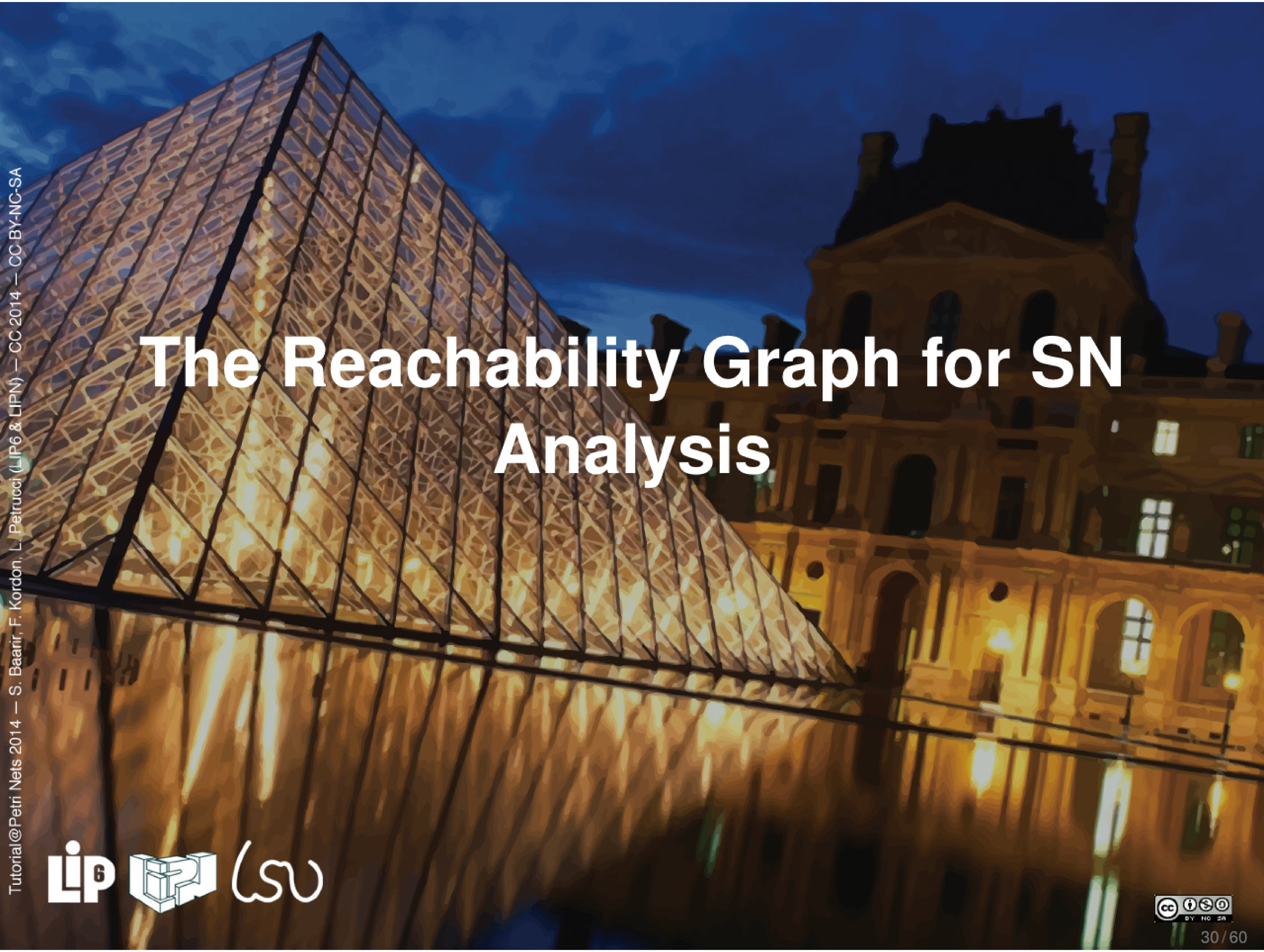
# The Reachability Graph for SN Analysis



You have seen:

- how to model using Symmetric Nets
- a comprehensive detailed example

You have seen:

