

Möbius Tool

LAB 02

Contacts

Maurizio Palmieri

PhD student of the Department of Information Engineering,
University of Pisa

Office: Largo Lucio Lazzarino 1 - 56122 Pisa (PI), Italy

Email: maurizio.palmieri@ing.unipi.it

Overview

- Introduction to SAN in Möbius
- Example of Failure-Detection-Repair
- Exercises

Stochastic Activity Networks (SAN)

The *Stochastic Activity Networks* are a wide-ranging and complex extension to Petri-Nets

Petri Net = places + transitions + enabling conditions + firing rules

Stochastic Petri Net = Petri Net + stochastic transition delay

Stochastic Activity Network = Stochastic Petri Net + stochastic transition outcome + advanced enabling condition + advanced firing rules

SAN in Möbius tool 1/2

NOTE: *the terms **activity**, **transition** and **action** will be used interchangeably*

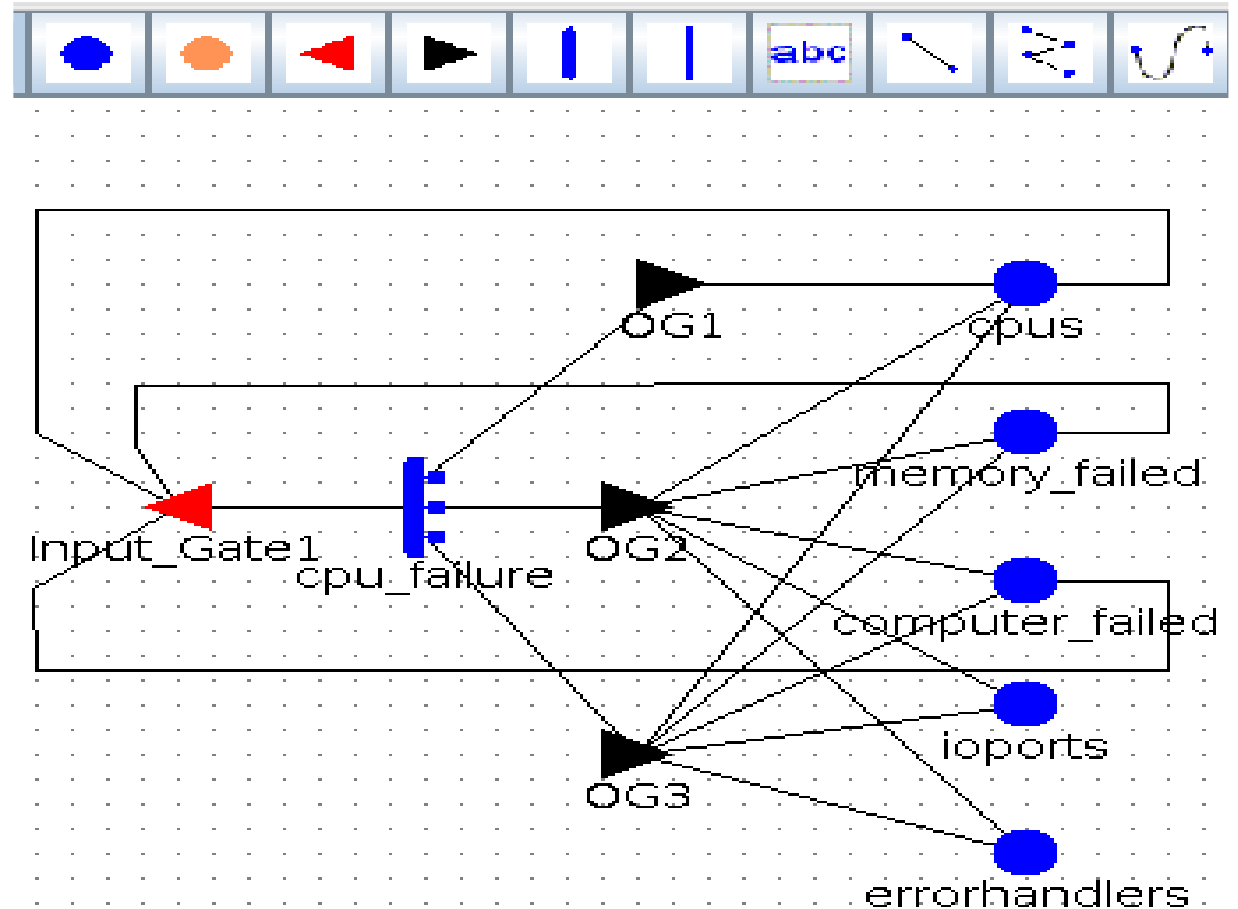
- Transitions may be **timed** or **instantaneous**
- **Enabling conditions** are defined with **input gates** associated with transitions
- **Firing rules**: user-defined functions specified in **input** or **output gates**

SAN in Möbius tool 2/2

Stochastic transition outcome:

Alternative results of a transition can be specified as mutually exclusive **cases** associated with the transition

Each case has a probability defined by a function of the **marking** (it may be a constant)

