Verification Conquers Fault Tree Analysis

Joost-Pieter Katoen

Software Modeling and Verification Chair | RWTH AACHEN UNIVERSITY

UNIVERSITY OF TWENTE.

Reliability

Reliability Engineering

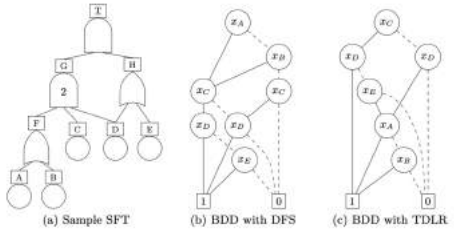
- Risk analysis ensures that critical assets, like medical devices and nuclear power plants, operate in a safe and reliable way.
- Fault tree analysis (FTA) is one of the most prominent techniques.
- Used by a wide range of industries (aerospace, automotive, nuclear, medical, process engineering)
- Used by many companies and institutions: FAA, NASA, ESA, Airbus, Honeywell, etc.
- Industrial standards by the IEC and by ISO for automotive applications

The SpaceEx Falcon-9 Explosion

A launch failure in 2015 resulted in a loss of a quarter billion dollars

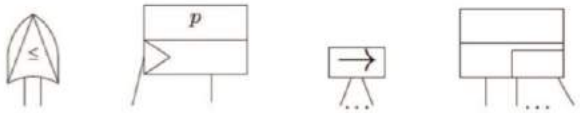
Talk Overview

1.



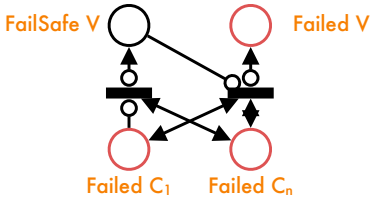
Classical Static Fault Trees

2.



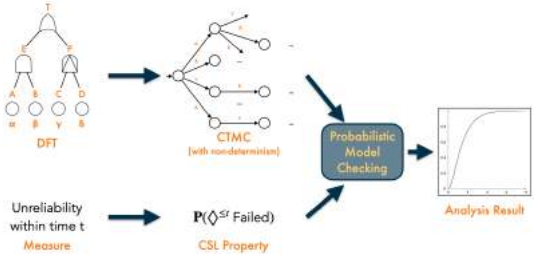
Dynamic Fault Trees

3.



Petri Net Semantics

4.



Scaling Up DFT Analysis

5.

Industrial Case Studies

6.

SAFEST
Dynamic Fault Tree Analysis

Commercialisation

Static Fault Trees

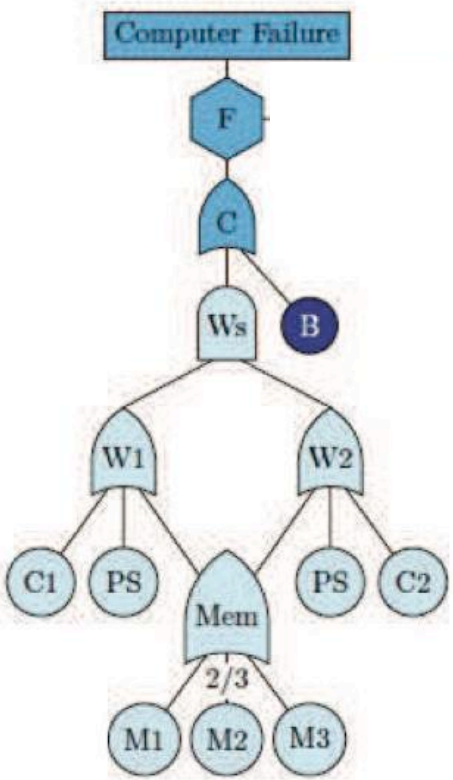
[Watson, 1961-62]

- Fault tree is a directed acyclic graph consisting of two types of nodes: events (depicted as circles) and gates:

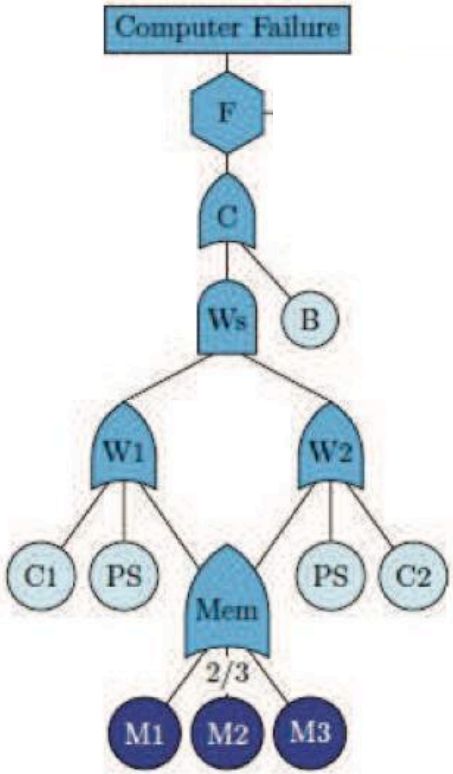
(a) AND gate (b) OR gate (c) k/N gate (d) INHIBIT gate

- An event is an occurrence within the system, typically the failure of a component or sub-system.
- Events can be divided into:
  - basic events (BEs), which occur on their own, and
  - intermediate events, which are caused by other events
- The root, called the top level event (TLE), models a system failure

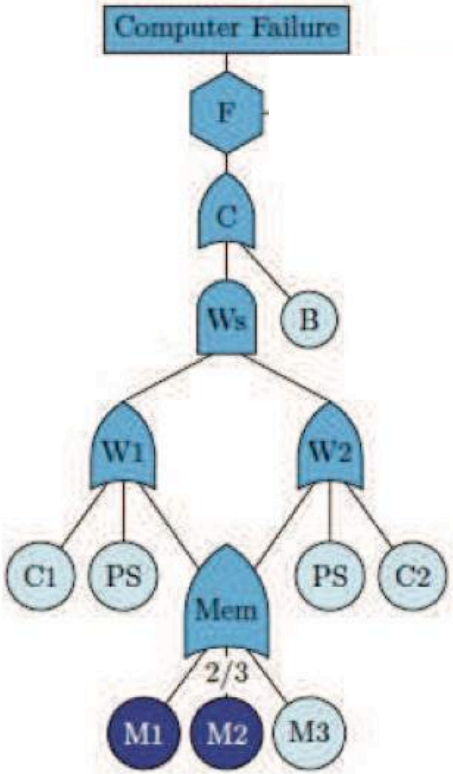
Minimal Cut Sets



minimal



not minimal

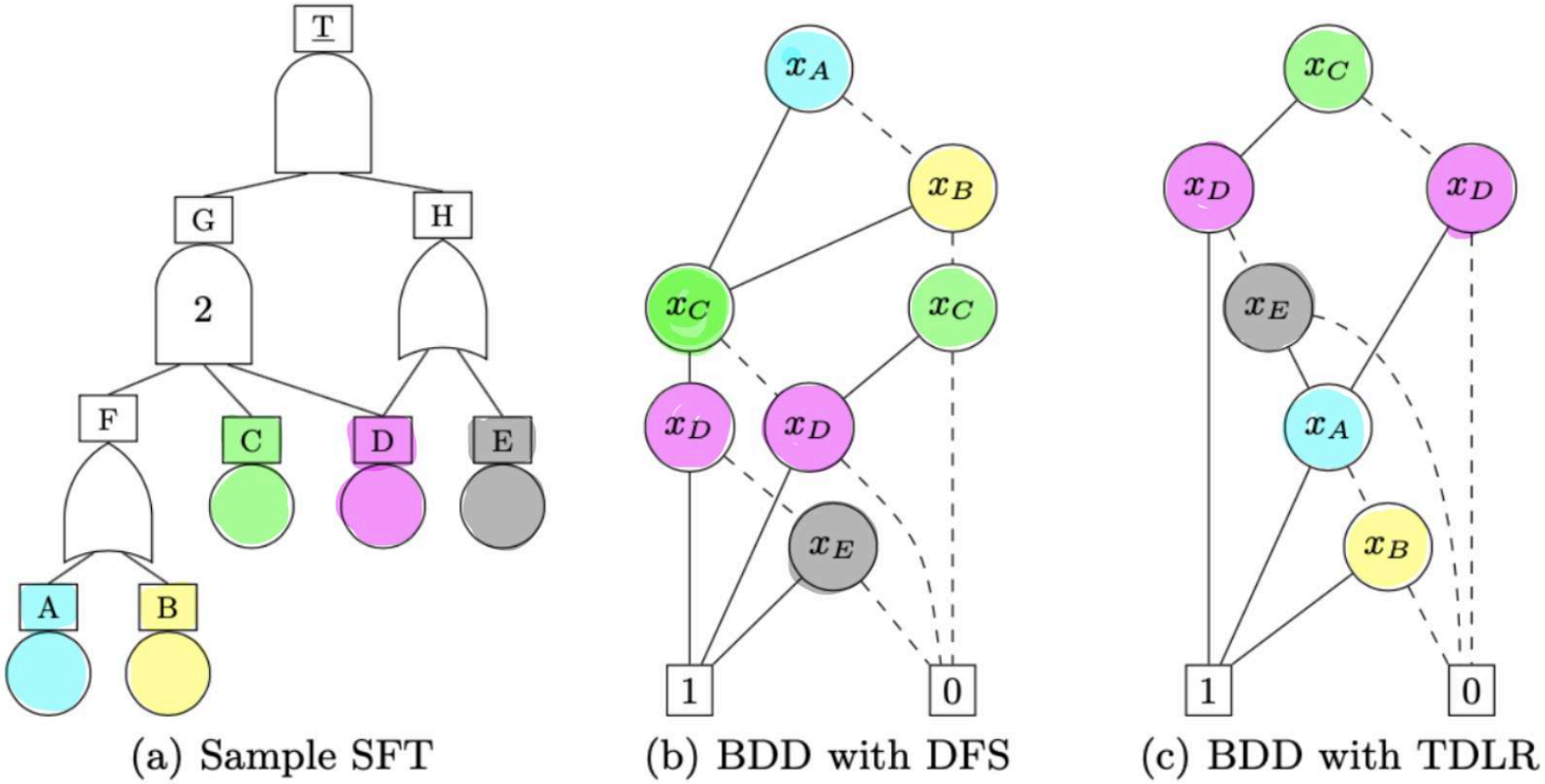


minimal

A cut set is a set of components that together can cause the system to fail.