- how to model using Symmetric Nets
- a comprehensive detailed example

**Let's see the Reachability Graph for analysis**

# Analysis of SNs



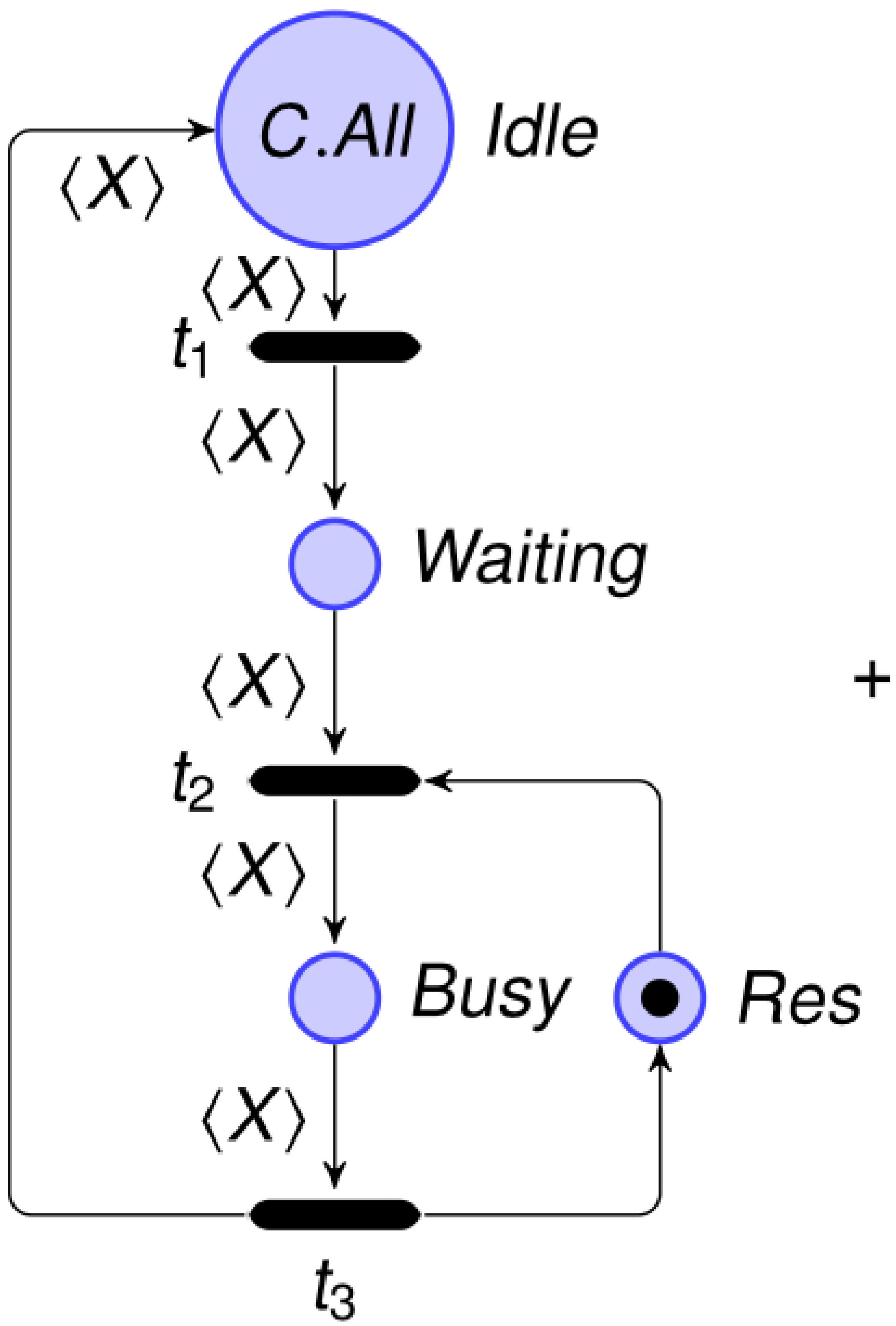
- Does a model conform to the specification ?
- Possibility of answers thanks to:
  - Linear invariants.
  - The reduction theory.
  - The construction of the reachability graph (when the system is finite!)
- Try to take benefits from the structure of the model induced by the colour functions.

