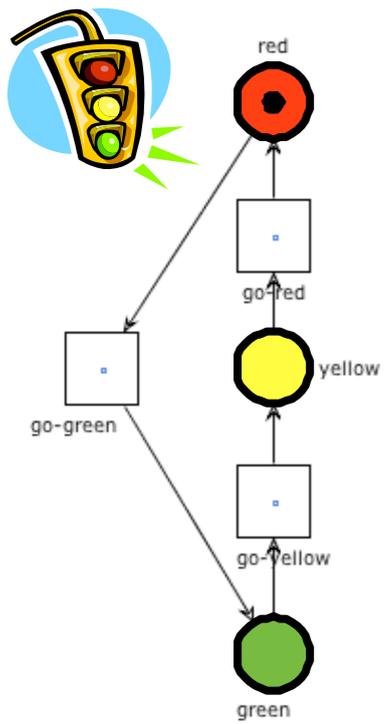
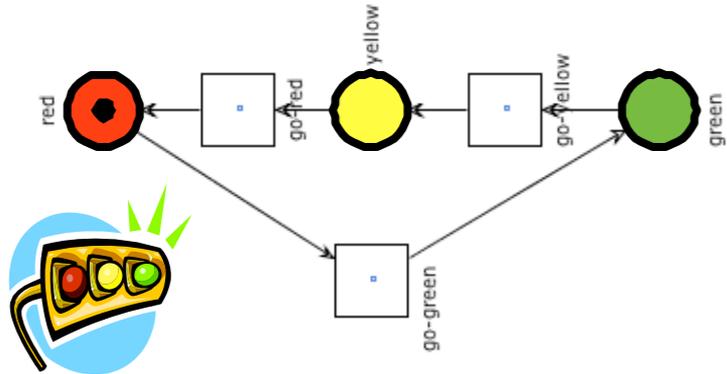
(we omit arc labels for readability issues)

# Example: two traffic lights



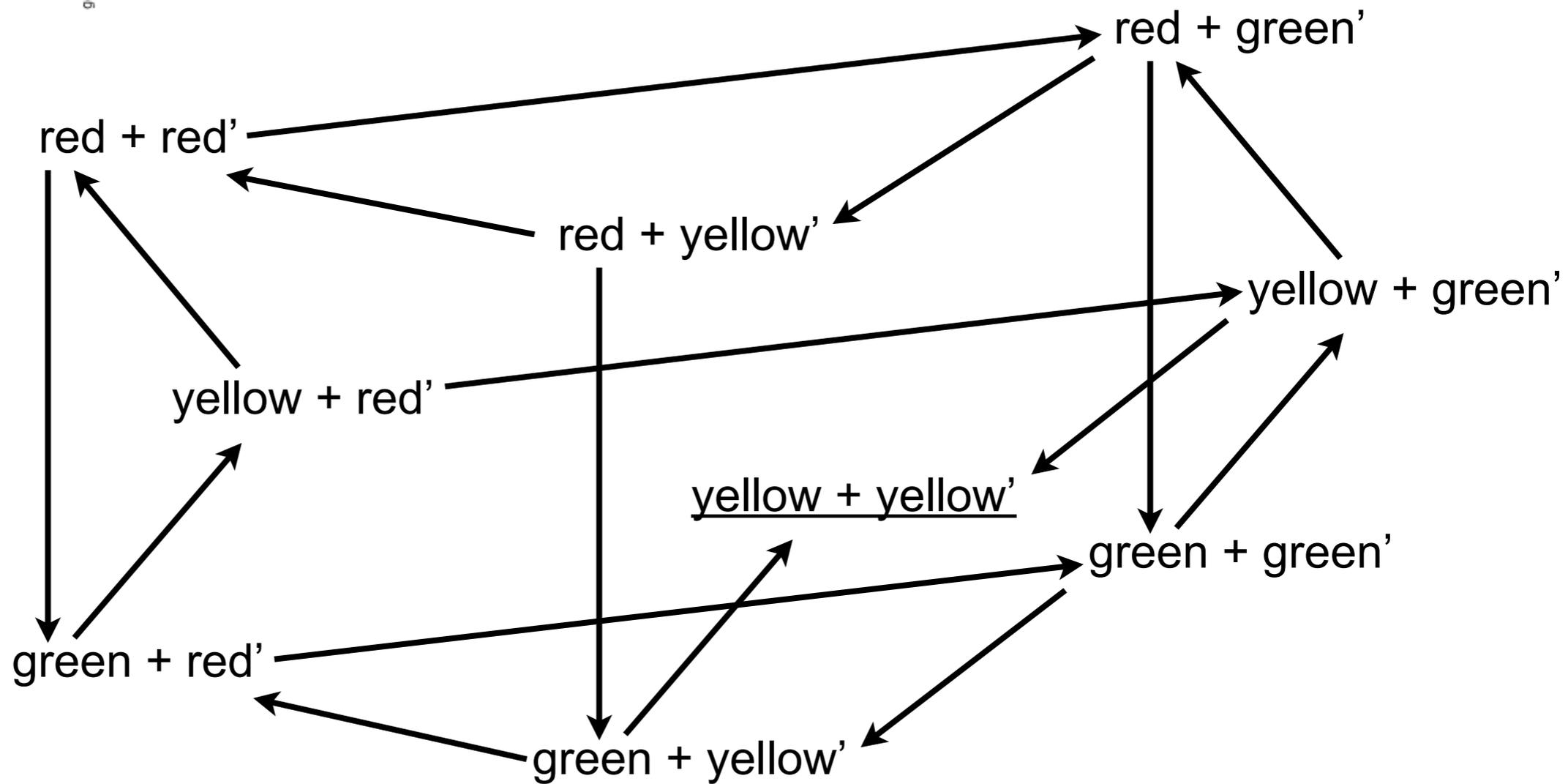
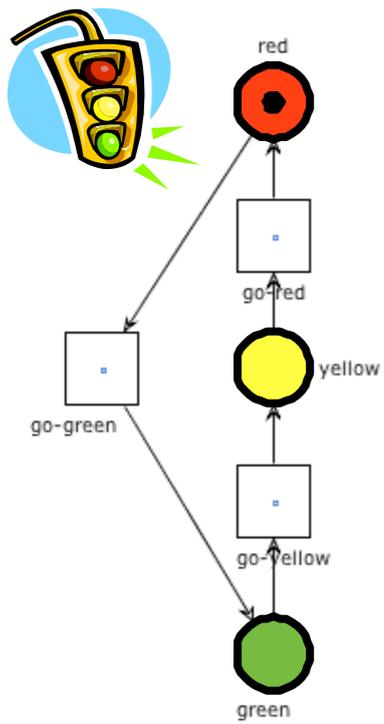
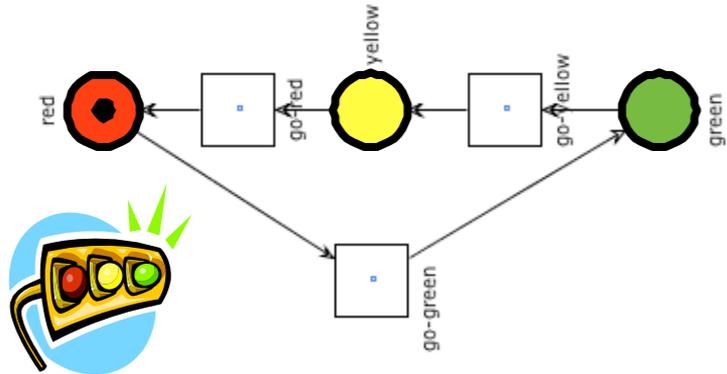
(we omit arc labels for readability issues)

# Example: two traffic lights



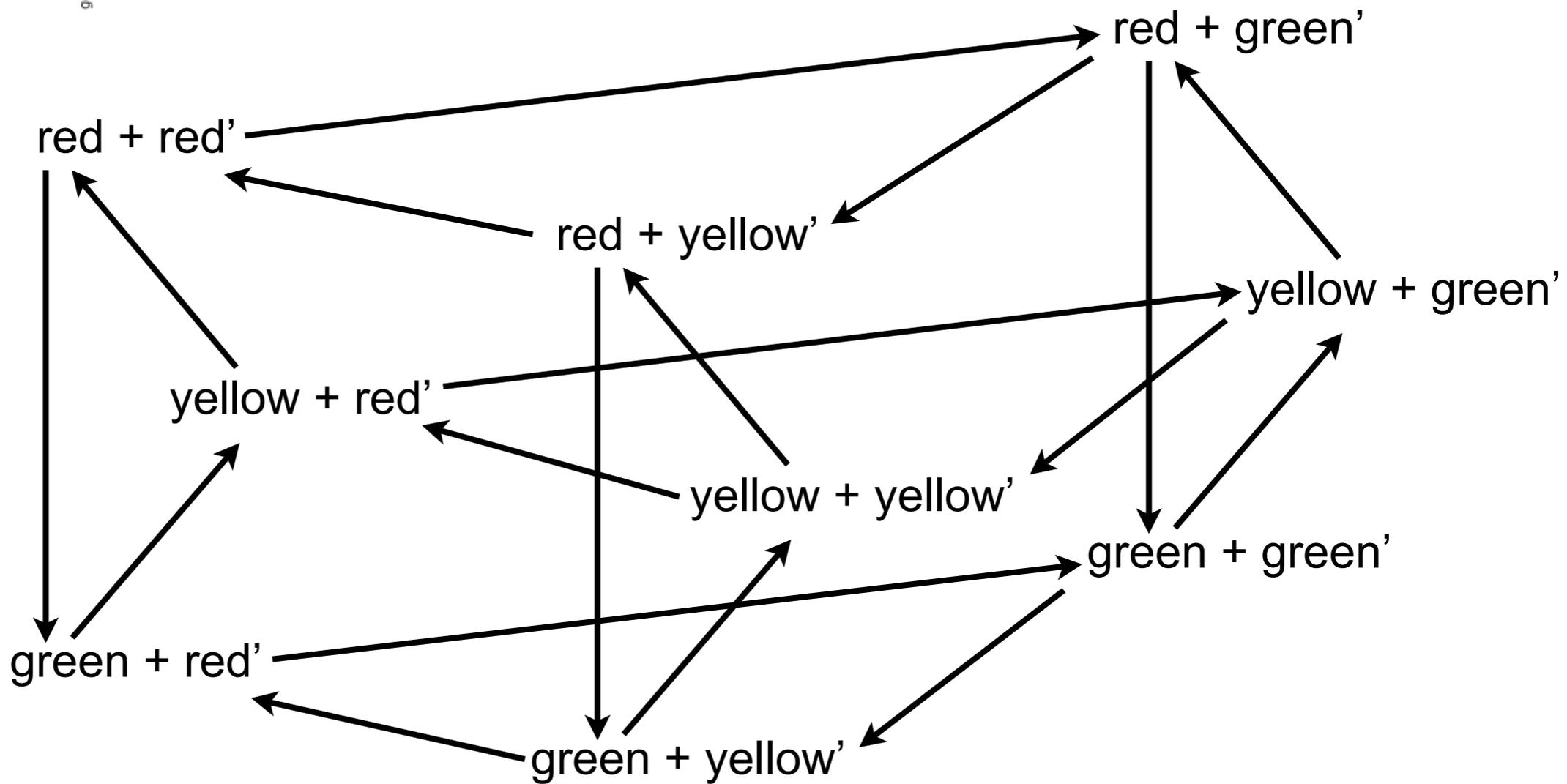
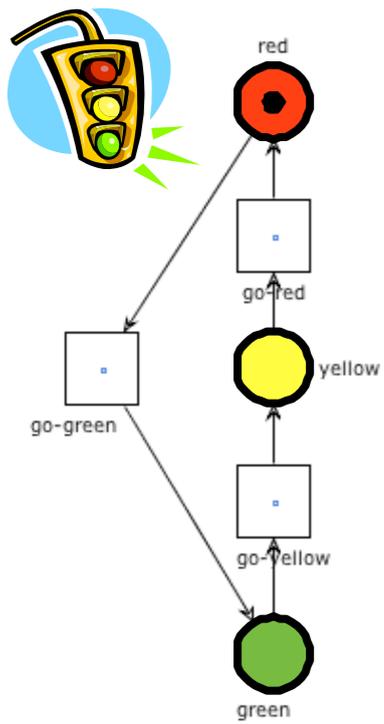
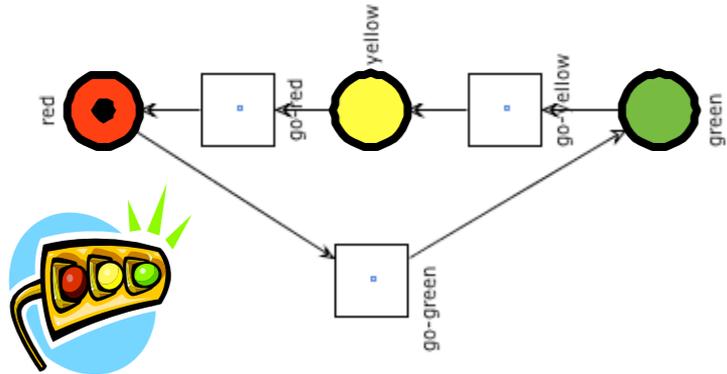
(we omit arc labels for readability issues)