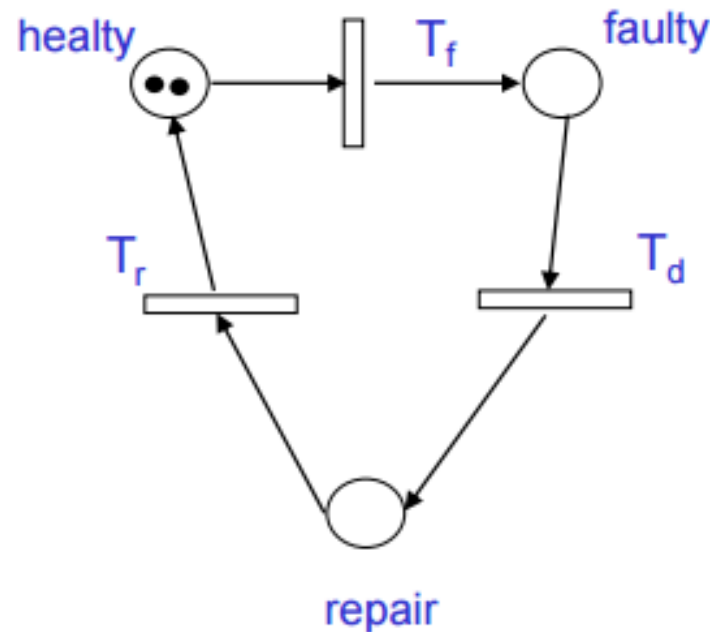


Elements of Möbius tool

- **Atomic and Composed model, reward, study, transformer and solver**
 - are used in the same way they are used for Fault Tree analysis.
- Möbius allows to evaluate a Performance Variable
 - **Steady State**
 - **Transient (instant of time)**

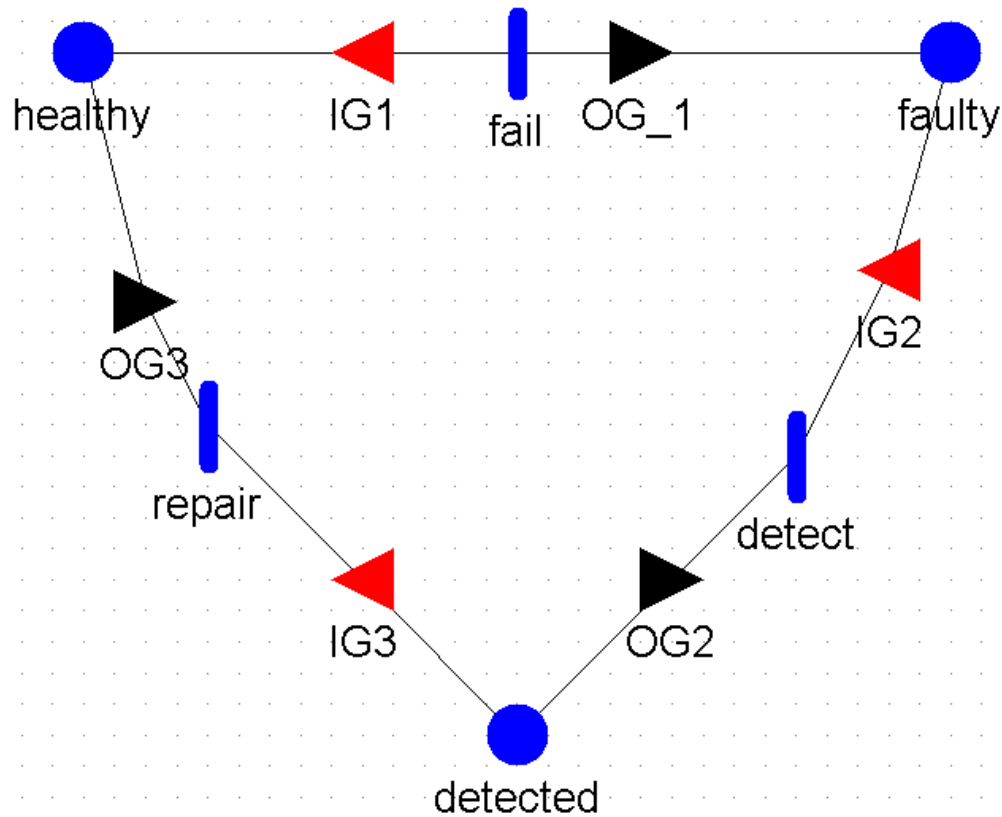
Failure-Detection-Repair

- Two identical CPUs
- **Failure of the CPU:** exponentially distributed with parameter λ
- **Fault detection:** exponentially distributed with parameter δ
- **CPU repair:** exponentially distributed with parameter μ



Evaluate
Availability of
the system
during steady
state,
considering that
the system is
working if at
least one CPU is
healthy

Atomic Model



- Remember to edit three global variables
 - (**lambda**, **mu** and **delta**)
- Set the initial state for places)
 - (**healthy** = 2, **faulty** = 0, **detected** = 0)
- Set the rate of each event as the number of token in the input place times the rate of the event
- Set the input enabling function, stating that the activities are enabled if there is at least one token in the input place
- Set the output function, so that it decreases the number of tokens in the input place and increases the ones of the output place by one

Fail example

Input Predicate

```
healthy->Mark() > 0
```

Output Function

```
faulty->Mark() ++;  
healthy->Mark() --;
```

Name:

fail

Time distribution function:

Exponential

1

Rate

```
return lambda*healthy->Mark();
```

Case quantity:

1

Case 1

1

Reward model

- Create a performance variable called **availability**.
- Express its **reward function** according to the condition of correctness of the system.
- Set a steady-state **Time** option with default configurations.