A minimal cut set is a cut set without proper subset being a cut set.

SFT Analysis



(a) Sample SFT (b) BDD with DFS (c) BDD with TDLR

- Turn SFT into propositional logic formula
- Encode as a binary decision diagram
- Calculate minimal cut sets, MTTF, reliability and sensitivity using BDDs

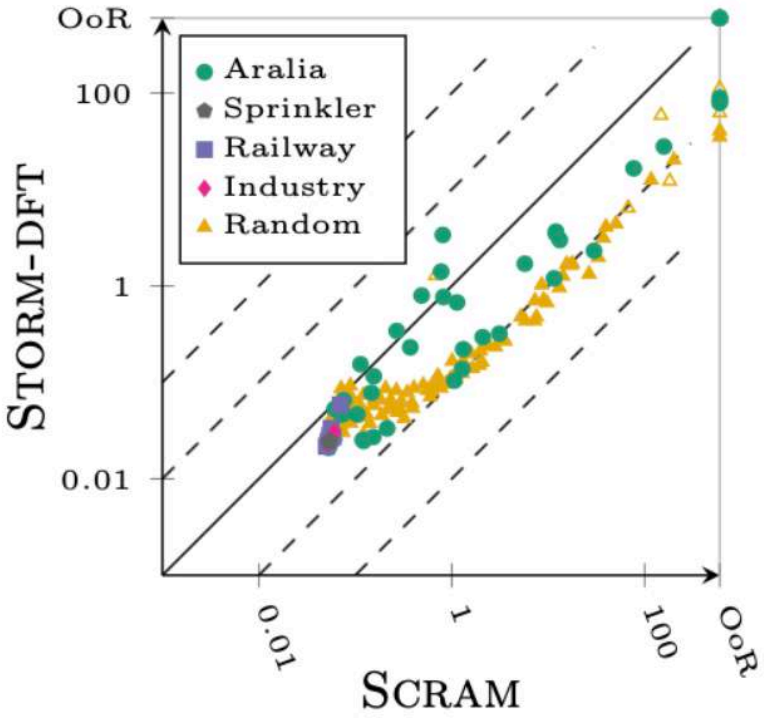
Experiments: Computing MCS

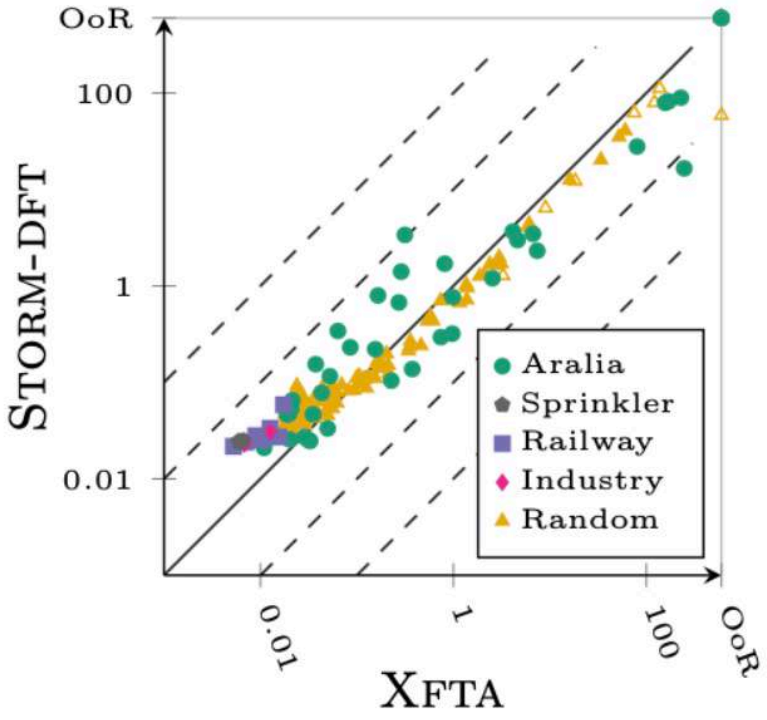
[Basgöze et al., NASA FM 2022]

	Aralia	Sprinkler	Railway	Industry	Random	Random (Large)
#BEs	25–1567	31	22–54	36–184	150	500
#Gates	20–1622	35	69–259	21–67	70-122	261-316

Storm

all run times in seconds





Storm-DFT computes MCSs faster than XFTA and SCRAM for large SFTs

SFT Deficiencies

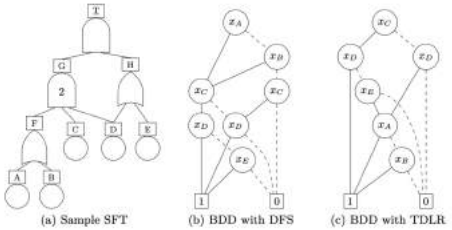
- Their simplicity
  - simple to comprehend and analyse
  - too simple to model realistic scenarios
- Lack of common dependability patterns
  - spare management
  - functional dependencies (e.g., common-cause failures)
  - redundancies
- Static behaviour
  - no temporal orderings of faults
  - top-level event only depends on set of failed events

Many variants:

state-event fault trees, boolean-logic driven Markov processes,
SD fault trees, PANDORA fault trees, Dugan's dynamic fault trees

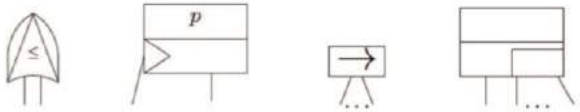
Talk Overview

1.



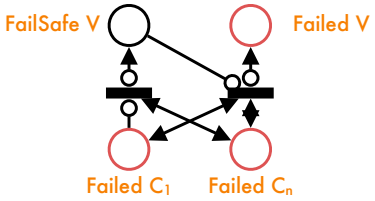
Classical Static Fault Trees

2.



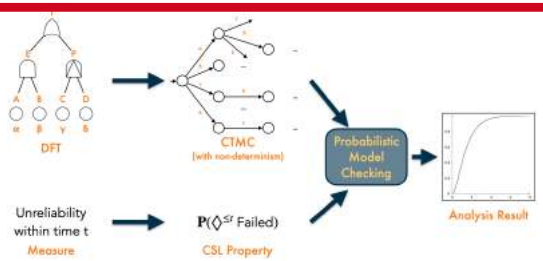
Dynamic Fault Trees

3.



Petri Net Semantics

4.



Scaling Up DFT Analysis

5.

Industrial Case Studies

6.