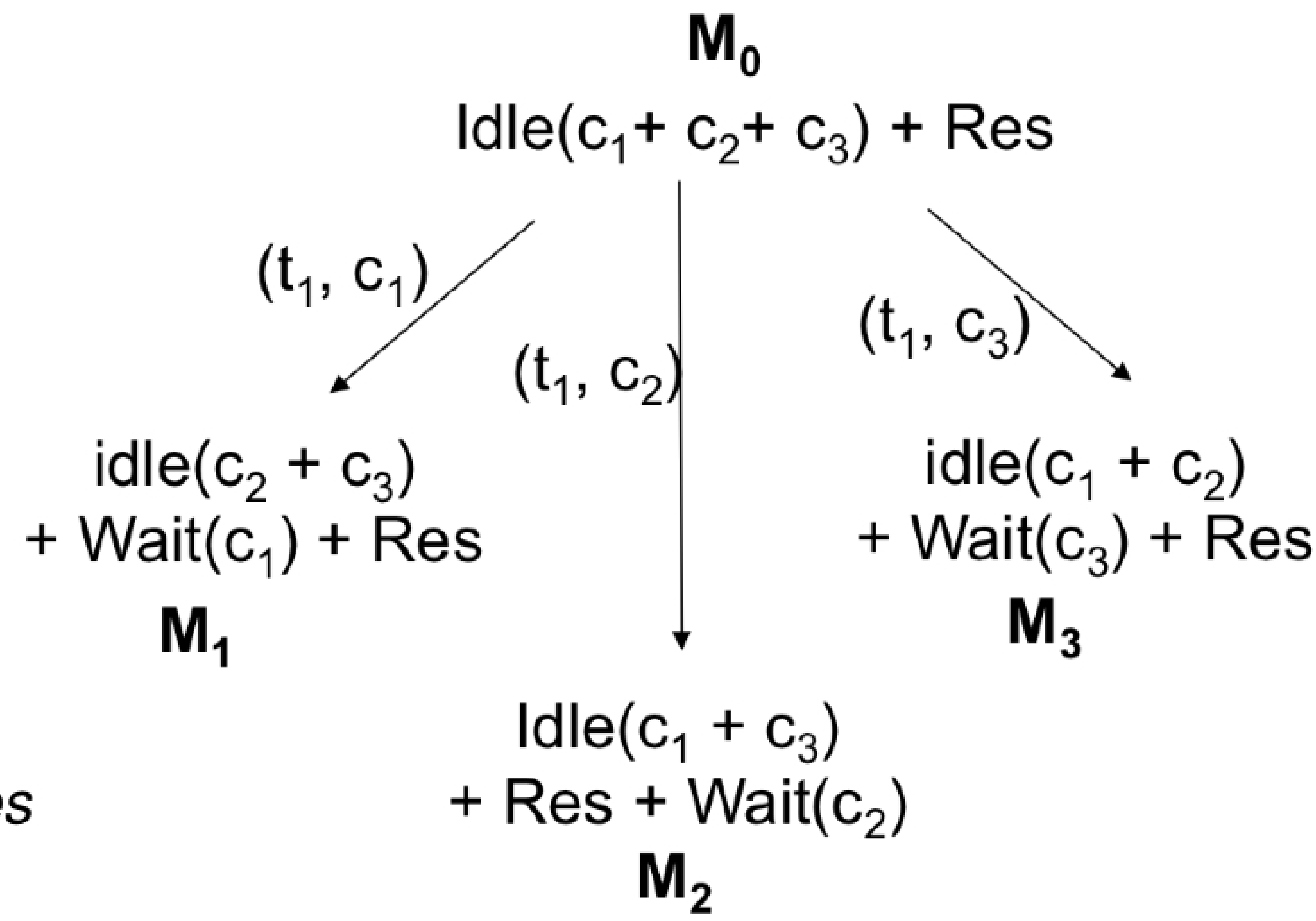
# Reachability Graph (RG) Construction Algorithm