# Example: two traffic lights



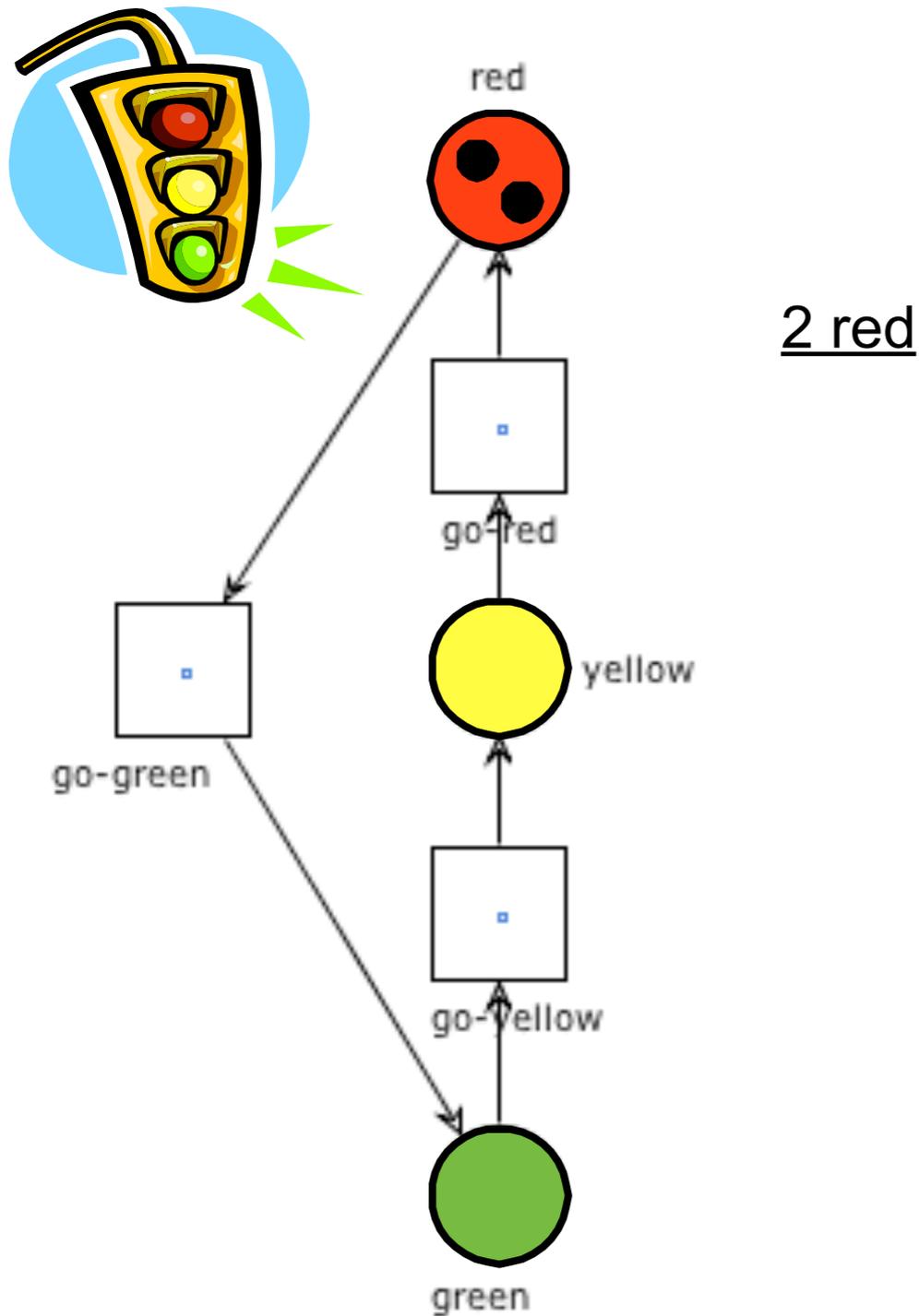
(we omit arc labels for readability issues)

# Example: two traffic lights

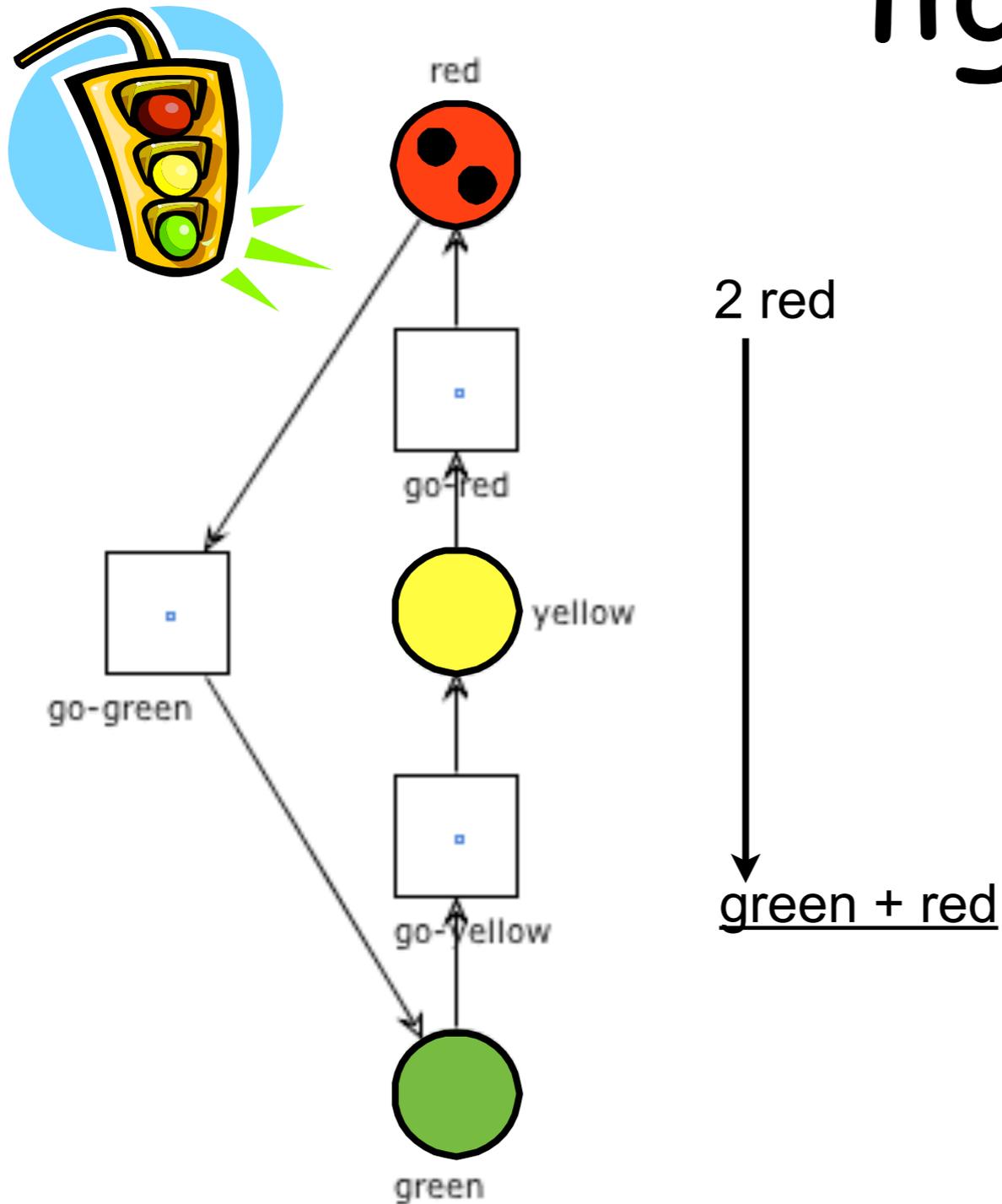


(we omit arc labels for readability issues)