Available State Variables (double click to insert)

```
ex1_san->healthy  
ex1_san->faulty  
ex1_san->detected
```

Reward Function

```
if( ex1_san->healthy->Mark() > 0) return 1;  
else return 0;
```

Study model

Type:	double
Initial	0.1
Final	0.5
	☒ Additive
	☐ Multiplicative
	☐ Exponential
Increment	0.2

Set a **range study** model where all the three rates vary from 0.1 to 0.5 with a step of 0.2.

All the possible combinations lead to **27 experiments**

Transformer and Solver

- Again use the State Space Generator (**NOT** Symbolic) as transformer
- Then, in order to evaluate the steady state behavior choose a Steady State solver.
 - For simplicity select the Direct Steady State Solver.
- The behavior of different solvers can be found in the Möbius wiki.

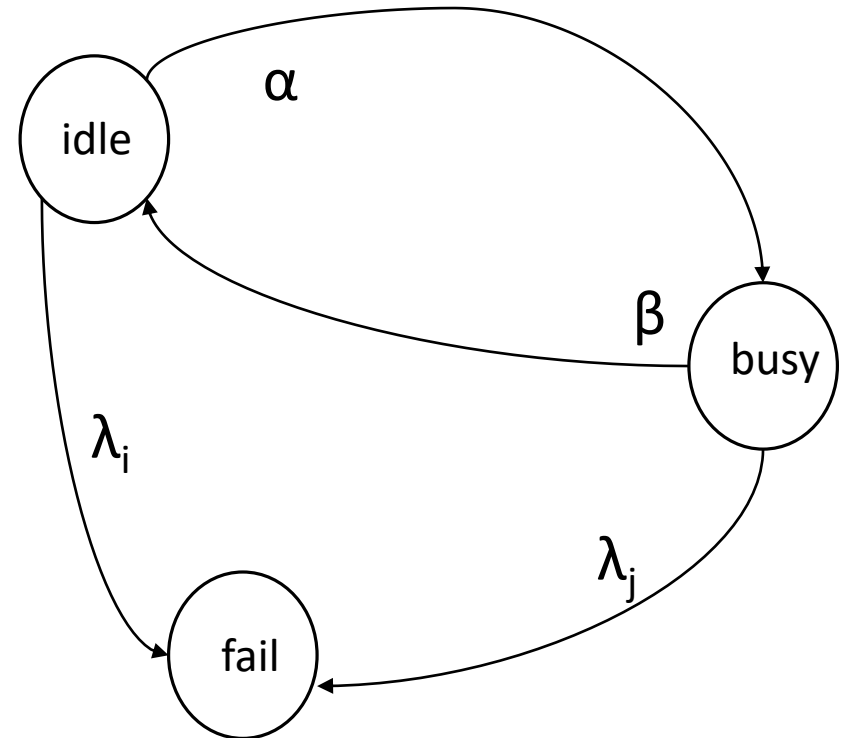
Results

- Experiment 1:
 $\lambda = 0.1$, $\delta = 0.1$, $\mu = 0.1 \rightarrow \text{Availability} = 0.55555$
- Experiment 14:
 $\lambda = 0.3$, $\delta = 0.3$, $\mu = 0.3 \rightarrow \text{Availability} = 0.55555$
- Experiment 4:
 $\lambda = 0.3$, $\delta = 0.1$, $\mu = 0.1 \rightarrow \text{Availability} = 0.2653$
- Experiment 3:
 $\lambda = 0.1$, $\delta = 0.5$, $\mu = 0.1 \rightarrow \text{Availability} = 0.702$

Exercise

- A computer is *idle*, *busy*, or *failed*;
- jobs arrive at a rate α ;
 - 1000
- jobs are completed at a rate β ;
 - 10000
- the computer fails at rate λ_i when idle;
 - 1.0E-7
- the computer fails at rate λ_j when busy.
 - From 1×10^{-6} to 5×10^{-6}

Evaluate reliability after 24
time units of operation



References

William H. Sanders and John F. Meyer, ``Stochastic Activity Networks: formal definitions and concepts'', in Lectures on formal methods and performance analysis: first EEF/Euro summer school on trends in computer science, 2002.

https://www.mobius.illinois.edu/wiki/index.php/Möbius_Documentation

Thanks to prof. Andrea Domenici for previous version of the slides.