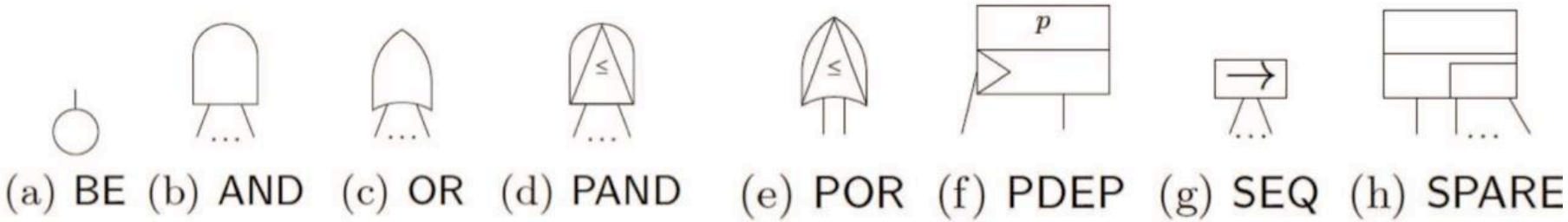
SAFEST
Dynamic Fault Tree Analysis

Commercialisation

Dugan's Dynamic Fault Trees

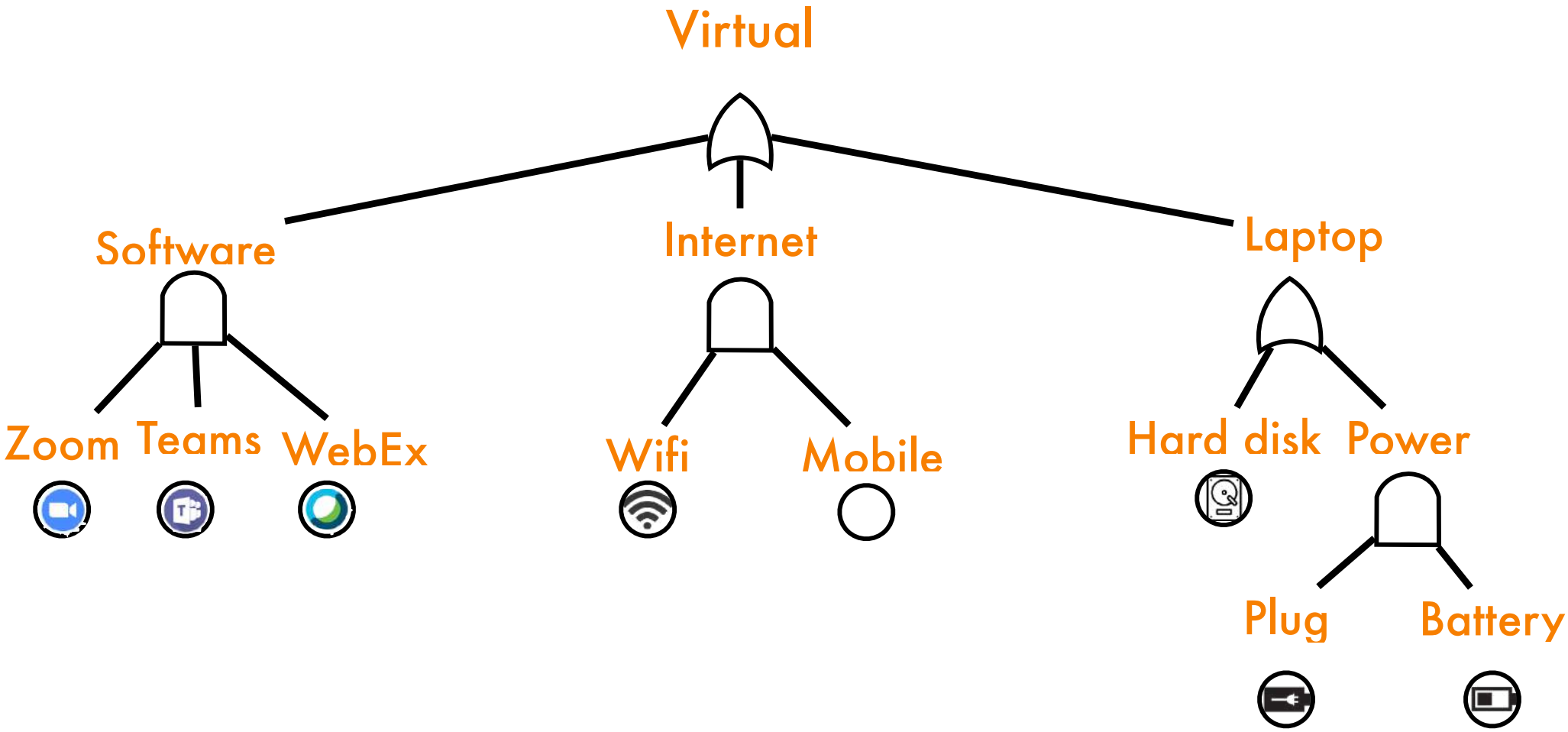
2000 IEEE Reliability Society Award

> *"Dynamic fault tree analysis has extended the state of the art and the state of the practice in analysis of the dependability of computer systems."*
>
> - JOANNE BECHTA DUGAN, PROFESSOR OF ELECTRICAL & COMPUTER ENGINEERING