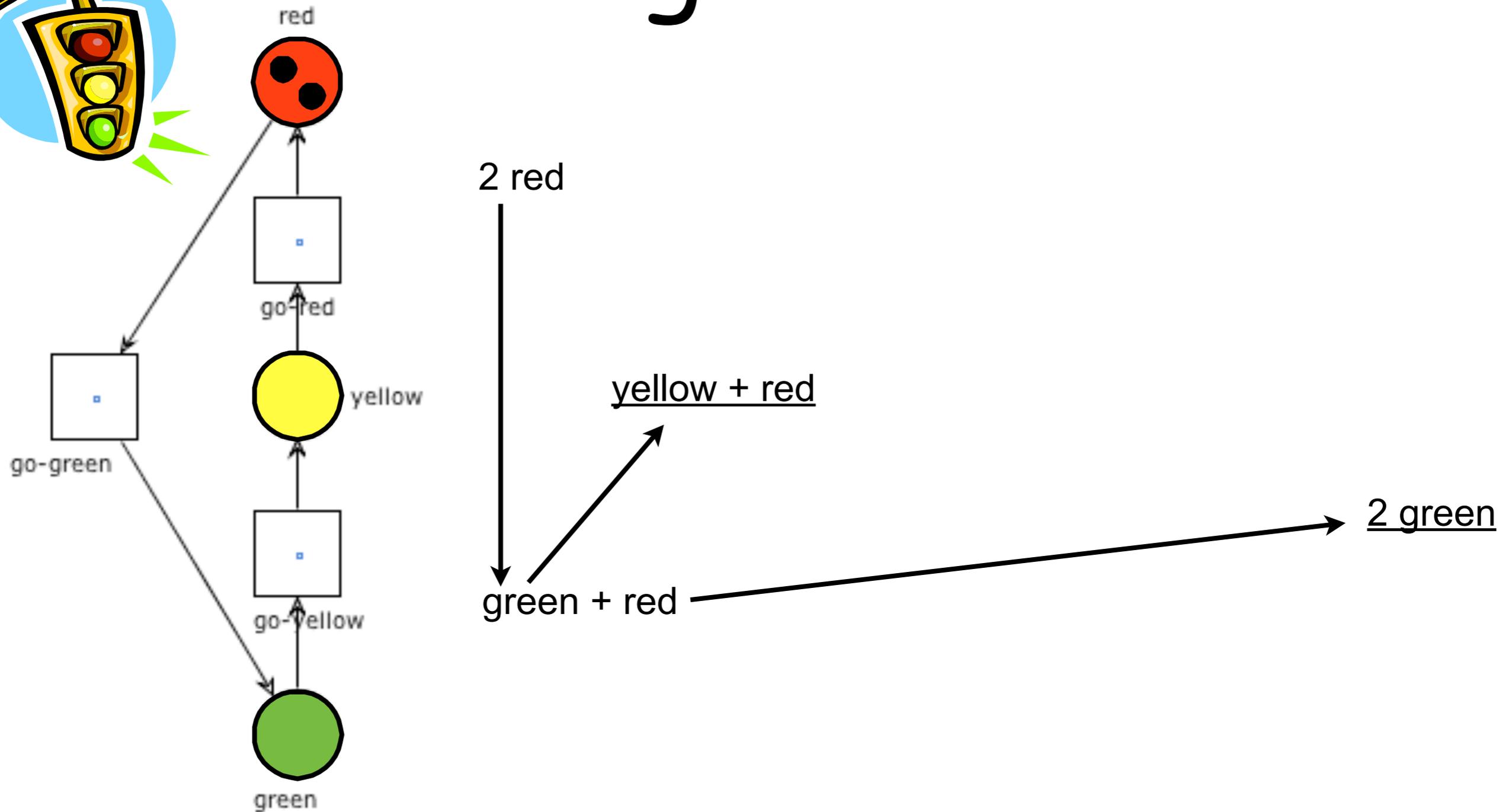
# Example: two traffic lights



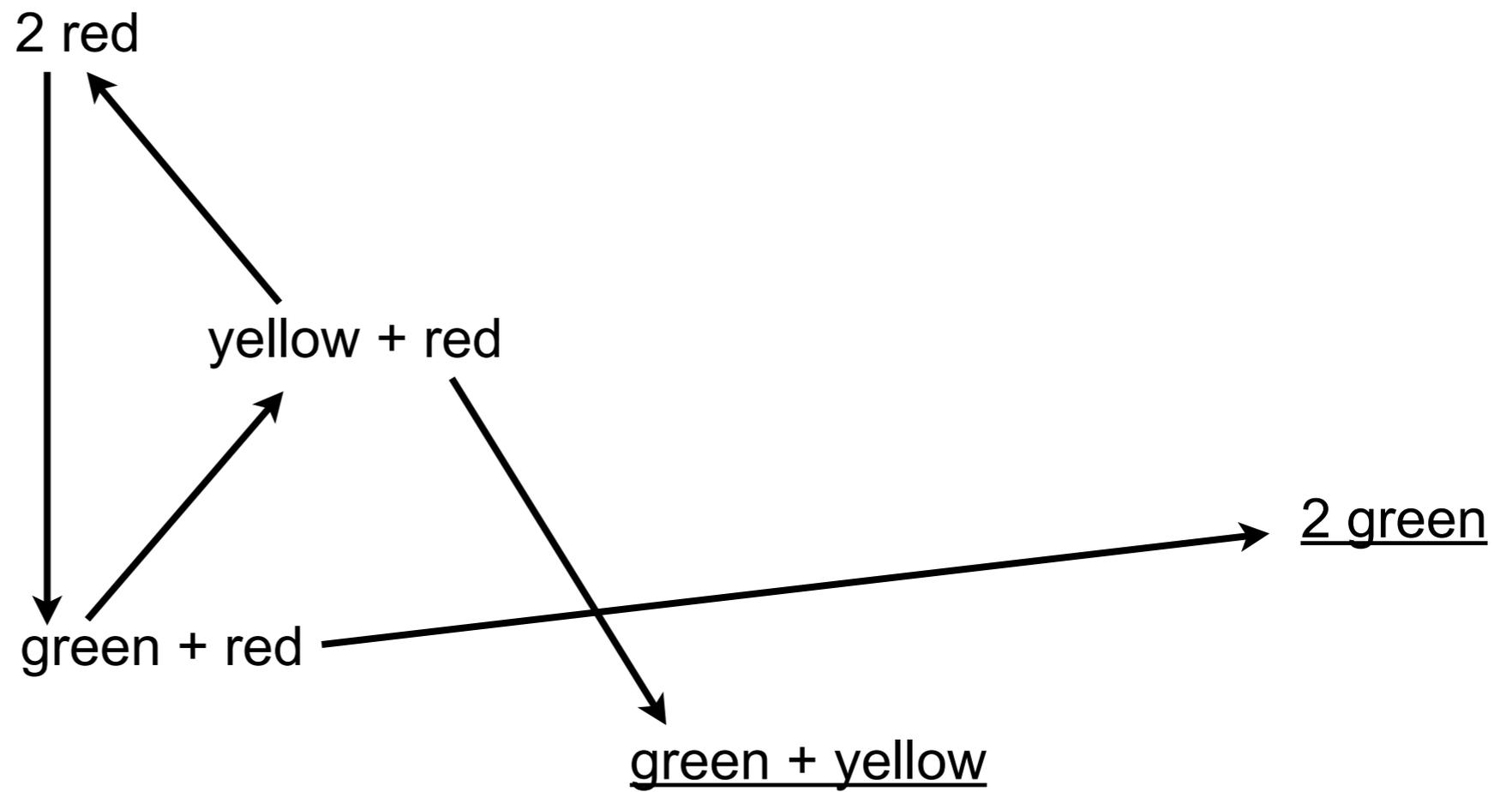
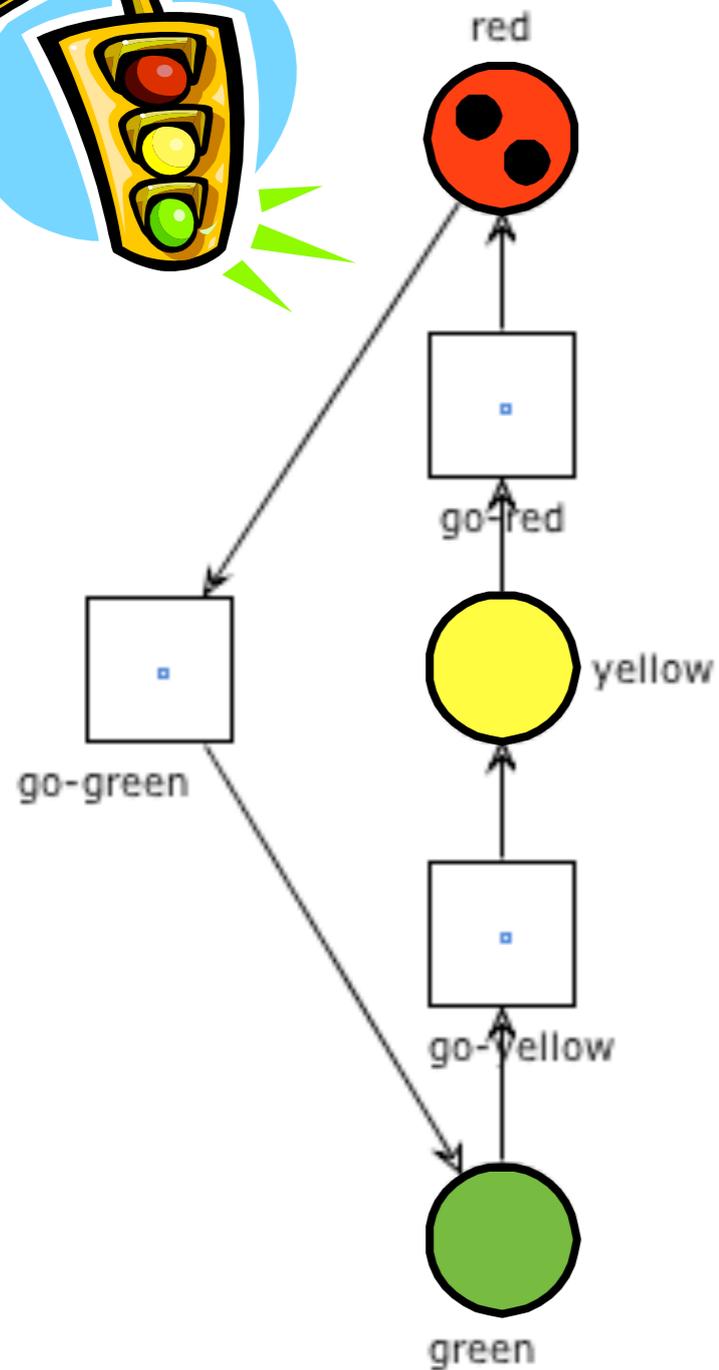
# Example: two traffic lights



