Probabilistic Model Checking with PRISM

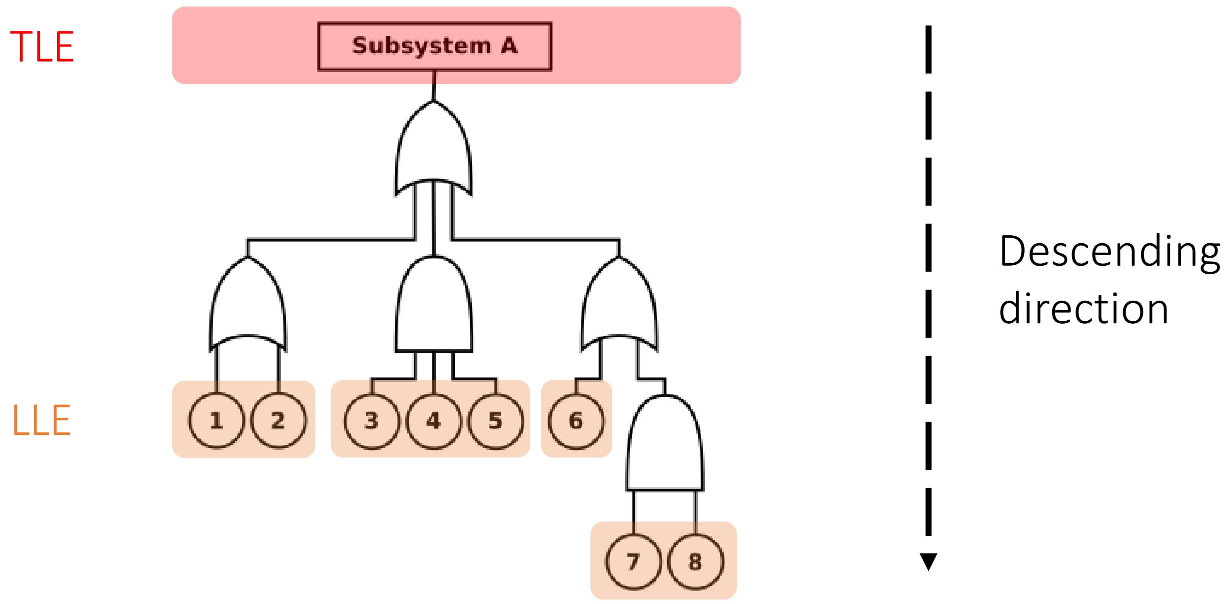
Production Line

Patrick SARDINHA

Fault Trees

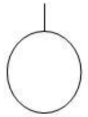
Fault Trees (FT) graphically represent possible combinations of events (Low Levels Events) leading to a predefined undesirable event (Top Level Event)

Representation:

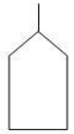


Graphic symbols

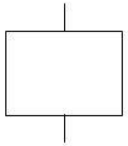
Events:



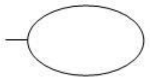
Basic event: failure in a system component



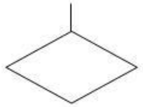
External event: expected to occur



Intermediate event: events occurring at the exit of a door



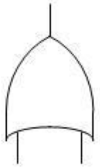
Conditioning event: an event with conditions



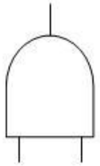
Undeveloped event: an event with insufficient information

Gates:

Describe the relationship between input and output events.



OR gate: the output occurs if any input occurs



AND gate: the output occurs only if all inputs occur

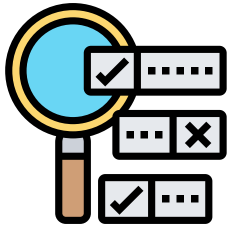
Fault Tree Analysis (FTA)

Method used to evaluate the risks of a system and allows to:

Understand how a system can fail

Know how to reduce the risks

Visualize the event rates of an accident



FTA is often performed by transforming the FT into a Boolean function which is used for simulation …

→ … but this methodology has a lot of constraints (time/resources)

A new formal Probabilistic FTA methodology

Efficient Probabilistic Fault Tree Analysis of Safety Critical Systems via Probabilistic Model Checking

Marwan Ammar, Ghaith Bany Hamad, Otmane Ait Mohamed, Yvon Savaria

A new formal Probabilistic FTA methodology

The idea is as follows:

1. Model the system (composed of components) and specify event parameters
2. Synthesize the system fault tree
3. Model the behavior of each FT gate as a probabilistic automaton (PA)
4. Generate a formal MDP(?) model of the fault tree with the parallel composition of the PA (PRISM)
5. Analyze the MDP(?) model to evaluate the maximum probability of Top Level Event (TLE)

System description

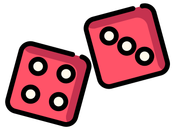
We have a **production chain** made up of:


Machines that extract resources


Machines that transform resources

We have different types of **disruptive primary events**:


Technical failures on machines with a certain probability


Non-deterministic quantities of extracted resources

… And others

Resource extraction

We have different kinds of machine:

	**Burner mining drill**
Inputs	Raw minerals (from a source)
Outputs	Minerals
Basic event	- Can break down - May be affected by an external event - The input quantity may vary

	**Offshore pump**
Inputs	Water (from a source)
Outputs	Water
Basic event	- Can break down - May be affected by an external event - The input quantity may vary

Resource transformation

We have different kinds of machine:

	**Boiler**
Inputs	Fuel & water
Outputs	Steam
Basic event	- Can break down - May be affected by an external event - Fuel not supplied - Water not supplied