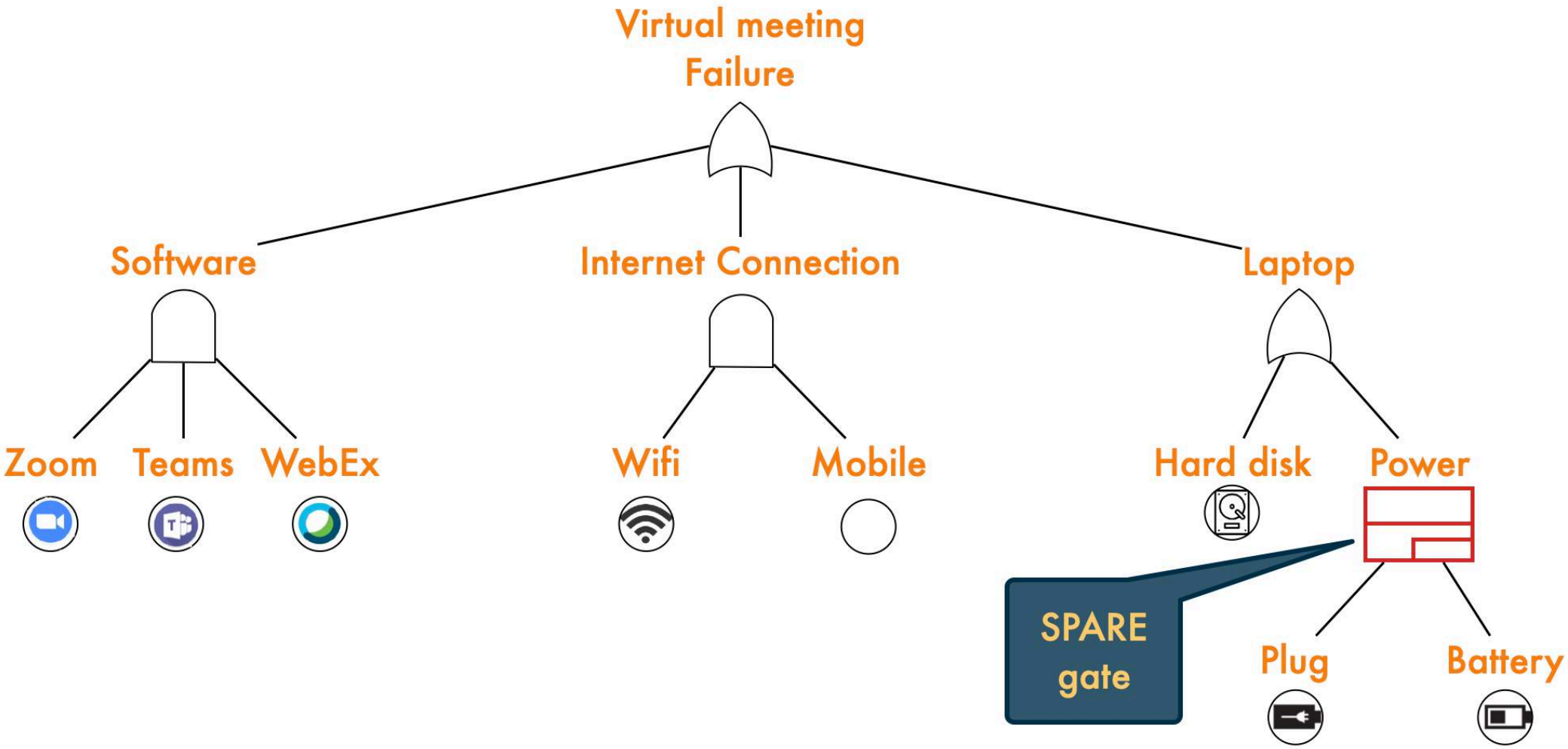


Galileo User's Manual & Design Overview

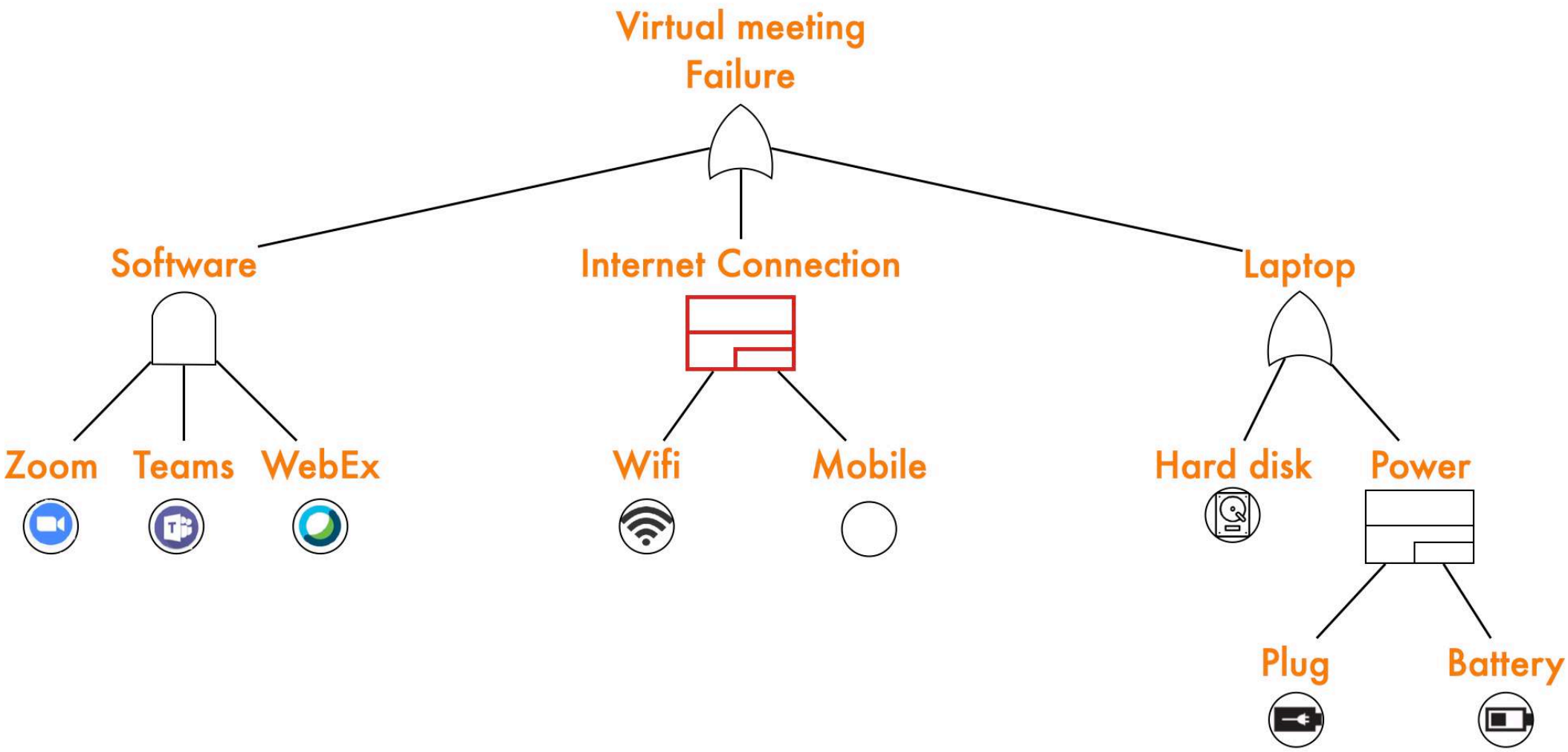
A Sample Dynamic Fault Tree



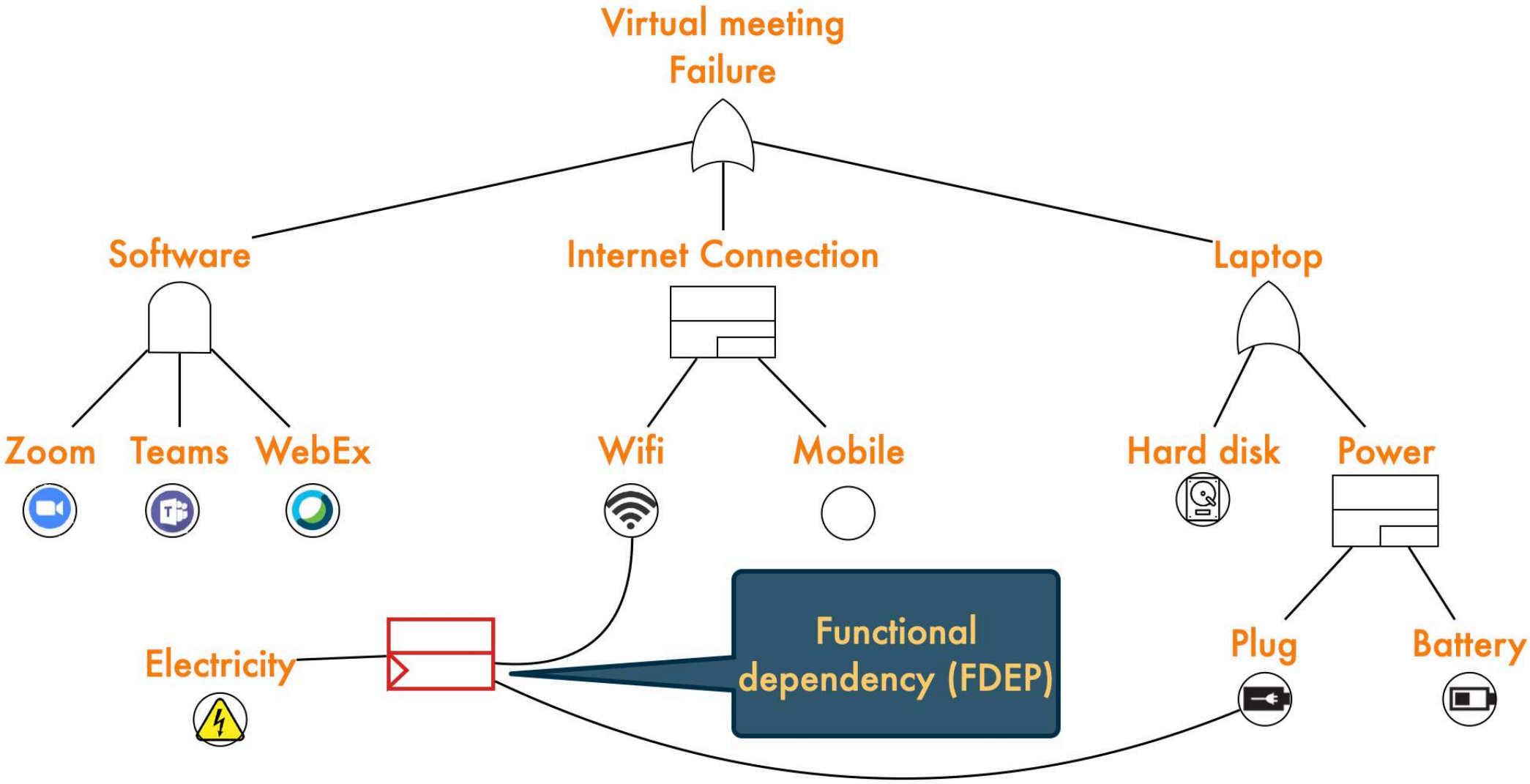
A Sample Dynamic Fault Tree



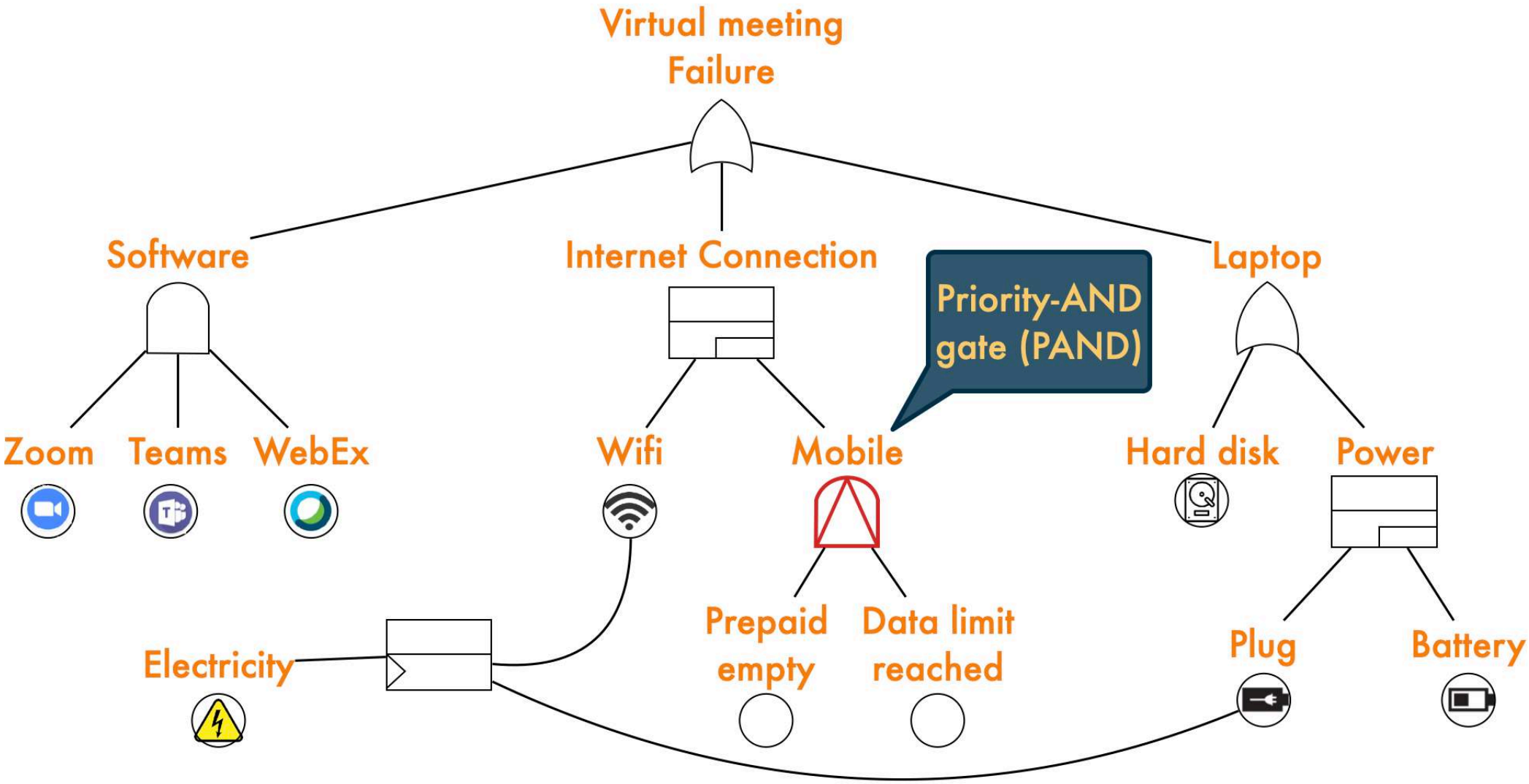
A Sample Dynamic Fault Tree



A Sample Dynamic Fault Tree



A Sample Dynamic Fault Tree



No Free Lunch

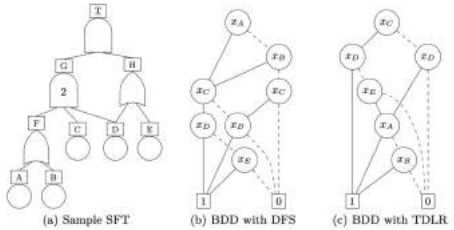
- Minimal cut set analysis not applicable
  - generalisation to cut sequences insufficient
  - the behaviour of a DFT is history-dependent
- Analysis by generating stochastic (decision) process
  - continuous-time Markov chains/decision processes
  - other approaches: via Bayesian networks, Petri nets
- Use Markov chain analysis to obtain measures