**RG_Construction**$(N = \langle P, T, C, W^-, W^+, M_0\rangle)$
  $RG.Q = \{M_0\}$; $RG.\delta = \emptyset$;
  $RG.q_0 = M_0$; $States = \{M_o\}$:
  **While** (States <> $\emptyset$) {
    s = pick a state in States ;
    States = States \ $\{s\}$;
    **for each** $t \in T, c \in C(t)$ {
      **if** $(s[(t,c)\rangle)$ {
        $s[(t,c)\rangle ns$;
        **if** $(ns \notin RG.Q)$ {
          $RG.Q = RG.Q \cup \{ns\}$ ;
          $States = States \cup \{ns\}$;
        }
        $RG.\delta = RG.\delta \cup \{(s, ns)\}$;
        $RG.\lambda(s, ns) = (t, c)$;
      }
    }
  }
  **return** $RG$;

# Example of RG construction

$C = \{c_1, c_2, c_3\}$



# More advanced verification of finite systems

To operate such verification, we need:

1. A representation of the behaviour of the system.

To operate such verification, we need:

1. A representation of the behaviour of the system.
   $\rightarrow$ **Kripke structure**: in our case, the RG.

# More advanced verification of finite systems



To operate such verification, we need:

1. A representation of the behaviour of the system.
   → **Kripke structure**: in our case, the RG.
   → **State-based propositions:** proposition $p(c)$, where $c \in C(p)$, is true in a state of the RG *iff* place $p$ is marked by colour $c$.

# More advanced verification of finite systems



To operate such verification, we need:

1. A representation of the behaviour of the system.
   → **Kripke structure**: in our case, the RG.
   → **State-based propositions:** proposition $p(c)$, where $c \in C(p)$, is true in a state of the RG *iff* place $p$ is marked by colour $c$.
   → **Event-based propositions:** proposition $t(c)$, where $c \in C(t)$, is true in a state of the RG *iff* transition $t$ is enabled for colour $c$.

# More advanced verification of finite systems

To operate such verification, we need:

1. A representation of the behaviour of the system.
   → **Kripke structure**: in our case, the RG.
   → **State-based propositions:** proposition $p(c)$, where $c \in C(p)$, is true in a state of the RG *iff* place $p$ is marked by colour $c$.
   → **Event-based propositions:** proposition $t(c)$, where $c \in C(t)$, is true in a state of the RG *iff* transition $t$ is enabled for colour $c$.

2. A representation of the specification (property) to be checked.

# More advanced verification of finite systems

To operate such verification, we need:

1. A representation of the behaviour of the system.
   $\rightarrow$ **Kripke structure**: in our case, the RG.
   $\rightarrow$ **State-based propositions:** proposition $p(c)$, where $c \in C(p)$, is true in a state of the RG *iff* place $p$ is marked by colour $c$.
   $\rightarrow$ **Event-based propositions:** proposition $t(c)$, where $c \in C(t)$, is true in a state of the RG *iff* transition $t$ is enabled for colour $c$.

2. A representation of the specification (property) to be checked.
   $\rightarrow$ **Reachability properties:** e.g., for all states of RG, $p(c)$ & $!p(c_1)$; for all states of RG, it exits $t \in T, c \in C(t), t(c)$.

# More advanced verification of finite systems



To operate such verification, we need:

1. A representation of the behaviour of the system.
   → **Kripke structure**: in our case, the RG.
   → **State-based propositions:** proposition $p(c)$, where $c \in C(p)$, is true in a state of the RG *iff* place $p$ is marked by colour $c$.
   → **Event-based propositions:** proposition $t(c)$, where $c \in C(t)$, is true in a state of the RG *iff* transition $t$ is enabled for colour $c$.

2. A representation of the specification (property) to be checked.
   → **Reachability properties:** e.g., for all states of RG, $p(c)$ & $!p(c_1)$; for all states of RG, it exits $t \in T, c \in C(t), t(c)$.
   → **Temporal properties:** two particular semantics are generally admitted:
   - the linear time semantics (an execution of the system is an infinite path in the Kripke structure);
   - the branching time semantics (the execution of the system is represented by the underlying infinite tree).

# Conclusion



At this stage:

- you know how to build a Reachability Graph
- you have seen how it can be used for system analysis

# Conclusion



At this stage:

- you know how to build a Reachability Graph
- you have seen how it can be used for system analysis

**Let's see two logics to express properties (next two sequences)**