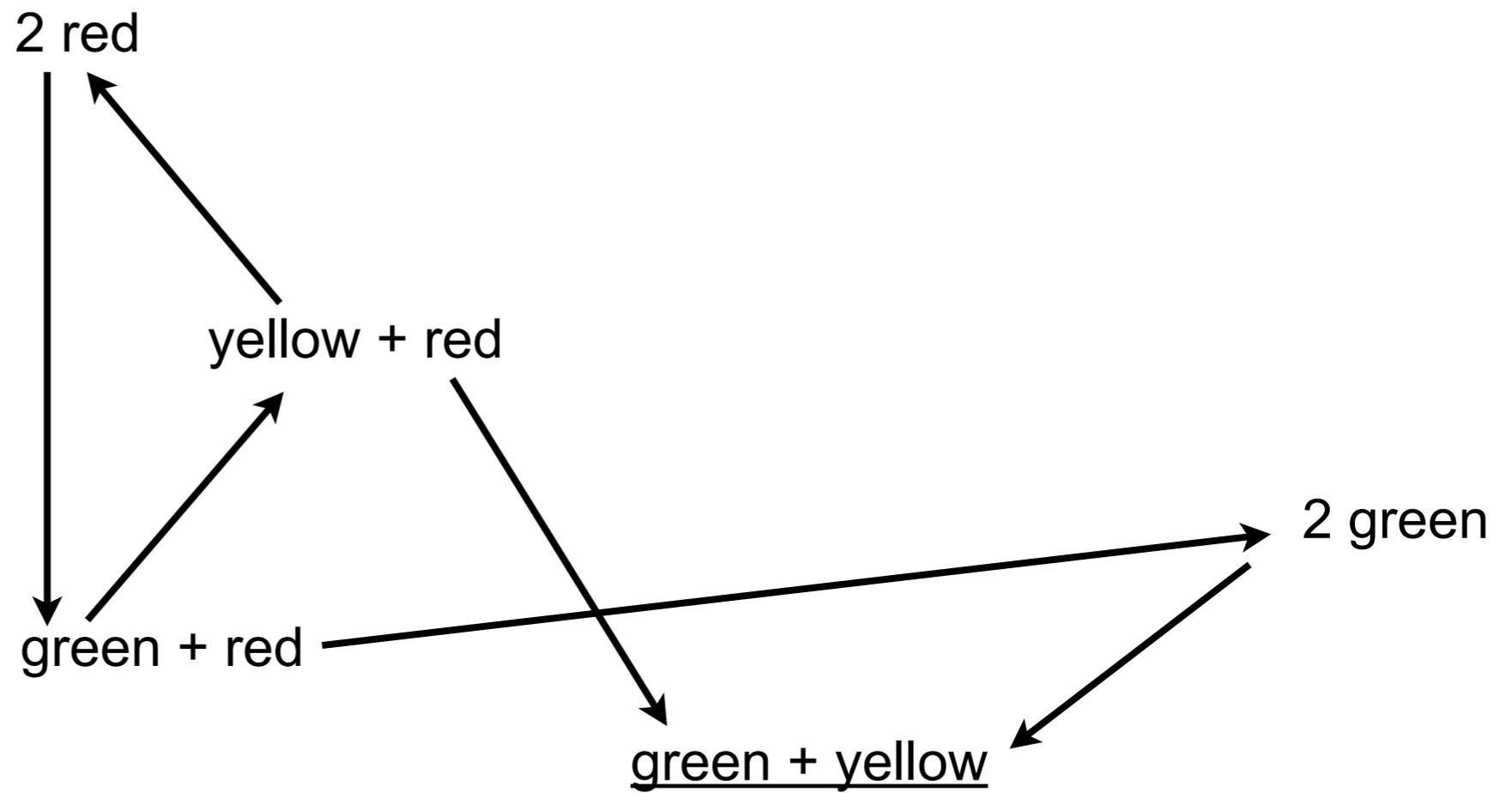
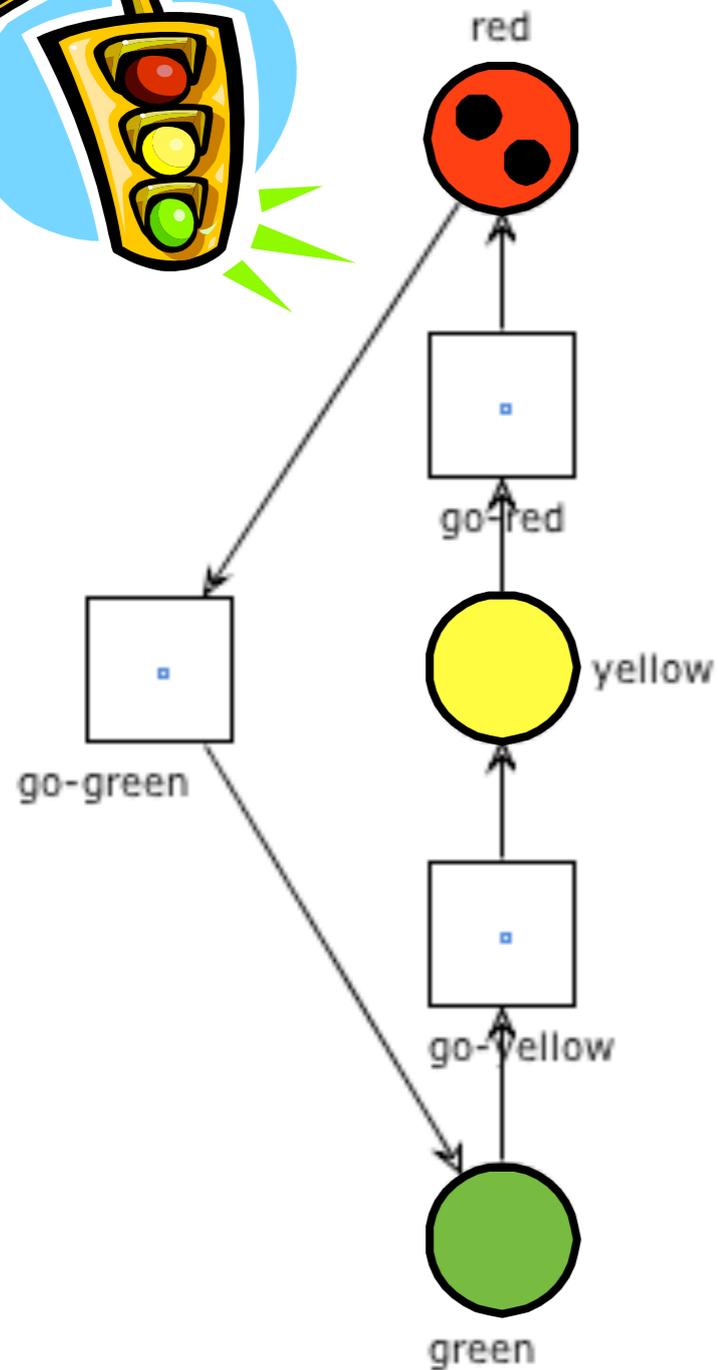
# Example: two traffic lights



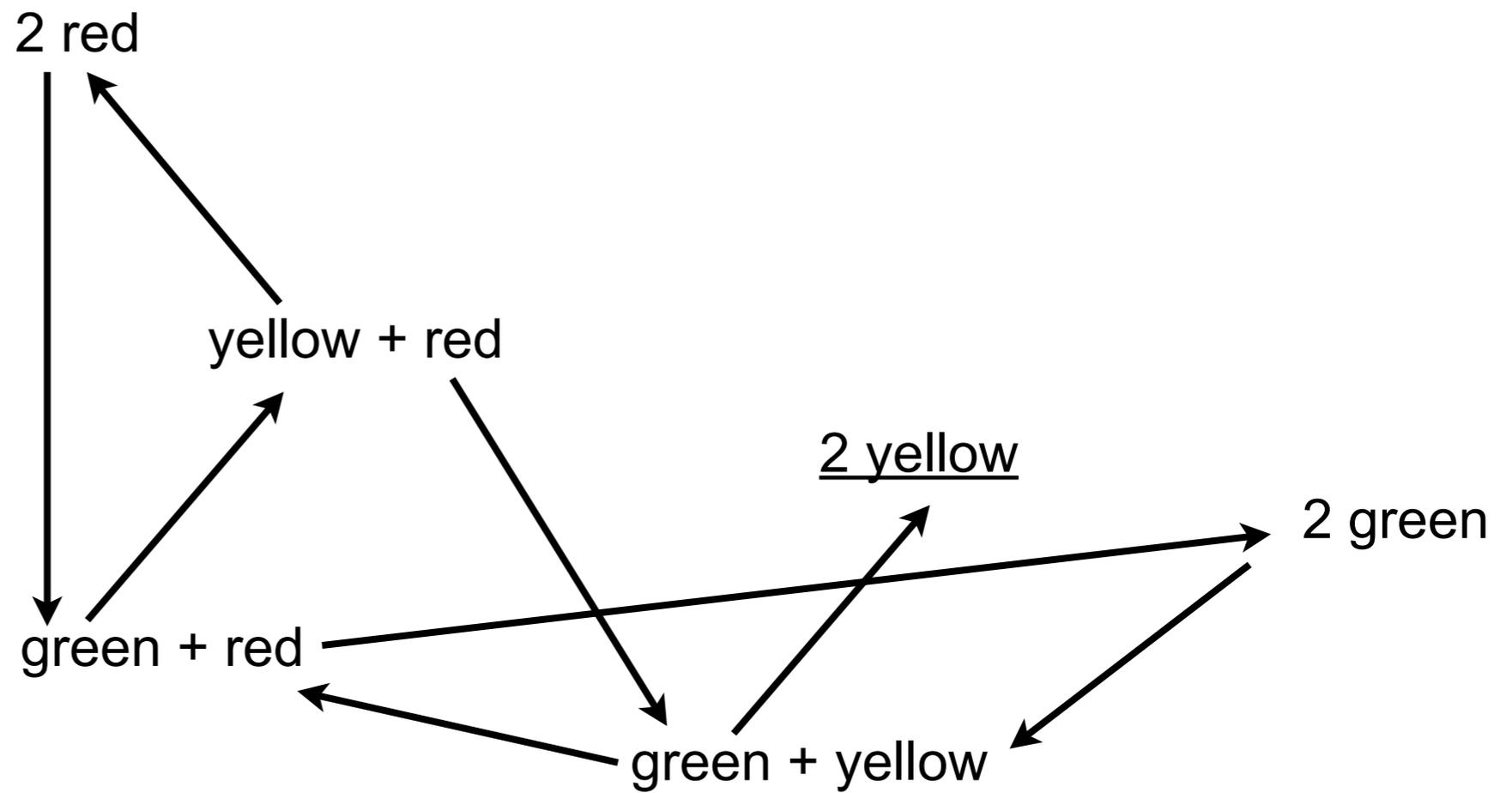
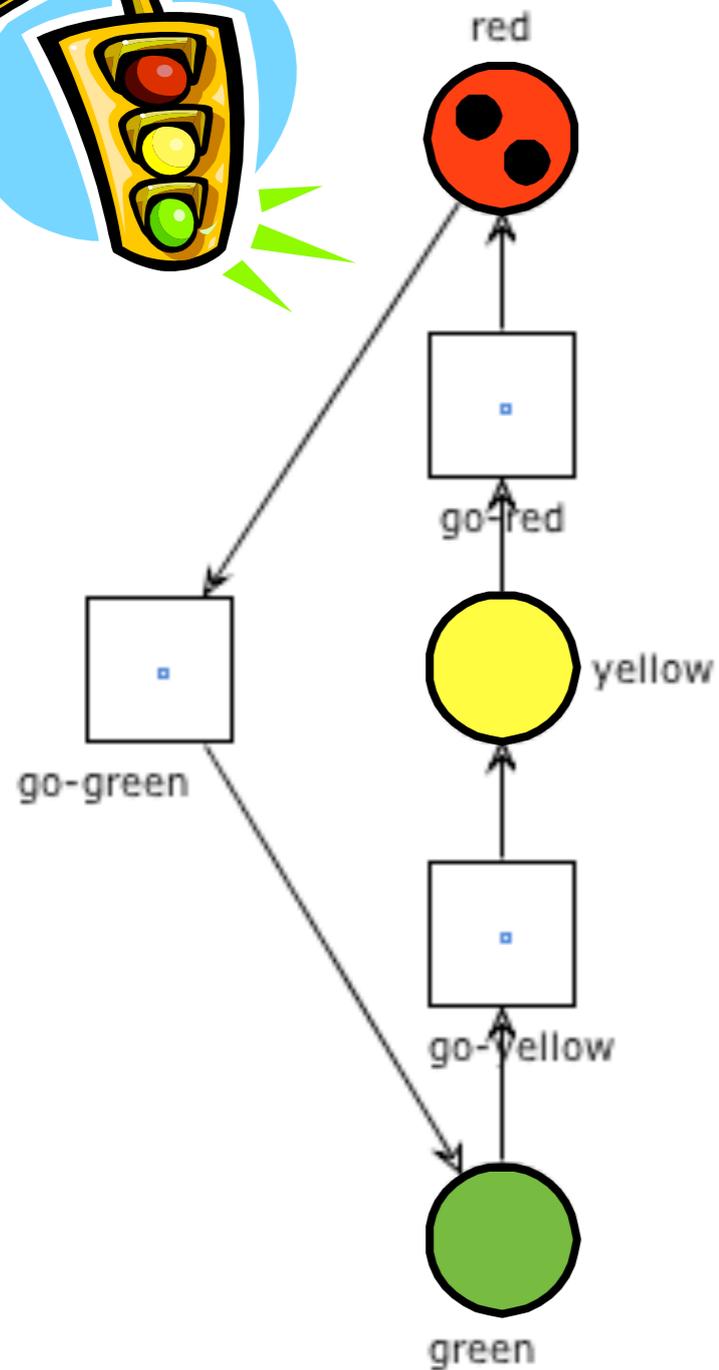
# Example: two traffic lights