„The construction of a Markov model for any but the simplest system is tedious and error prone.“

[Dugan et al., 1992]

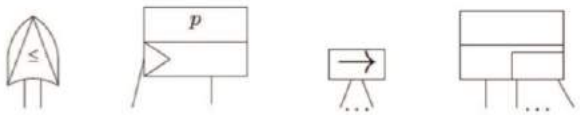
Talk Overview

1.



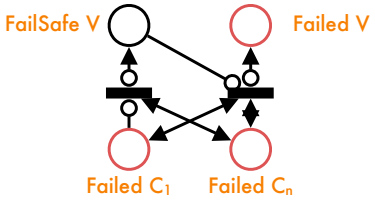
Classical Static Fault Trees

2.



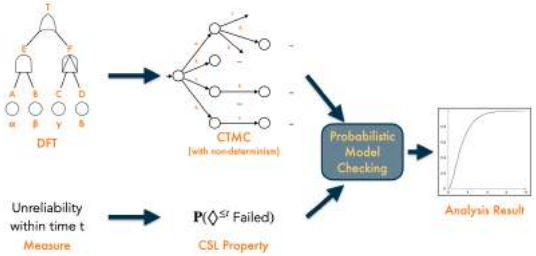
Dynamic Fault Trees

3.



Petri Net Semantics

4.



Scaling Up DFT Analysis

5.

Industrial Case Studies

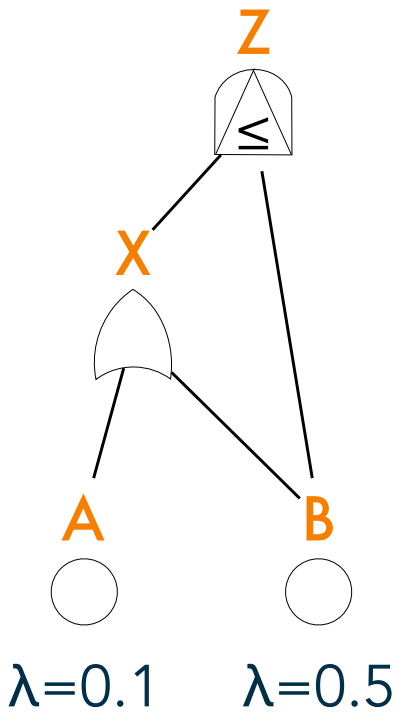
6.

SAFEST
Dynamic Fault Tree Analysis

Commercialisation

Motivation

[Coppit et al., 2000] & [Junges et al., 2016]



- ✓ Unreliability within 1 time unit
  - ✓ DFTCalc **0.0175**
  - ✓ Storm **0.3935**
- ✓ Different semantics for failure propagation
- ✓ Semantic issues when combining gates