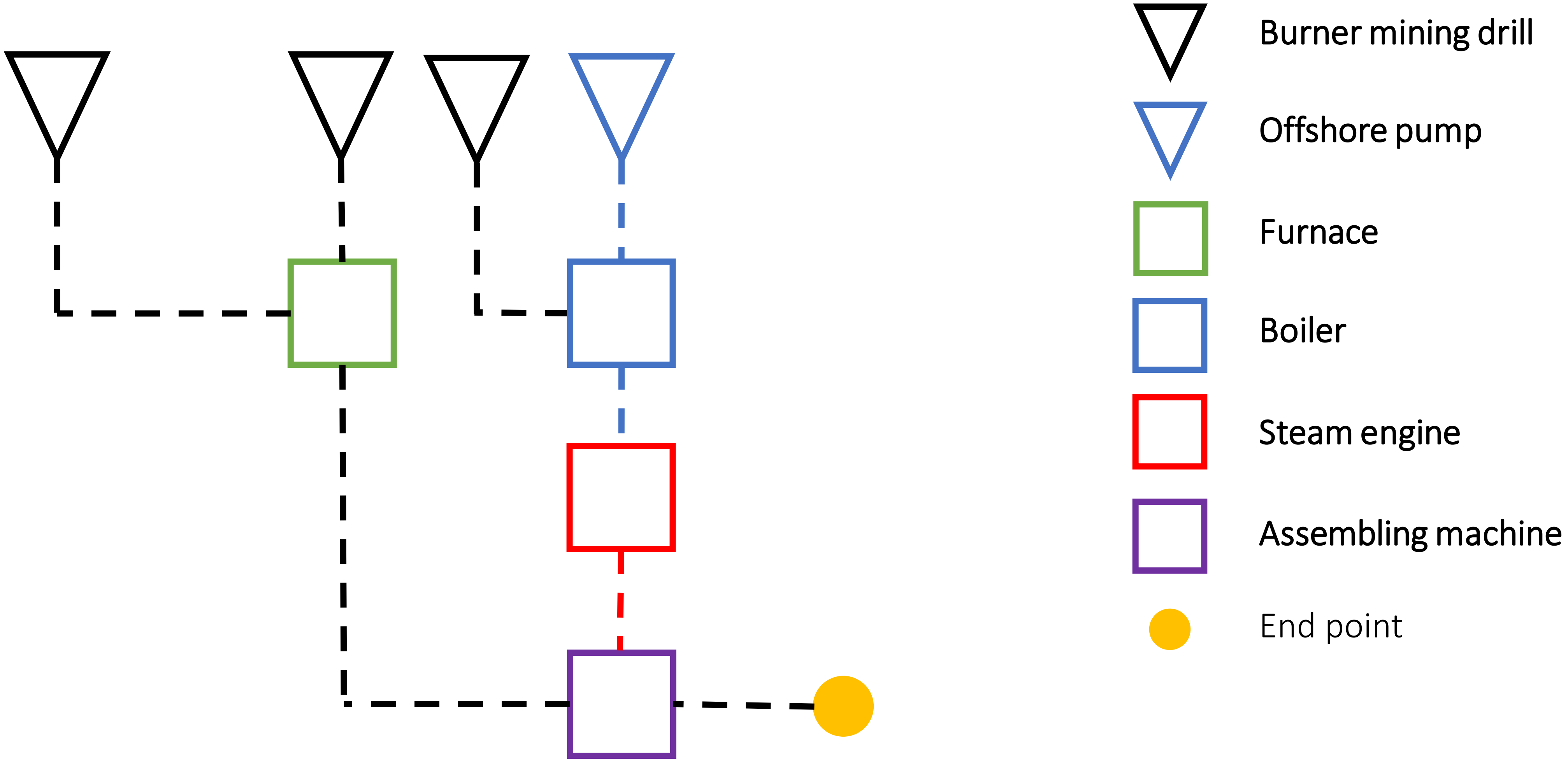
	**Steam engine**
Inputs	Steam
Outputs	Electricity
Basic event	- Can break down - May be affected by an external event - Steam not supplied

Resource transformation

	Furnace
Inputs	Fuel & minerals
Outputs	Processed minerals
Basic event	- Can break down - May be affected by an external event - Fuel not supplied - Minerals not supplied

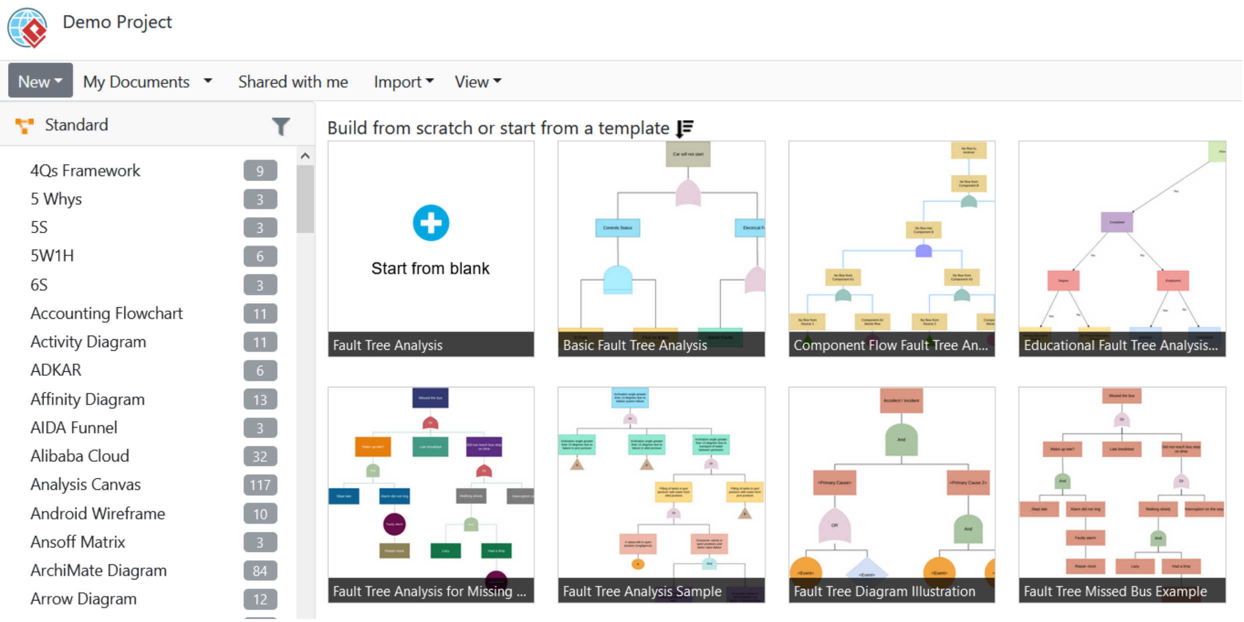
	Assembling machine
Inputs	Electricity & Processed minerals
Outputs	Final product
Basic event	- Can break down - May be affected by an external event - Electricity not supplied - Processed minerals not supplied