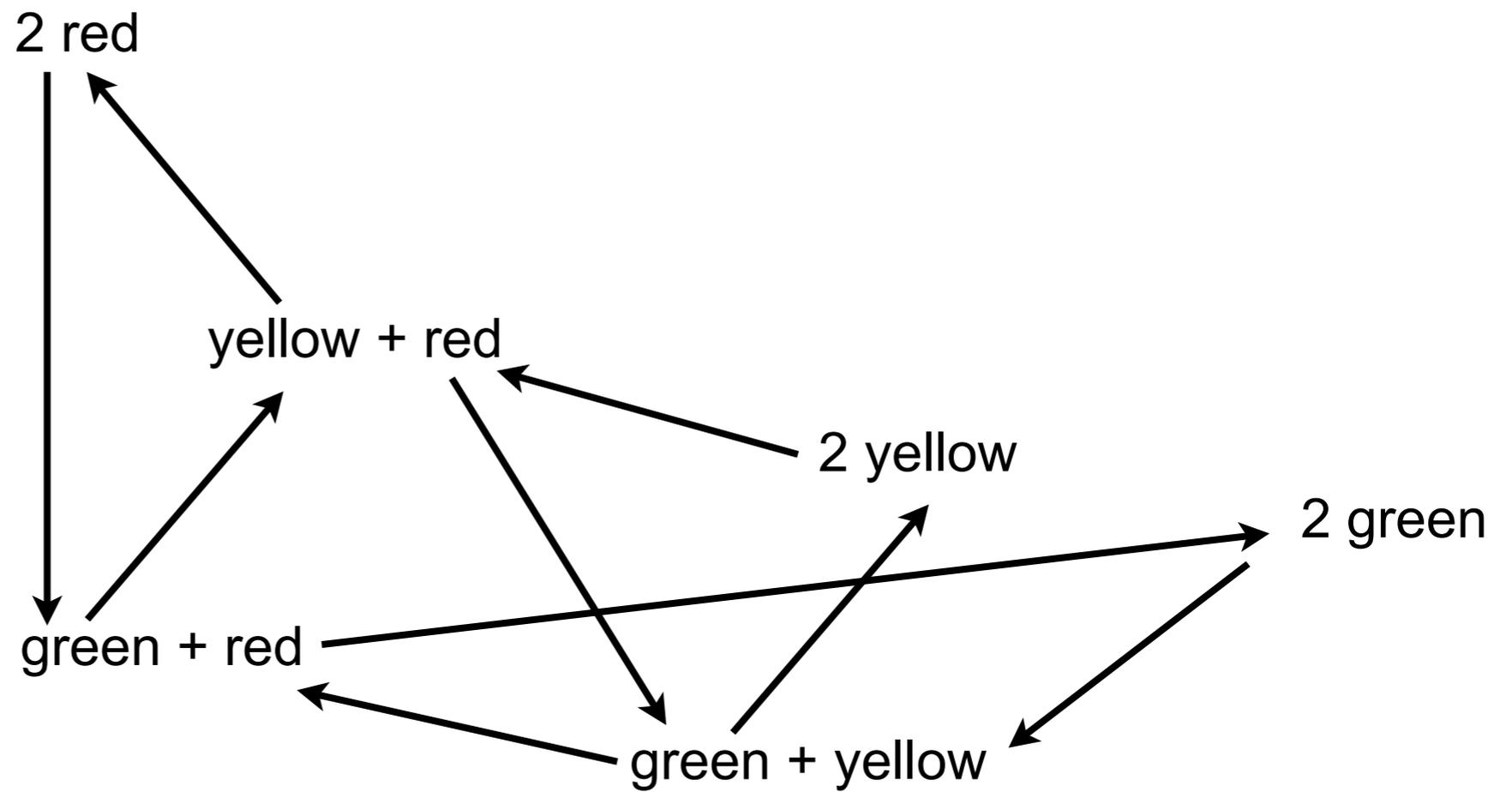
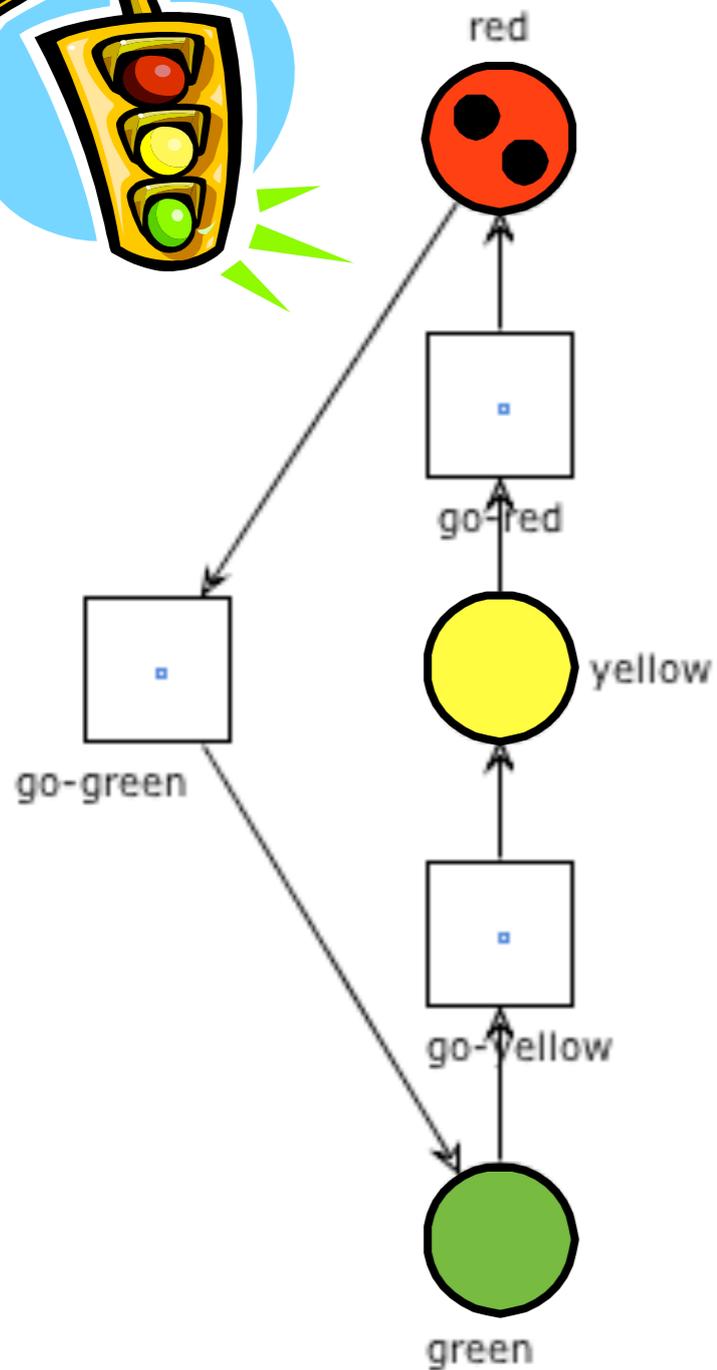
# Example: two traffic lights



# Example: two traffic lights

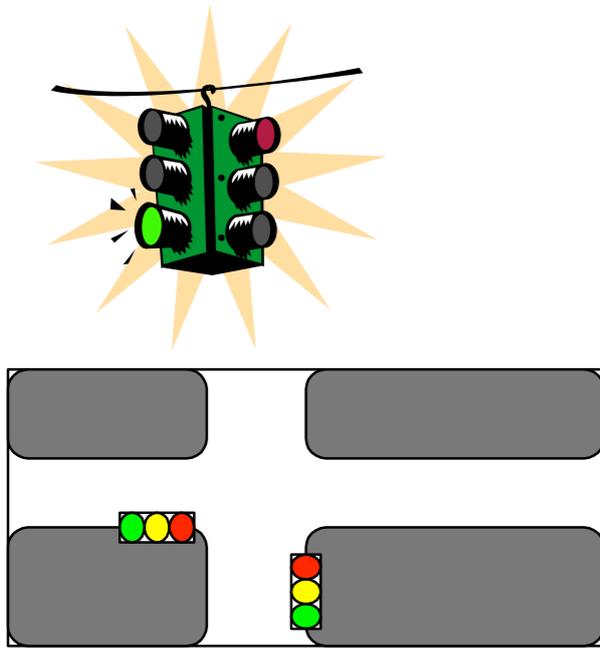


# Example: two traffic lights

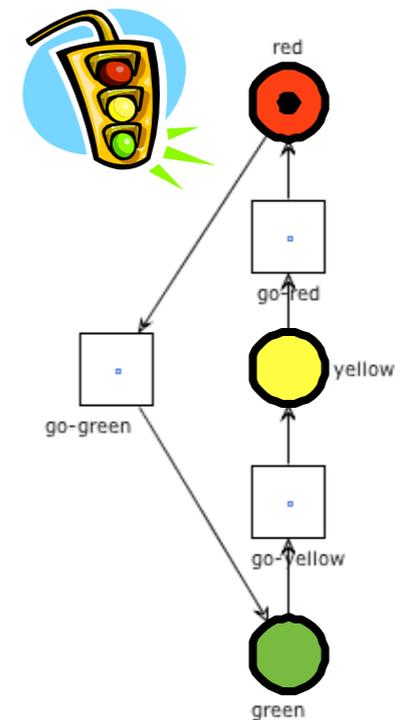
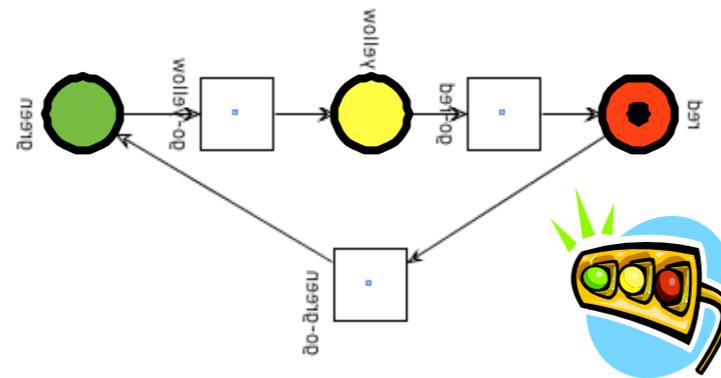




# Question time



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

On the web (Petri net applet):

[http://wwwis.win.tue.nl/~wvdaalst/workflowcourse/pn\\_applet/pn\\_applet.htm](http://wwwis.win.tue.nl/~wvdaalst/workflowcourse/pn_applet/pn_applet.htm)

On your PC (Workflow Petri net Designer):

<http://www.woped.org>

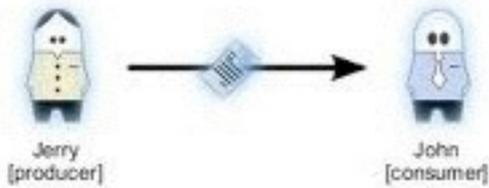
# Exercise:

## 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**.

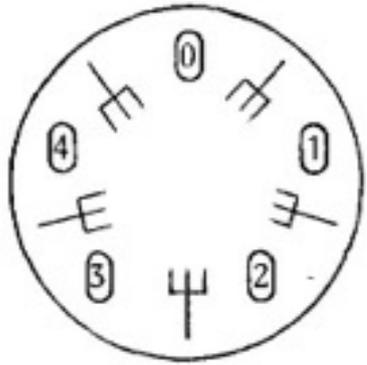
Each one alternates between these two states

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 the 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?