Expressing gates with other gates

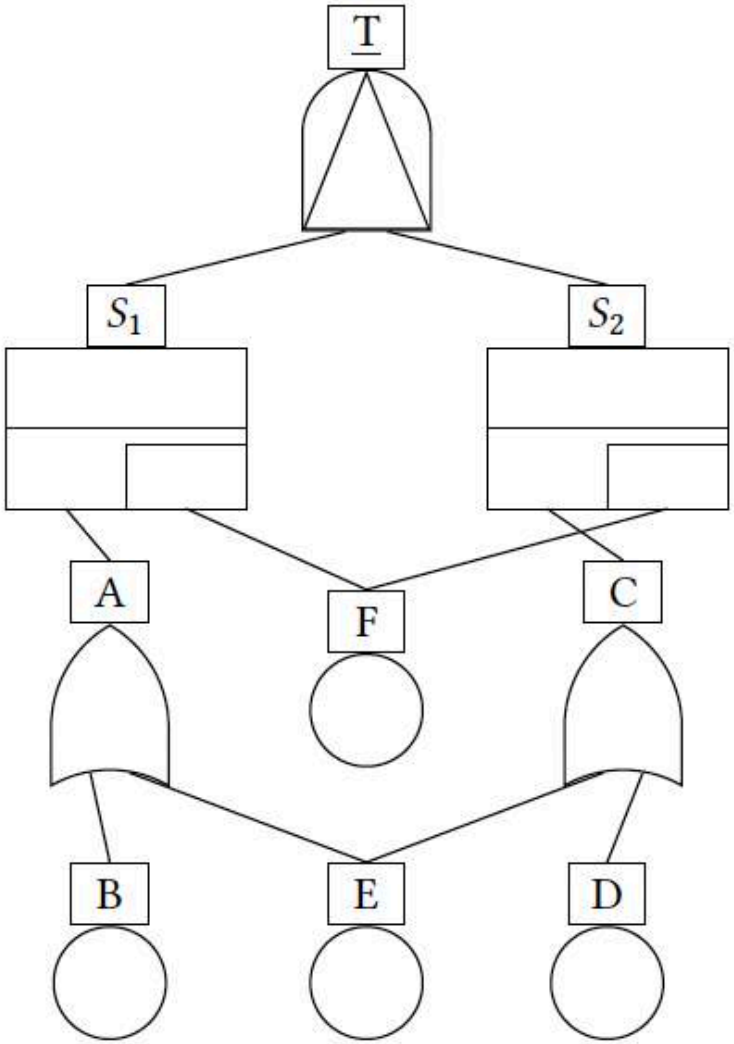
Simultaneous failures in priority gates

SPARE races

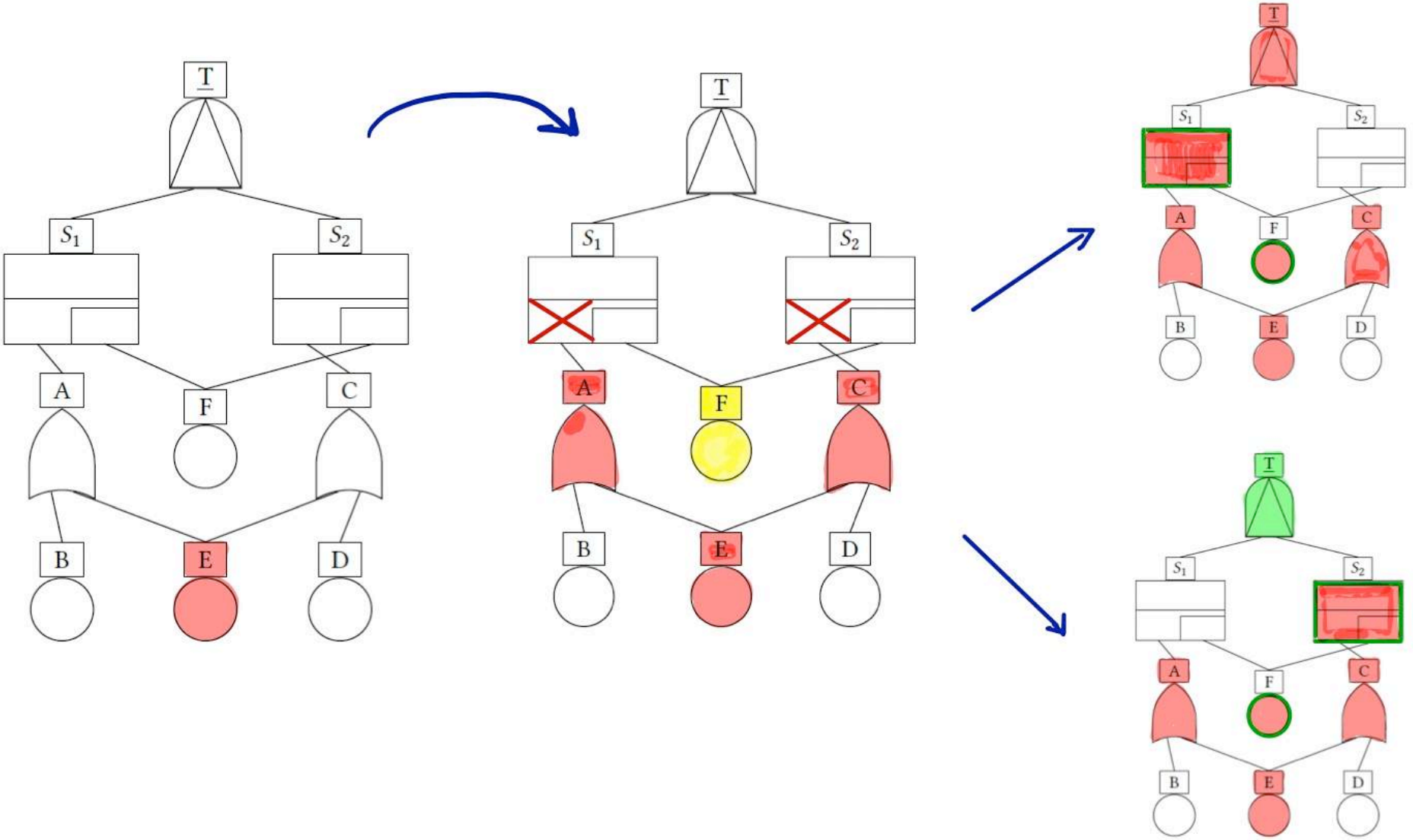
Nested SPARE gates

Combining SEQ and FDEP

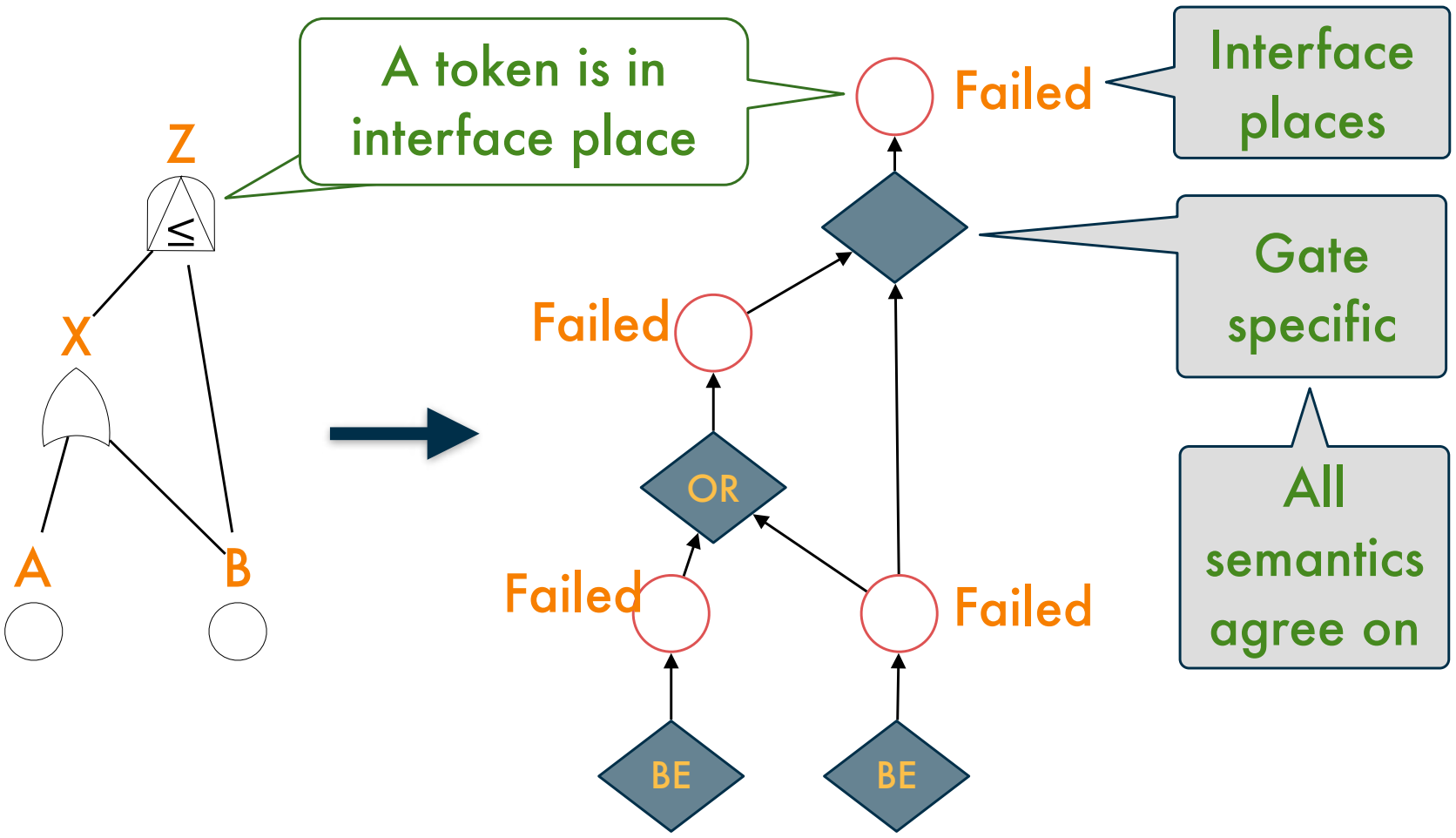
Spare Races



Spare Races

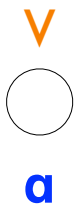


A Petri Net Approach

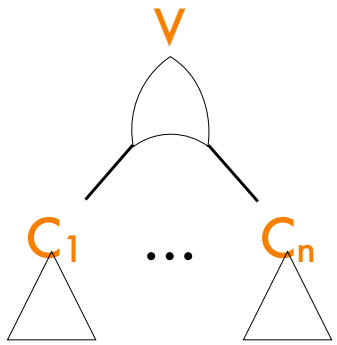


Petri Net Templates

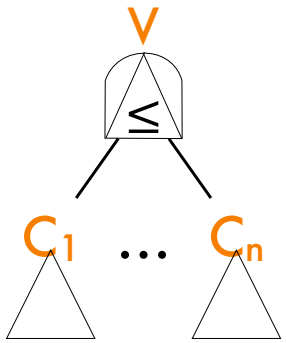
Basic Event