Production line: Example

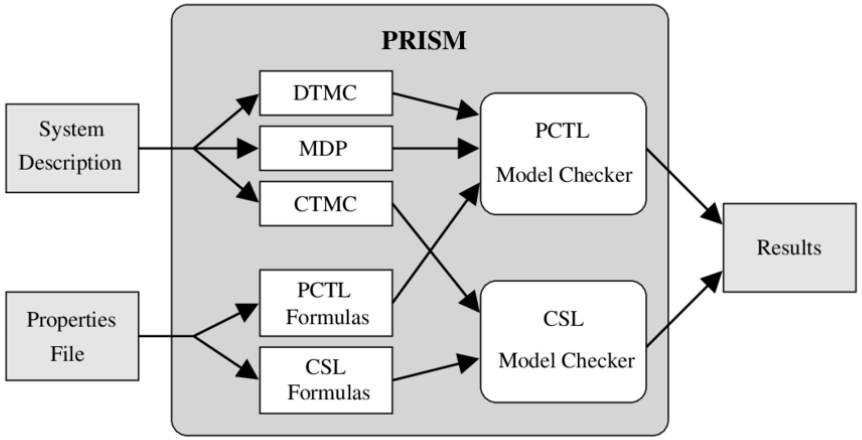


Some tools

VisualParadigm Online



PRISM



Interesting property to check

Estimate and compare the probability that faults from different low-level events cause a system failure

→ Production of Assembling Machine is zero

System Failure
Undesirable event

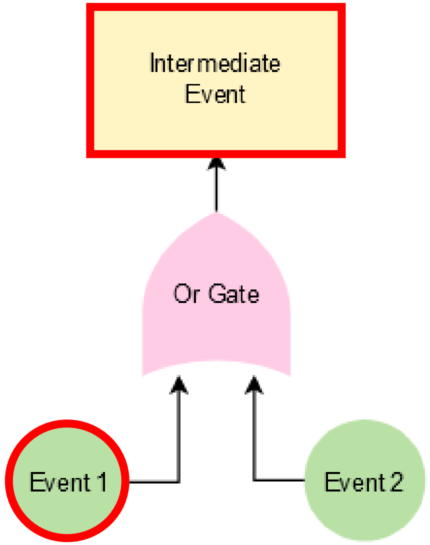
Top Level Event (TLE)

This, allows us to identify the most critical component of the system and we can then apply redundancy (TMR) on this element for example

Some mechanisms

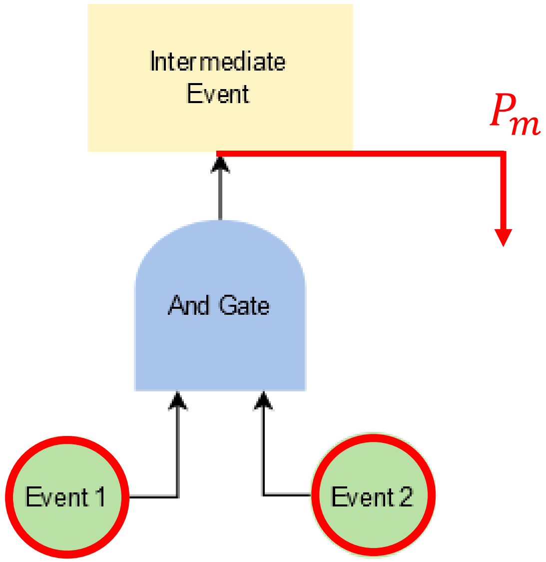
Fault propagation:

An error in a node at a lower level of the tree can propagate to a higher level

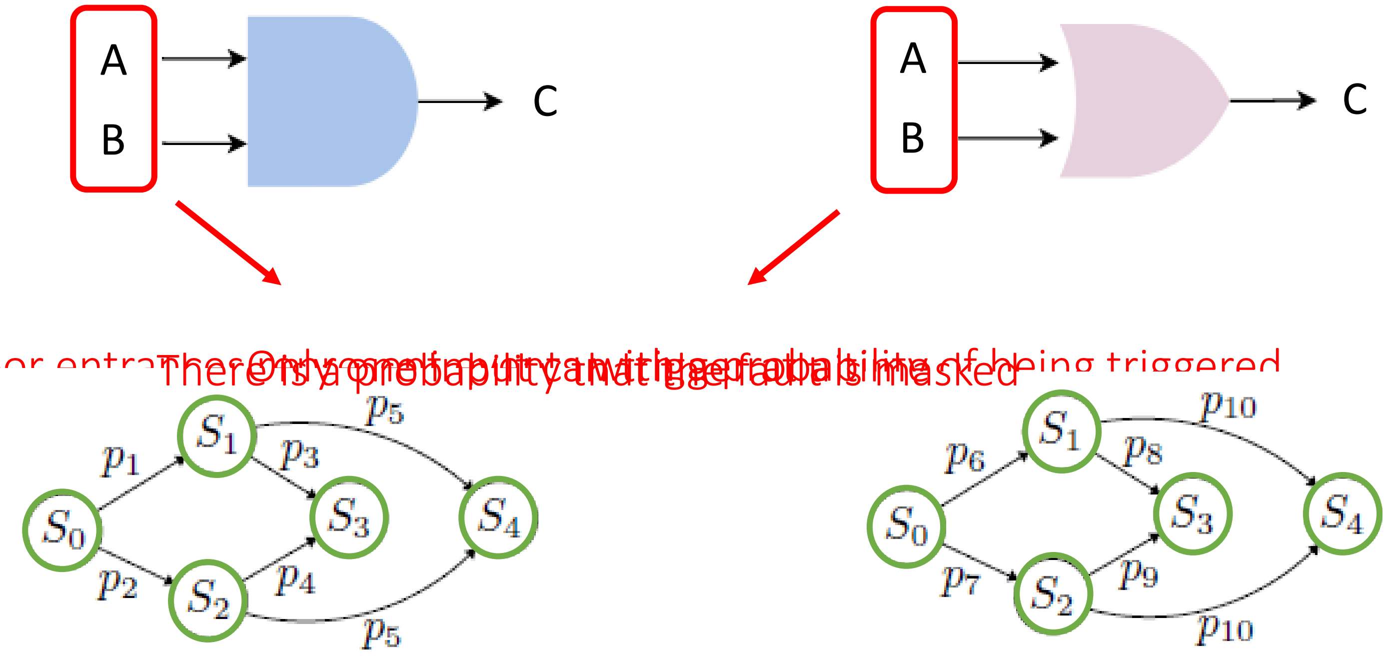


Fault masking:

Adds a probability of fault mitigation inside the gates

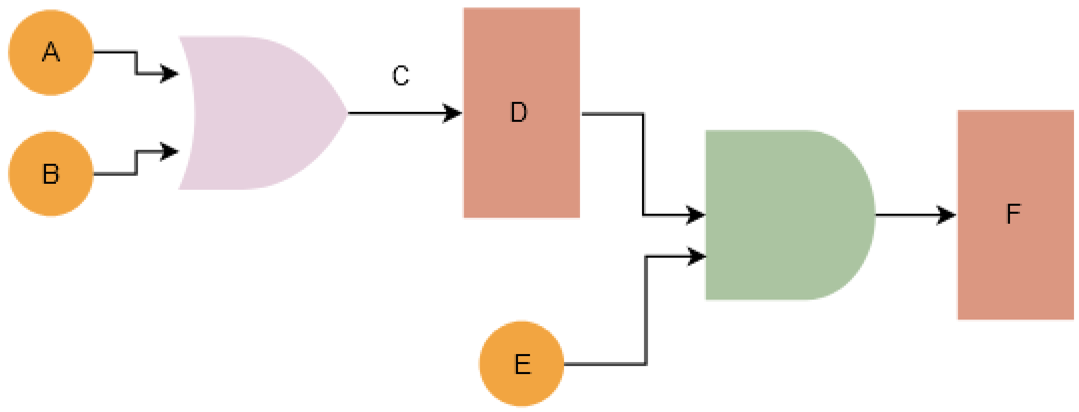


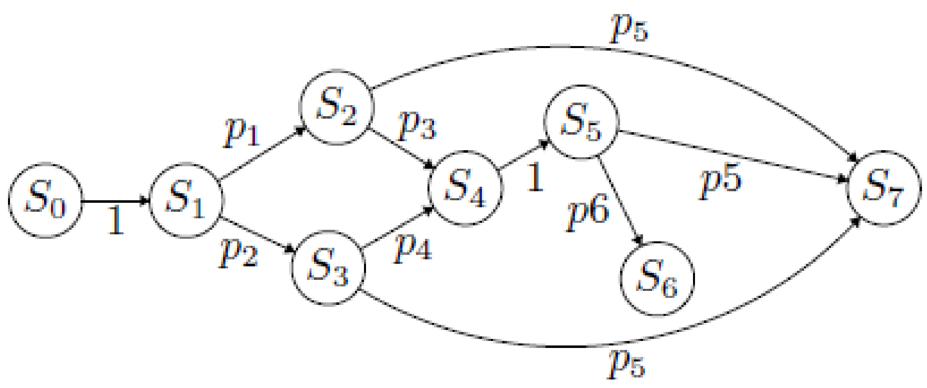
FT gates



A
B
C
There is a probability that the fault is masked
S_0 S_1 S_2 S_3 S_4
p_1 p_2 p_3 p_4 p_5
p_6 p_7 p_8 p_9 p_{10}

Example: Door combination





FT gates with PRISM

AND gate

```
module and_gate
[] (and=1)&(X=0)&(Y=0)&(M=0)&(Z=0)->p1:(X'=1)&(and'=2)
                                  +p2:(Y'=1)&(and'=2);
[] (X=1)&(Y=0)&(M=0)->p5:(M'=1)&(X'=0)+p3:(Y'=1)&(Z'=1);
[] (Y=1)&(X=0)&(M=0)->p5:(M'=1)&(Y'=0)+p4:(X'=1)&(Z'=1);
endmodule
```