# Properties of Petri nets

We describe, in an informal way, some of the properties of Petri nets that can play an important role in the verification of business processes

Liveness  
Deadlock-freedom  
Boundedness  
Cyclicity (also Reversibility)

# Liveness

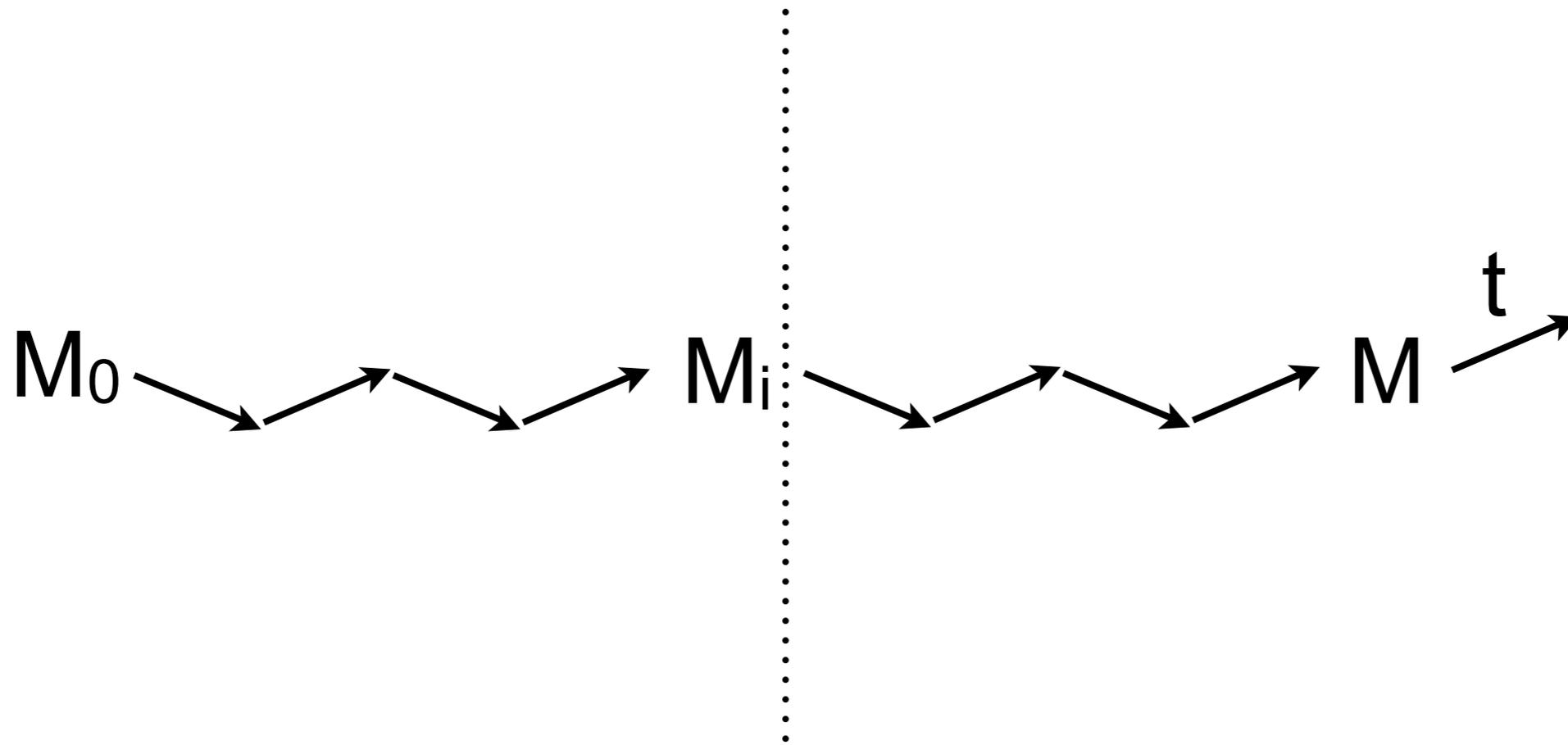
A transition  $t$  is **live**, if from any reachable marking  $M$  another marking  $M'$  can be reached where  $t$  is enabled

In other words, at any point in time of the computation, we cannot exclude that  $t$  will fire in the future

A Petri net is **live** if all of its transitions are live

# Liveness illustrated

For any reachable marking  $M_i$



Can we find a way to enable  $t$ ?

# Liveness, formally

$$(P, T, F, M_0)$$

$$\forall t \in T, \quad \forall M \in [M_0 \rangle, \quad \exists M' \in [M \rangle, \quad M' \xrightarrow{t}$$

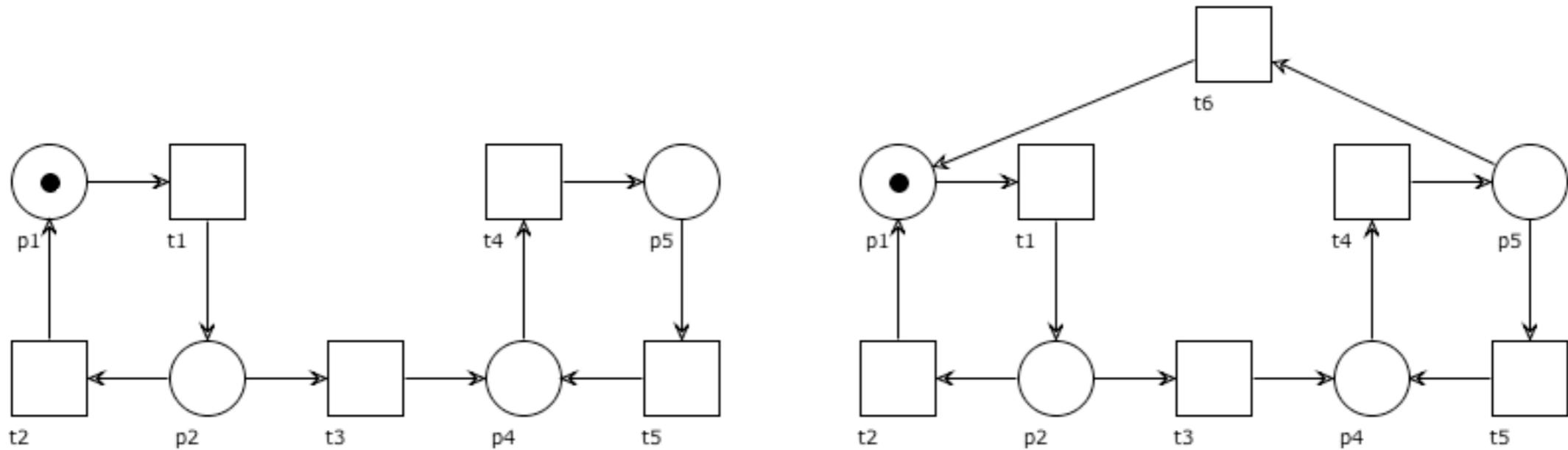
# Liveness: pay attention!

Liveness should not be confused with the following property:

"starting from the initial marking  $M_0$  it is possible to reach a marking  $M$  that enables  $t$ "

(this property just ensures that  $t$  is not "dead" in  $M_0$ )

# Liveness: example



Which transitions are live?  
Which are not?  
Is the net live?

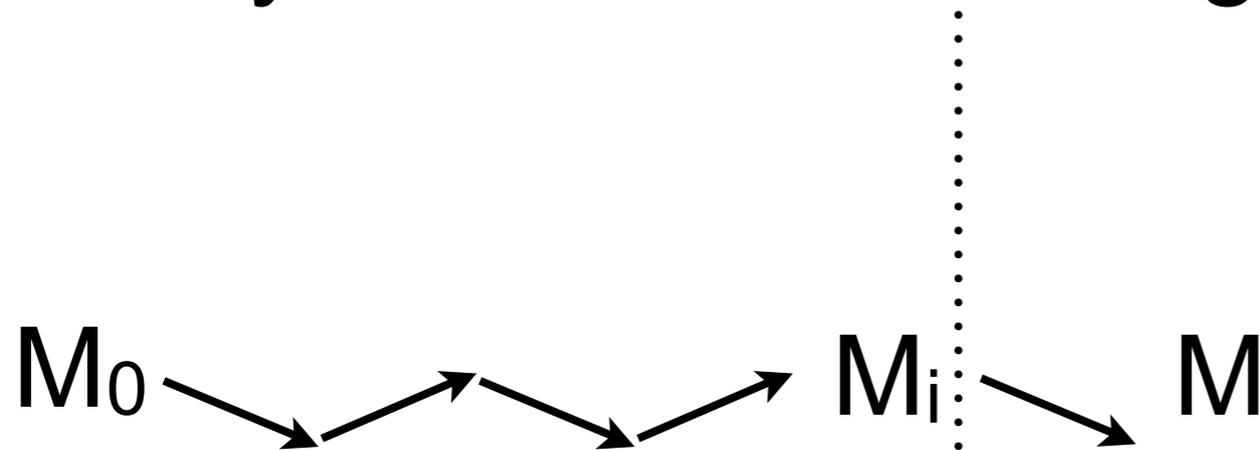
# Deadlock-freedom

A Petri net is **deadlock free**, if every reachable marking enables some transition

In other words, we are guaranteed that at any point in time of the computation, some transition can be fired

# Deadlock-freedom illustrated

For any reachable marking  $M_i$



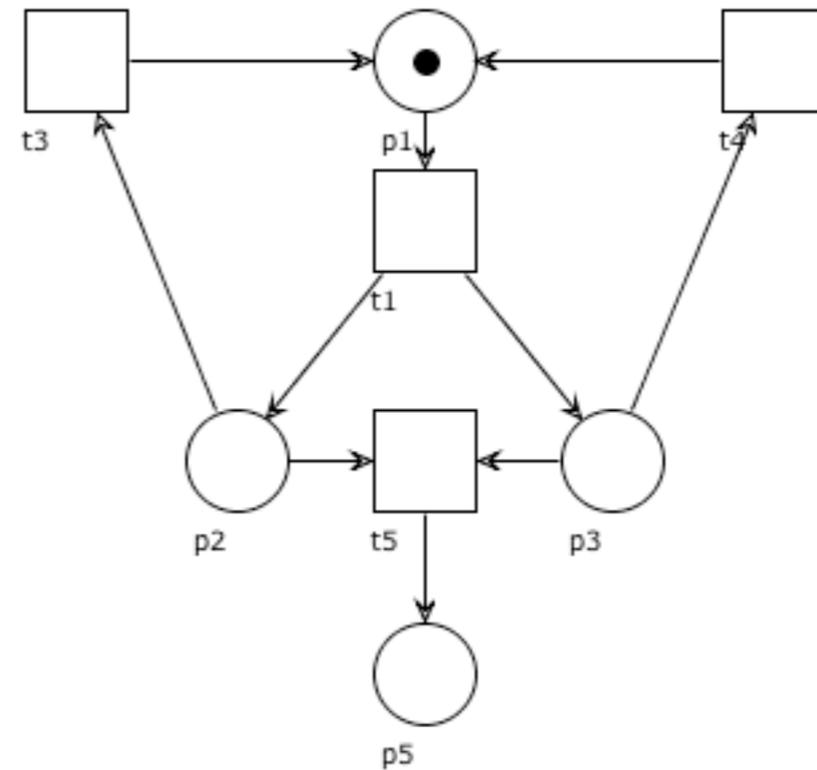
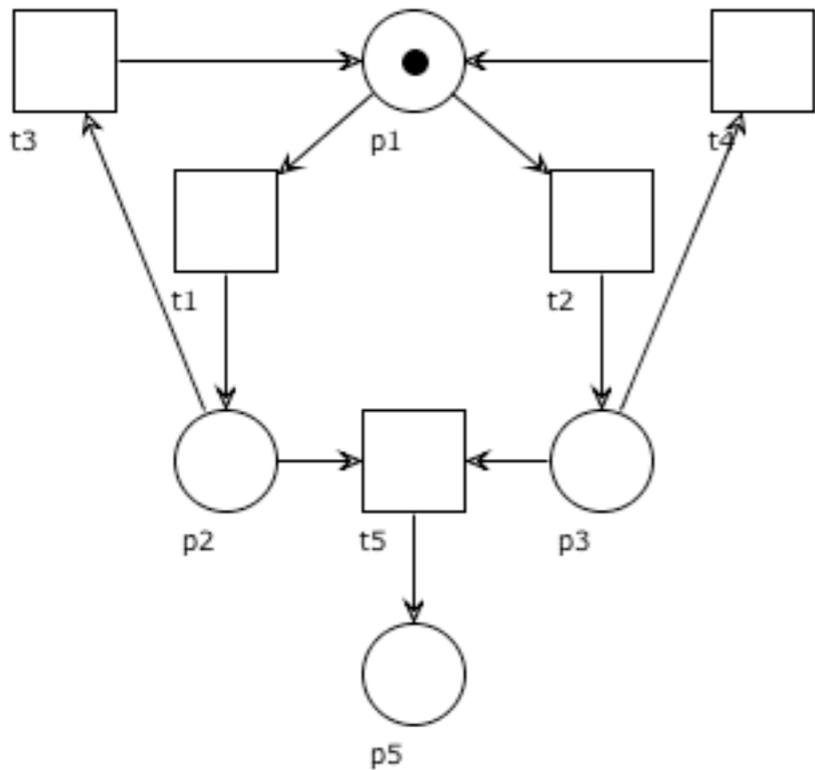
Can we fire some transition?