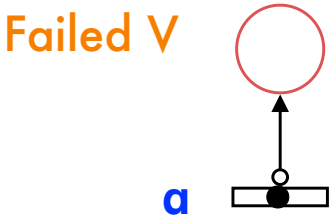


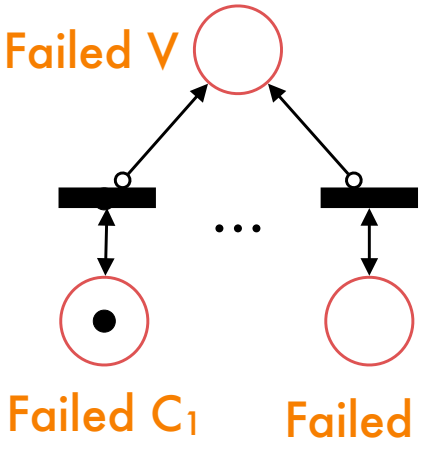
OR-gate

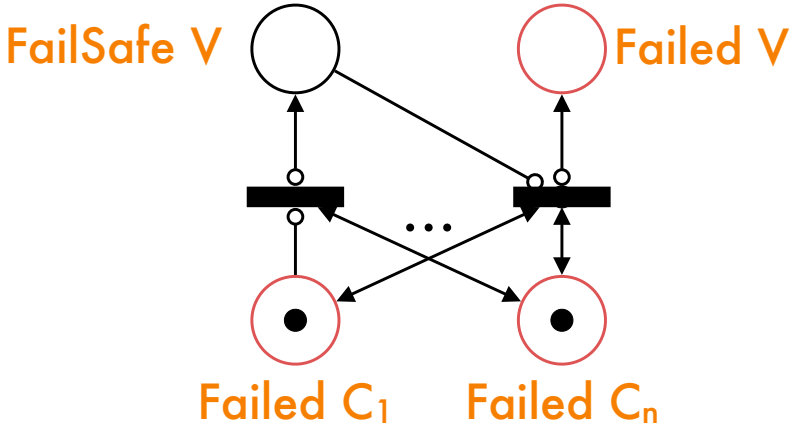


Priority-AND



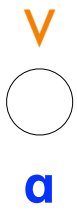




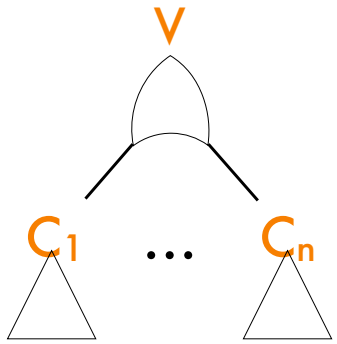


Petri Net Templates

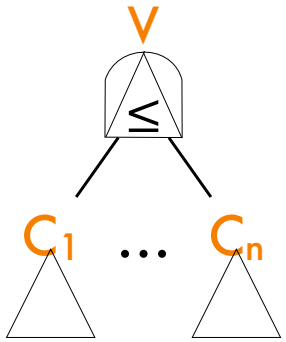
Basic Event

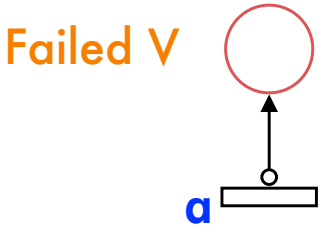


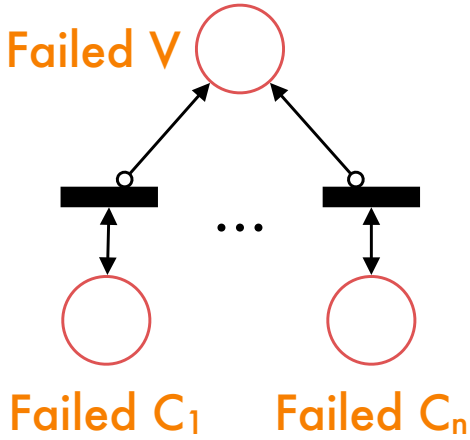
OR-gate

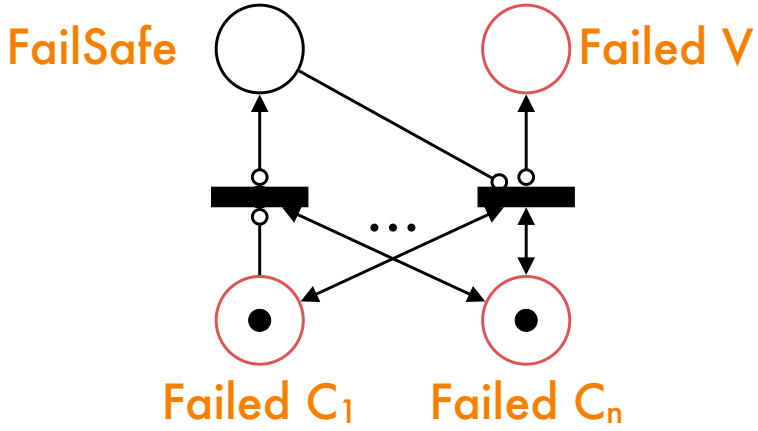


Priority-AND

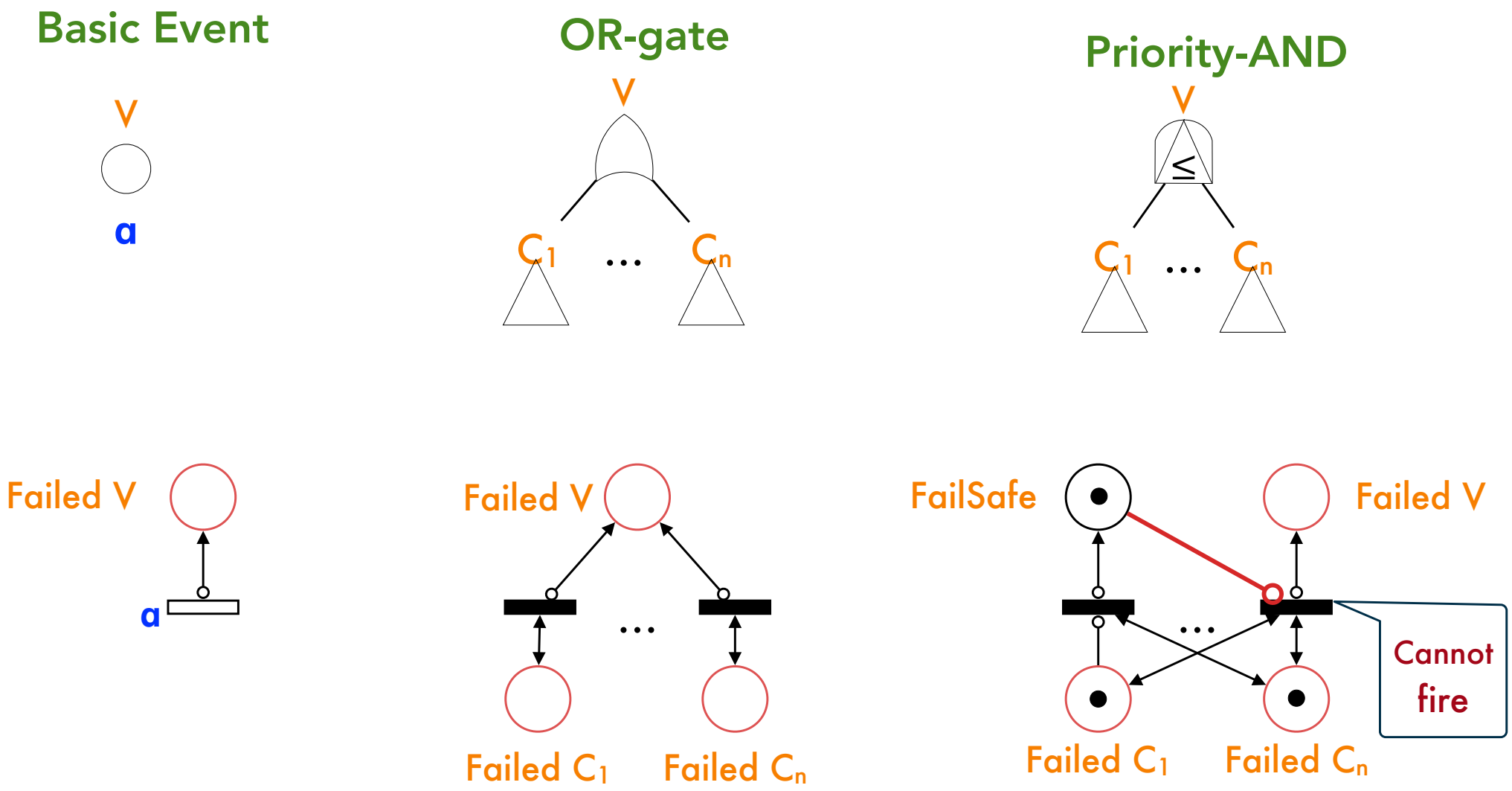