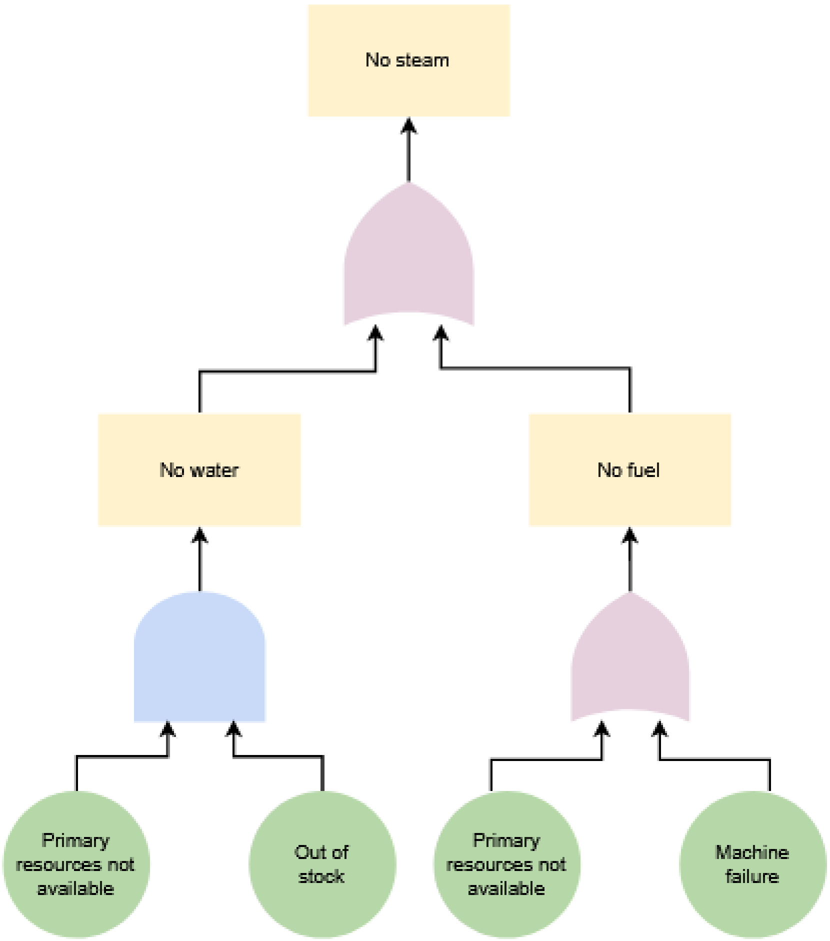
OR gate

```
module or_gate
[] (or=1)&(A=0)&(B=0)&(M=0)&(C=0)->p1:(A'=1)&(or'=2)
                                 +p2:(B'=1)&(or'=2);
[] (A=1)&(C=0)&(M=0)->p5:(M'=1)&(A'=0)+p3:(A'=0)&(C'=1);
[] (B=1)&(C=0)&(M=0)->p5:(M'=1)&(B'=0)+p4:(B'=0)&(C'=1);
endmodule
```

```
module twogate
[]  or=0  ->  (or'=1);
[]  c=1  ->  (c'=0)&(d'=1);
endmodule
module or_gate
[] (or=1)&(a=0)&(b=0)&(m=0)&(c=0)->p1:(a'=1)&(or'=2)
                                 +p2:(b'=1)&(or'=2);
[] (a=1)&(c=0)&(m=0)->p5:(m'=1)&(a'=0)+p3:(a'=0)&(c'=1);
[] (b=1)&(c=0)&(m=0)->p5:(m'=1)&(b'=0)+p4:(b'=0)&(c'=1);
endmodule
module and_gate
[] (and=1)&(d=0)&(e=0)&(m=0)&(f=0)->p6:(d'=1)&(and'=2)
                                  +p7:(e'=1)&(and'=2);
[] (d=1)&(e=0)&(m=0)->p5:(m'=1)&(d'=0)+p8:(e'=1)&(f'=1);
[] (e=1)&(d=0)&(m=0)->p5:(m'=1)&(e'=0)+p9:(d'=1)&(f'=1);
endmodule
```