# Deadlock freedom, formally

$(P, T, F, M_0)$

$\forall M \in [M_0 \rangle, \quad \exists t \in T, \quad M \xrightarrow{t}$

# Deadlock-freedom: example



Is the net deadlock-free?

# Question time

Does liveness imply deadlock-freedom?

(Can you exhibit a live Petri net that is not deadlock-free?)

Does deadlock-freedom imply liveness?

(Can you exhibit a deadlock-free net that is not live?)

# Liveness implies deadlock freedom

**Lemma** If  $(P, T, F, M_0)$  is live, then it is deadlock-free

By contradiction, let  $M \in [M_0 \rangle$ , with  $M \not\rightarrow$

Let  $t \in T$  ( $T$  cannot be empty).

By liveness,  $\exists M' \in [M \rangle$  with  $M' \xrightarrow{t}$ .

Since  $M$  is dead,  $[M \rangle = \{M\}$ .

Therefore  $M = M' \xrightarrow{t}$ , which is absurd.

# $k$ -Boundedness

Let  $k$  be a natural number

A place  $p$  is  **$k$ -bounded** if no reachable marking has more than  $k$  tokens in place  $p$

A net is  **$k$ -bounded** if all of its places are  $k$ -bounded

In other words, if a net is  $k$ -bounded, then  $k$  is a capacity constraint that can be imposed over places without any risk of causing “overflow”

# Safe nets

A place  $p$  is **safe** if it is 1-bounded

A net is **safe** if all of its places are safe

In other words, if the net is safe, then we know that, in any reachable marking, each place contains one token at most

# Boundedness

A place  $p$  is **bounded** if it is  $k$ -bounded for some natural number  $k$

A net is **bounded** if all of its places are bounded

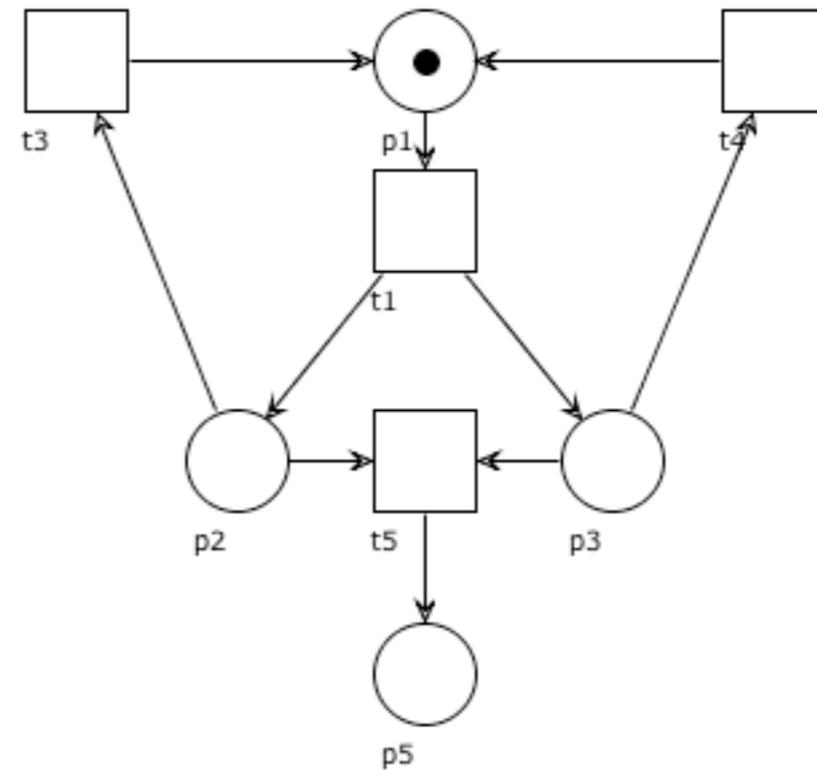
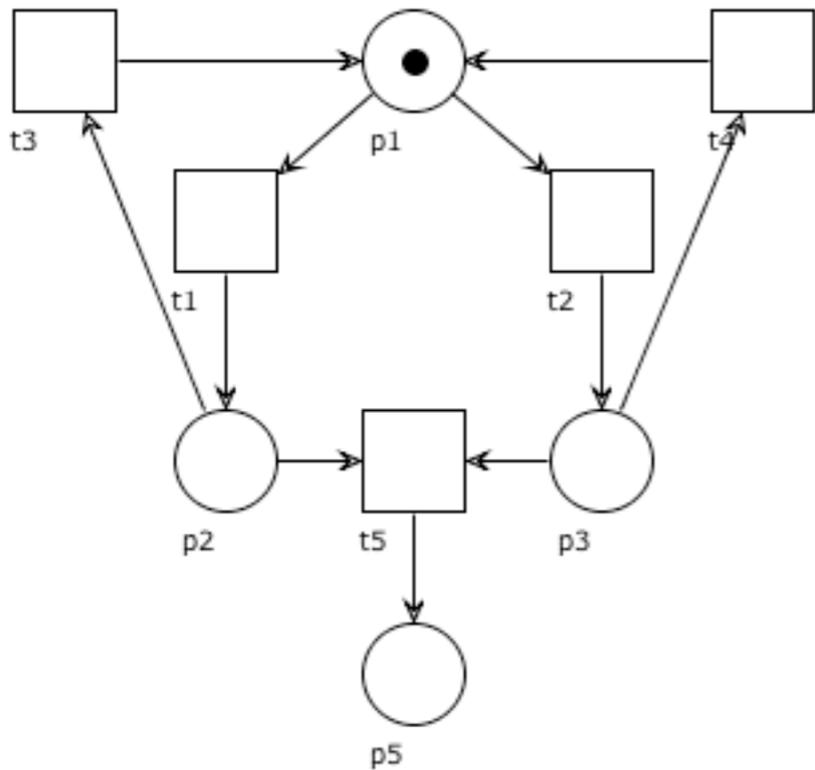
A net is **unbounded** if it is not bounded

# Boundedness, formally

$$(P, T, F, M_0)$$

$$\exists k \in \mathbb{N}, \quad \forall M \in [M_0 \rangle, \quad \forall p \in P, \quad M(p) \leq k$$

# Boundedness: example



Which places are bounded?

Is the net bounded?

Which places are safe?

Is the net safe?

# A puzzle about reachability

**Theorem:** If a system is... then its reachability  
graph is finite

**Theorem:** A system is... iff its reachability graph is  
finite

(fill the dots and the proofs)

# A puzzle about boundedness

**Theorem:** If a system is  $k$ -bounded then any reachable marking contains a number of tokens less than or equal to ...

**Theorem:** If a system is safe then any reachable marking contains a number of tokens less than or equal to ...

(fill the dots and the proofs)

# Cyclicity (aka Reversibility)

A marking  $M$  is a **home marking** if it can be reached from every reachable marking

A net is **cyclic** (or **reversible**) if its initial marking is a home marking

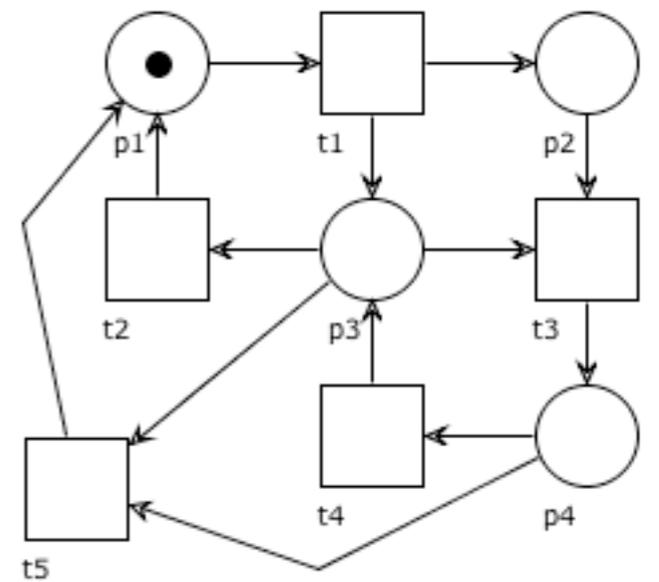
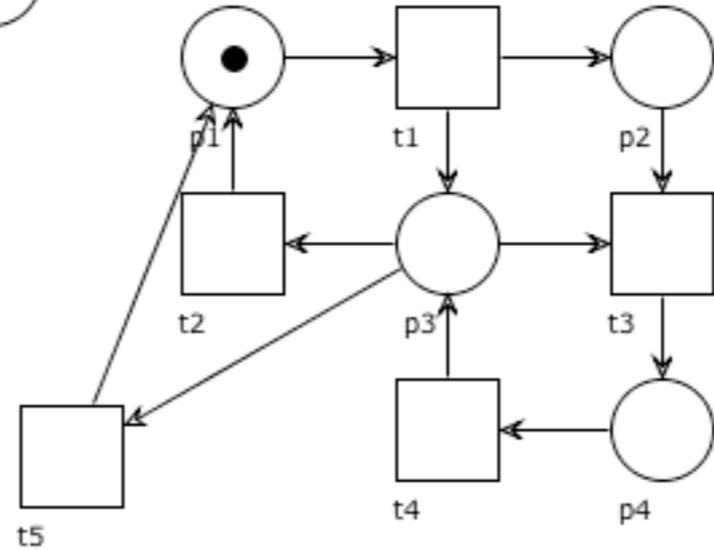
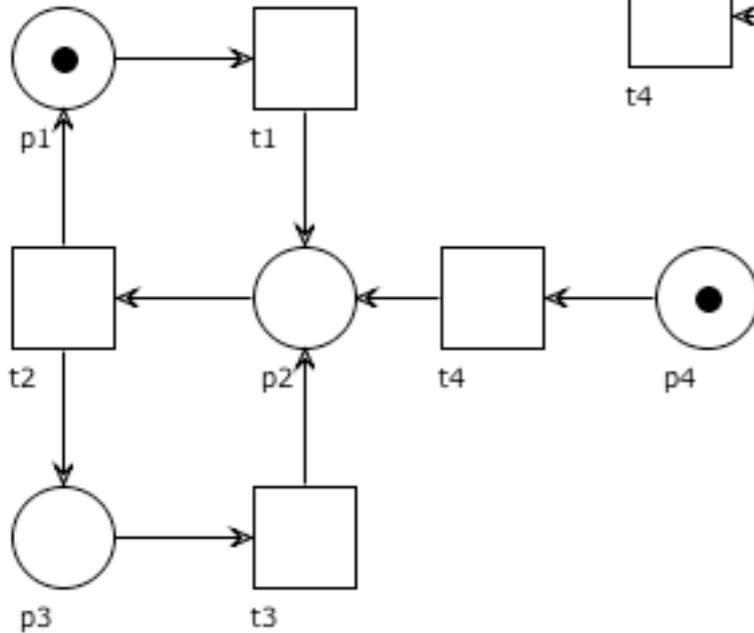
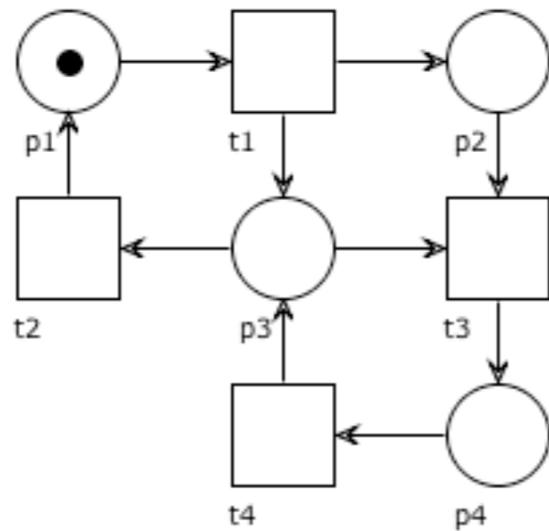
# Orthogonal properties

Liveness, boundedness and cyclicity are independent of each other

In other words, you can find nets that satisfy any arbitrary combination of the above three properties (and not the others)

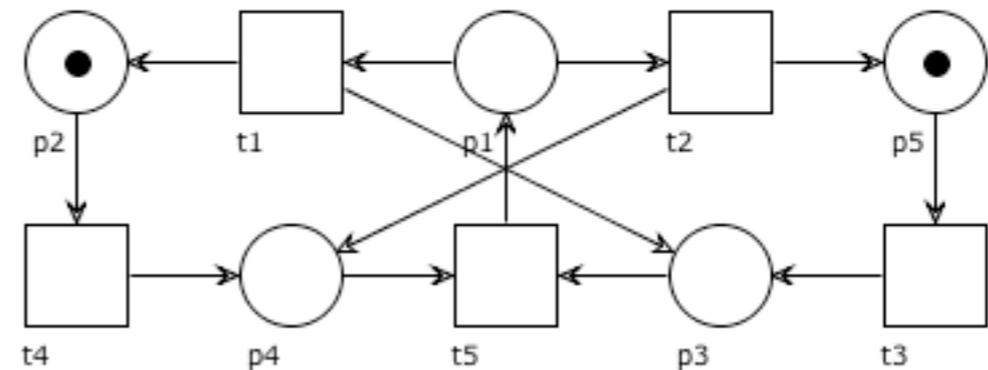
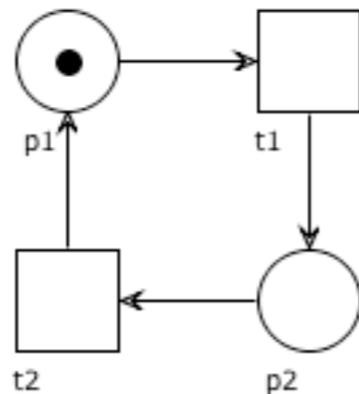
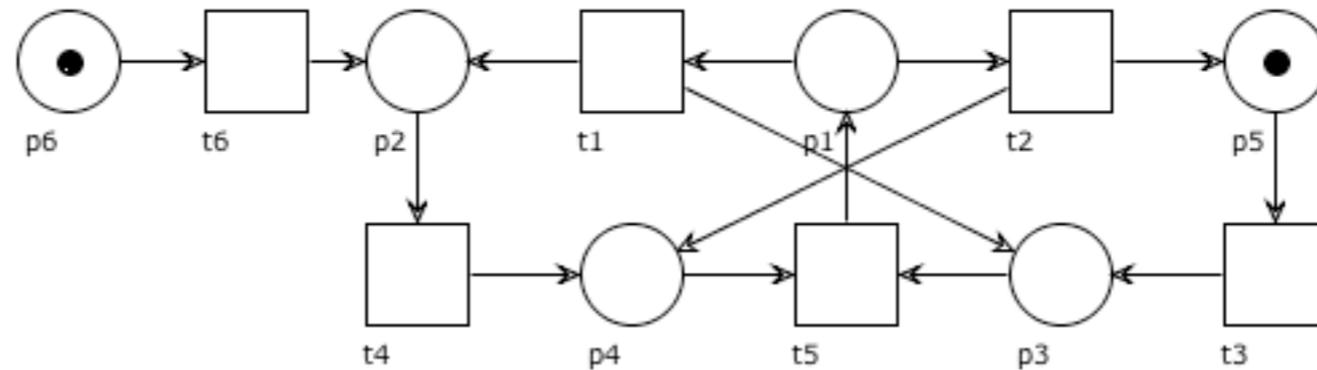
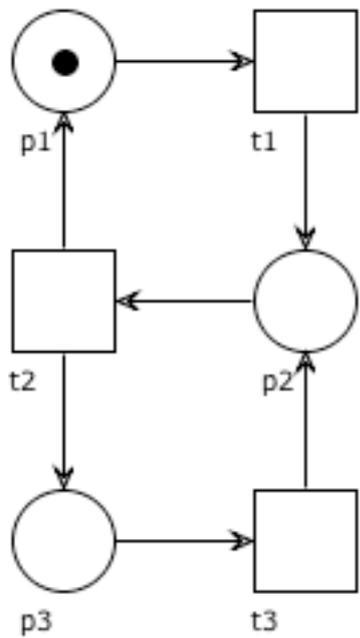
# Exercises

For each of the following nets, say if they are live, deadlock-free, bounded, safe, cyclic



# Exercises

For each of the following nets, say if they are live, deadlock-free, bounded, safe, cyclic



Live and dead places

# Live place, intuitively

A place  $p$  is live

if every time it becomes unmarked

there is still the possibility to be marked in the future

(or if it is always marked)

# Live place

**Definition:** Let  $(P, T, F, M_0)$  be a net system.

A place  $p \in P$  is **live** if  $\forall M \in [M_0 \rangle. \exists M' \in [M \rangle. M'(p) > 0$

# Place liveness

## **Definition:**

A net system  $(P, T, F, M_0)$  is **place-live** if every place  $p \in P$  is live

# Liveness implies place-liveness

**Proposition:** Live systems are place-live

Take any  $p$  and any  $t \in \bullet p \cup p \bullet$

Let  $M \in [M_0 \rangle$

By liveness: there is  $M', M'' \in [M \rangle$  s.t.  $M' \xrightarrow{t} M''$

Then  $M'(p) > 0$  or  $M''(p) > 0$

# Dead nodes

**Definition:** Let  $(P, T, F)$  be a net system.

A transition  $t \in T$  is **dead** at  $M$  if  $\forall M' \in [M \rangle. M' \not\xrightarrow{t}$

A place  $p \in P$  is **dead** at  $M$  if  $\forall M' \in [M \rangle. M'(p) = 0$

# Some obvious facts

If a system is not live, it has a transition dead at some reachable marking

If a system is not place-live, it has a place dead at some reachable marking

If a place / transition is dead at  $M$ , then it remains dead at any marking reachable from  $M$   
(the set of dead nodes can only increase during a run)

Every transition in the pre- or post-set of a dead place is also dead

# Behavioural vs Structural Properties

# Structural properties

All the properties we have seen so far are  
**behavioural (or dynamic)**

(i.e. they depend on the initial marking and firing rules)

It is sometimes interesting to connect them to  
**structural** properties

(i.e. the shape of the graph representing the net)

This way we can give **structural characterization** of  
behavioural properties for a class of nets  
(computationally less expensive to check)

# A matter of terminology

To better reflect the above distinction, it is frequent:

to use the term **net system** for denoting a Petri net  
with a given initial marking  
(we study behavioural properties of systems)

to use the term **net** for denoting a Petri net without  
specifying any initial marking  
(we study structural properties of nets)

# Paths and circuits

A **path** of a net  $(P, T, F)$  is a non-empty sequence  $x_1x_2\dots x_k$  such that

$$(x_i, x_{i+1}) \in F \quad \text{for every } 1 \leq i < k$$

(and we say that it leads from  $x_1$  to  $x_k$ )

A path from  $x$  to  $y$  is called a **circuit** if:

no element occurs more than once in it and  $(y, x) \in F$

(since for any  $x$  we have  $(x, x) \notin F$ , hence a circuit involves at least two nodes)

# Connectedness

A net  $(P, T, F)$  is **weakly connected** iff it does not fall into (two or more) unconnected parts (i.e. no two subnets  $(P_1, T_1, F_1)$  and  $(P_2, T_2, F_2)$  with disjoint and non-empty sets of elements can be found that partition  $(P, T, F)$ )

A weakly connected net is **strongly connected** iff for every arc  $(x, y)$  there is a path from  $y$  to  $x$

# Connectedness, formally

A net  $(P, T, F)$  is **weakly connected** if every two nodes  $x, y$  satisfy

$$(x, y) \in (F \cup F^{-1})^*$$

(i.e. if there is an undirected path from  $x$  to  $y$ )

It is **strongly connected** if  $(x, y) \in F^*$

# A note

In the following we will consider (implicitly) weakly connected nets only

(if they are not, then we can study each of their subsystems separately)

# S-systems

A Petri net is called **S-system** if every transition has one input place and one output place  
(S comes from *Stellen*, the German word for place)

This way any synchronization is ruled out

The theory of S-systems is very simple

# T-systems

A Petri net is called **T-system** if every place has one input transition and one output transition

This way all conflicts are ruled out

T-systems have been studied extensively since the early Seventies

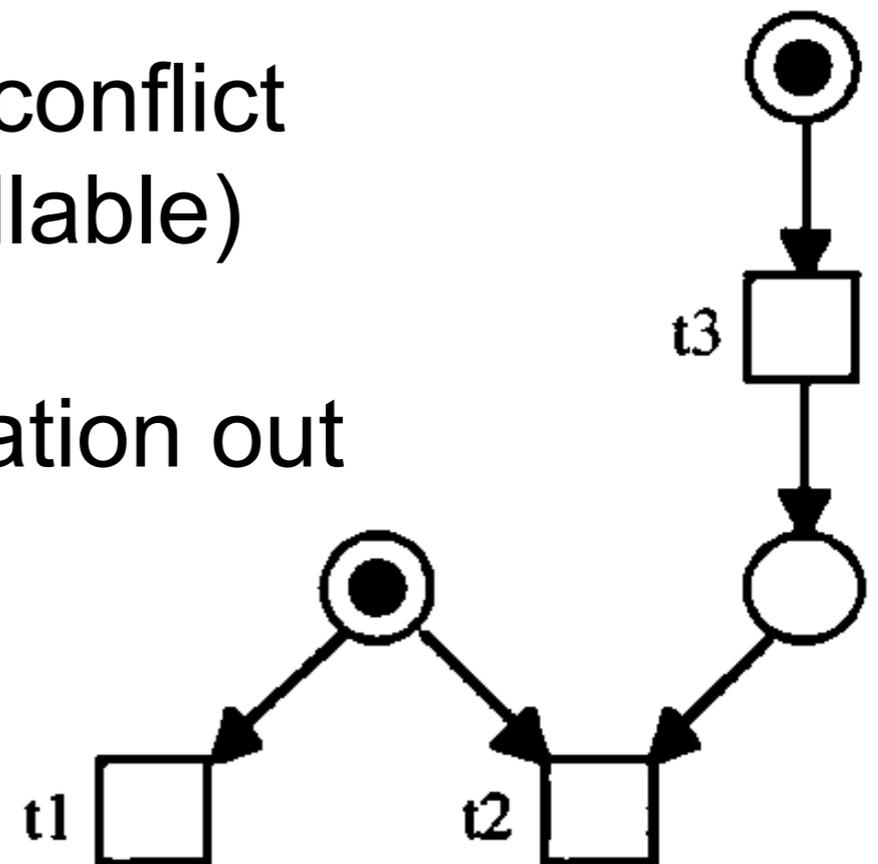
# Interference of conflicts and synch

Typical situation:

initially t1 and t2 are not in conflict

but when t3 fires they are in conflict  
(the firing of t3 is not controllable)

Free-choice nets rule this situation out



# Free-choice nets

The aim is to avoid that a choice between transition is influenced by the rest of the system

Easiest way:

keep places with more than one output transition apart from transitions with more than one input place

In other words, if  $(p,t)$  is an arc, then it means that  $t$  is the only output transition of  $p$  (no conflict)

OR

$p$  is the only input place of  $t$  (no synch)

# Free-choice nets

But we can study a slightly more general class of nets  
by requiring a weaker constraint

A Petri net is **free-choice** if  
whenever there is an arc  $(p,t)$ , then there is an arc  
from any input place of  $t$   
to any output transition of  $p$

# Exercise

Is the net an S-system, a T-system, free-choice?