Combination OR / AND

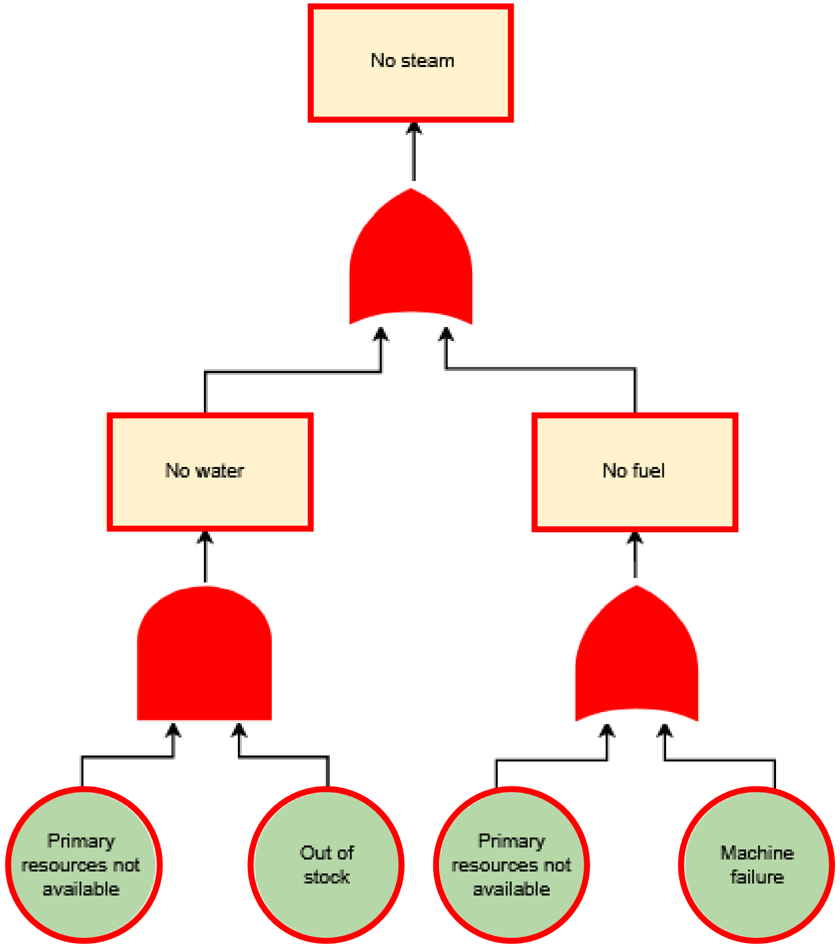
Application example: Boiler

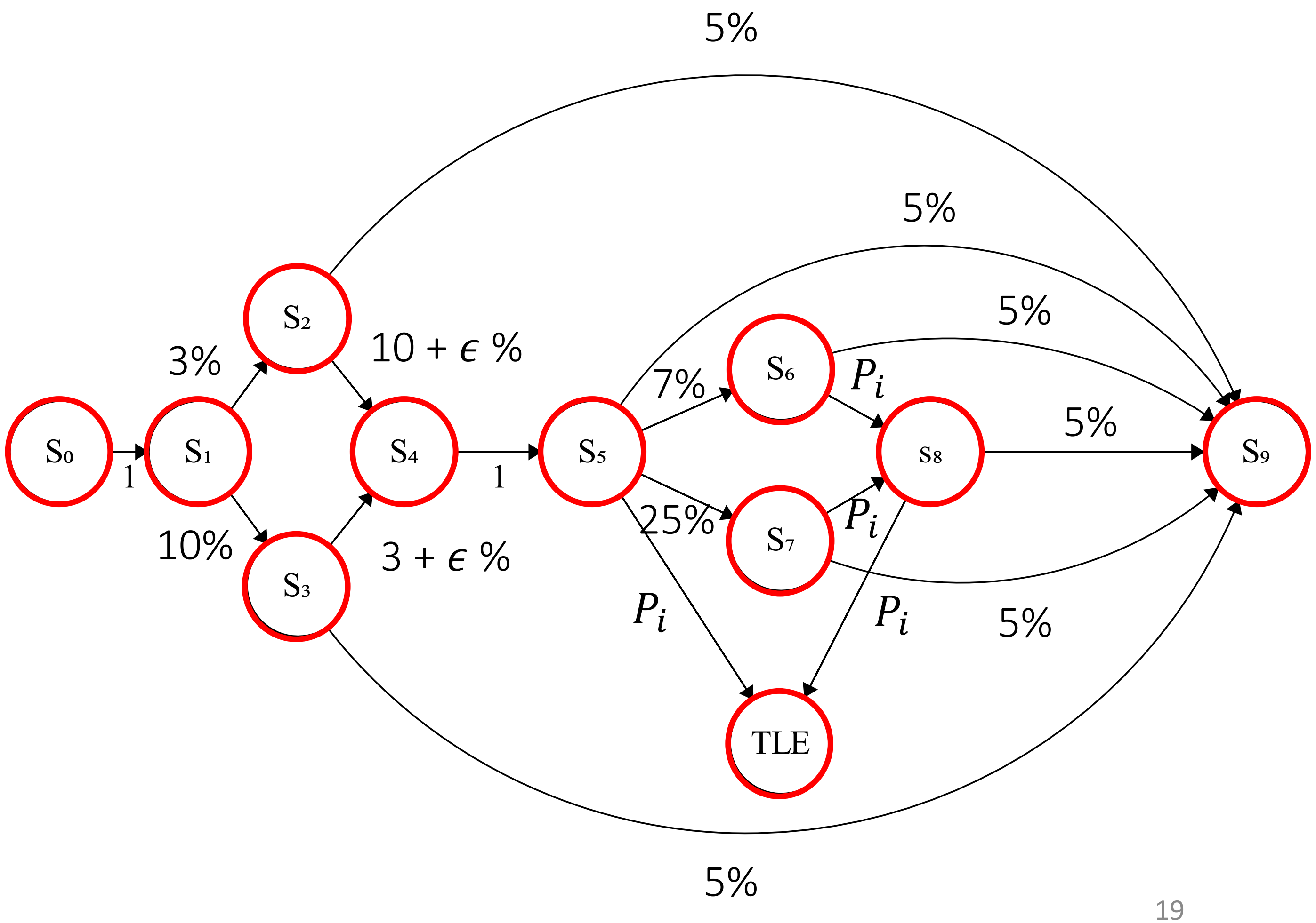


Bottom Event	Probability
Primary resources not available	3%
Out of stock	10%
Primary resources not available	7%
Machine failure	25%

Fault masking prob: 5%

Application example: Boiler



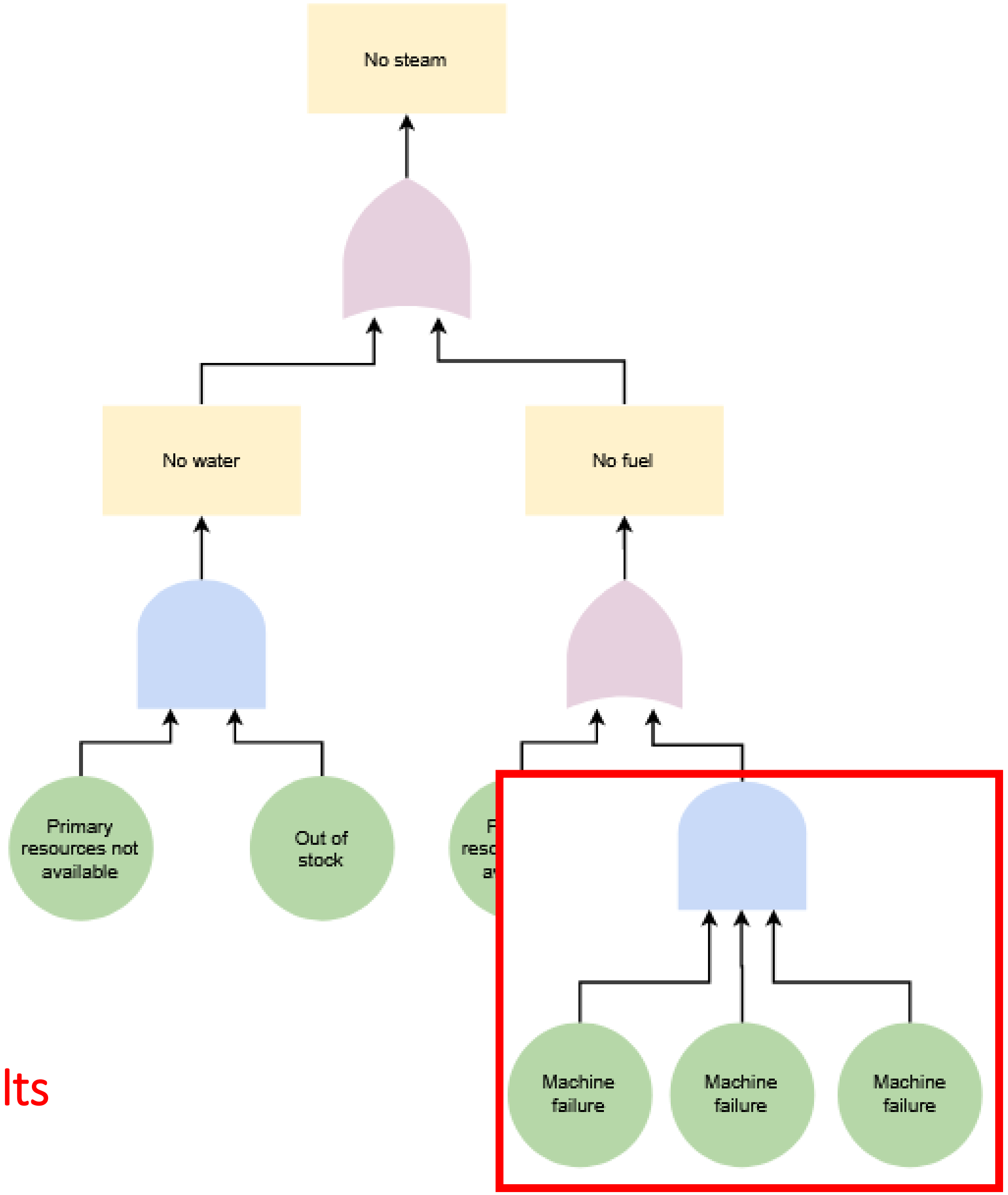


Add redundancy

Example of TMR
(Triple Modular Redundancy)

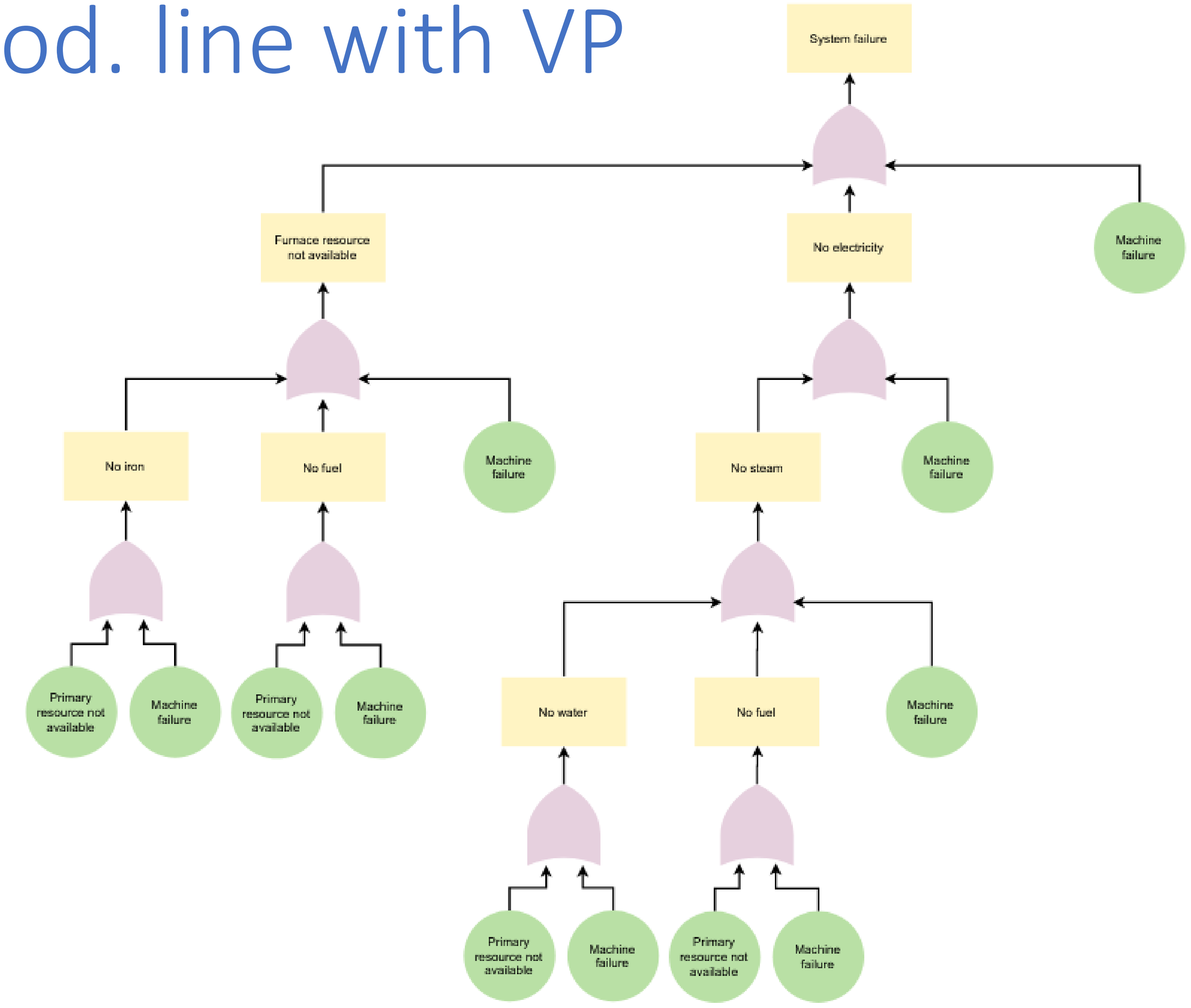
It's a way to improve a production system

with the use of thresholds …



→ We can abstract this method by **masking faults**

Prod. line with VP



Production line fault tree

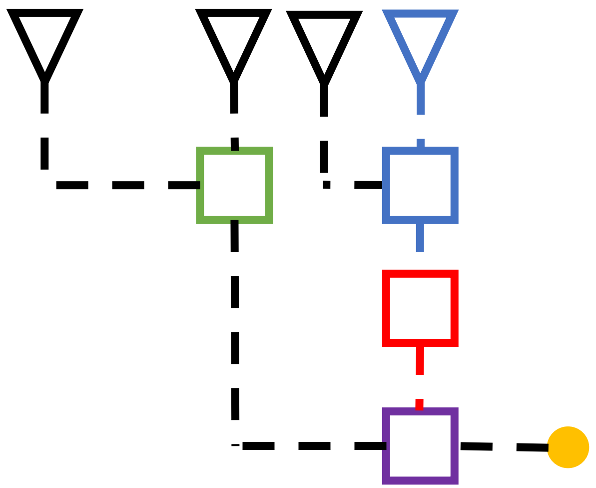


Diagram of the production line

Check property with PRISM

To estimate the probability that a low-level event leads to system failure with PRISM

For a node connected to an **OR gate**:

$$P_{max} = ?\ [\ (F\ X_i = 1)\ \&\ (F\ TLE = 1)]$$

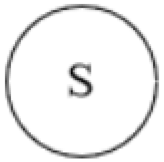
The maximum probability that event X_i will trigger and the fault is propagated to the TLE ?

For a node connected to an **AND gate**:

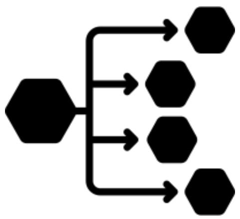
$$P_{max} = ?\ [\ (F\ X_i = 1)\ \&\ (F\ X_j = 1)\ \&\ (F\ TLE = 1)]$$

The maximum probability that event X_i will trigger and event X_j will trigger and the fault is propagated to the TLE ?

Work incoming



Represent the production line with the PA



See which model use between DTMC & MDP

Implement our system with PRISM

References

PRISM: https://www.prismmodelchecker.org/

Factorio wiki: https://wiki.factorio.com/Main_Page

Modélisation et simulation de flux de production:

Franck Fontanili, *Intégration d'outils de simulation et d'optimisation pour le pilotage d'une ligne d'assemblage multiproduit à transfert asynchrone,* Partie IV, page 87-133

References

FTA: https://en.wikipedia.org/wiki/Fault_tree_analysis

FTA via PMC: M. Ammar, G. B. Hamad, O. A. Mohamed and Y. Savaria, "*Efficient probabilistic fault tree analysis of safety critical systems via probabilistic model checking* "

https://ieeexplore.ieee.org/abstract/document/7880373/metrics#metrics

SML: https://fr.wikipedia.org/wiki/Systems_Modeling_Language

VP: https://online.visual-paradigm.com/fr/diagrams/features/fault-tree-analysis-software/

MDP: https://en.wikipedia.org/wiki/Markov_decision_process

References

FMTs:

M. Ammar, G. B. Hamad and O. Ait Mohamed, "Probabilistic High-Level Estimation of Vulnerability and Fault Mitigation of Critical Systems Using Fault-Mitigation Trees (FMTs)," *2019 IEEE Latin American Test Symposium (LATS)*, Santiago, Chile, 2019, pp. 1-6
https://ieeexplore.ieee.org/document/8704589

FTA via PMC (case study):

M. Ammar, K. A. Hoque and O. A. Mohamed, "Formal analysis of fault tree using probabilistic model checking: A solar array case study," *2016 Annual IEEE Systems Conference (SysCon)*, Orlando, FL, USA, 2016, pp. 1-6

https://ieeexplore.ieee.org/abstract/document/7490556

Finite State Machine Designer:

https://www.cs.unc.edu/~otternes/comp455/fsm_designer/

Probabilistic Model Checking with PRISM

Production Line

Patrick SARDINHA