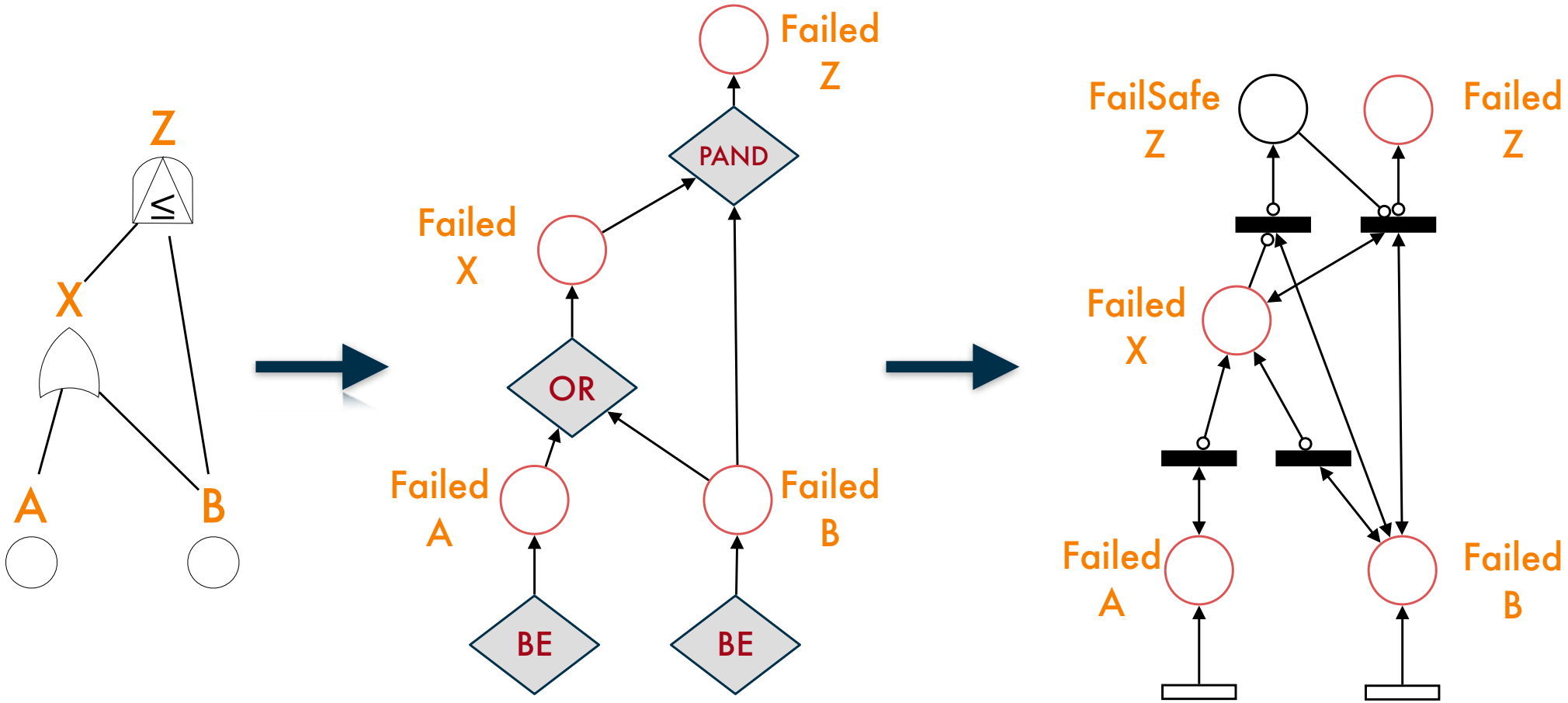




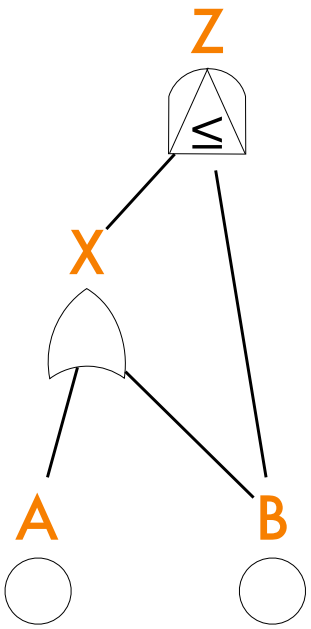
Petri Net Templates



A Petri Net Approach

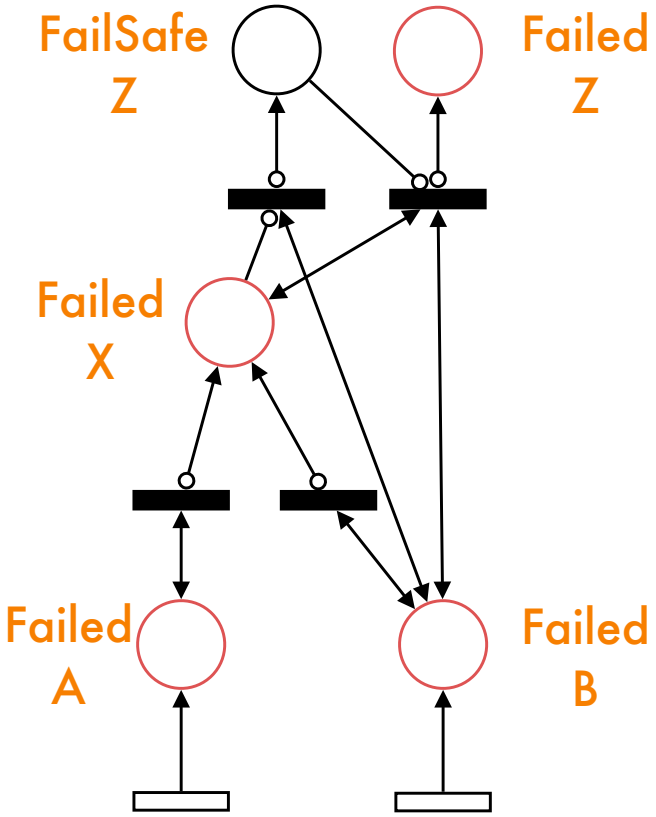


A Petri Net Approach

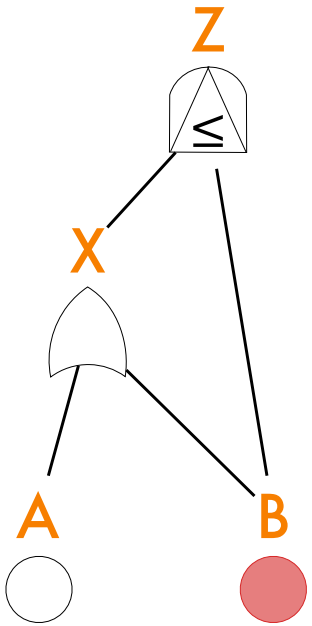


$\tau_1 =$

$\tau_2 =$

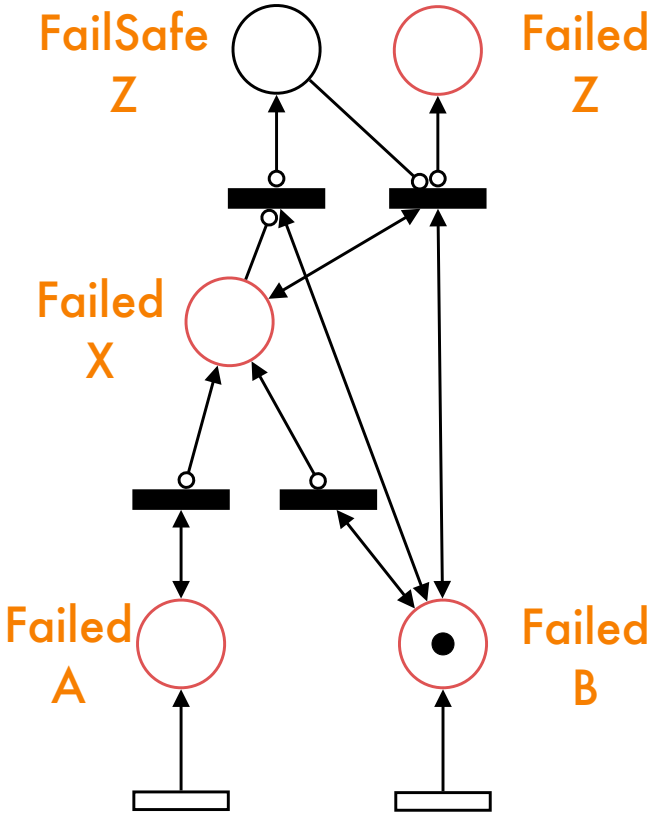


A Petri Net Approach

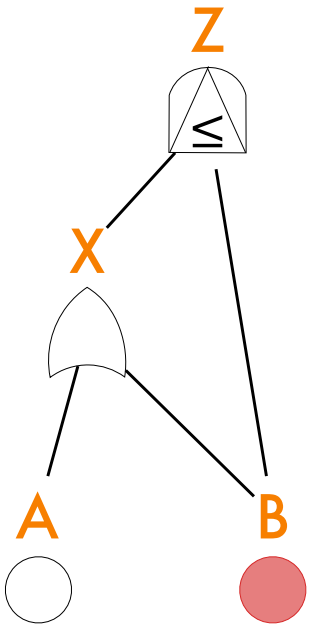


$\tau_1 = f_B$

$\tau_2 = f_B$

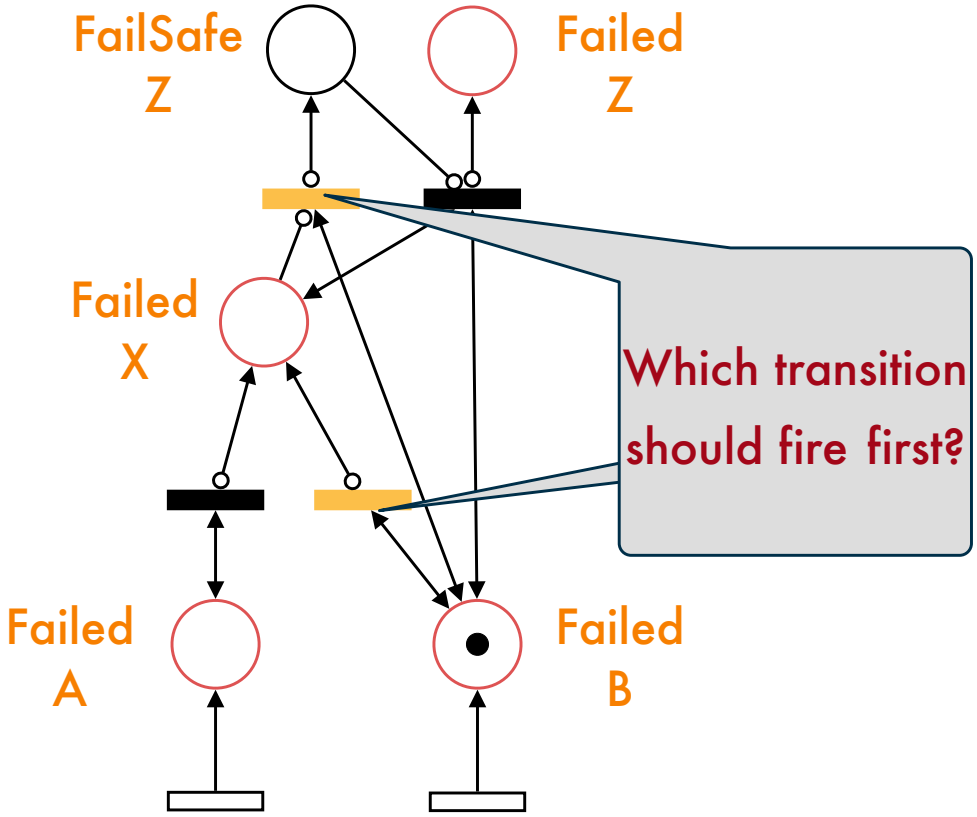


A Petri Net Approach

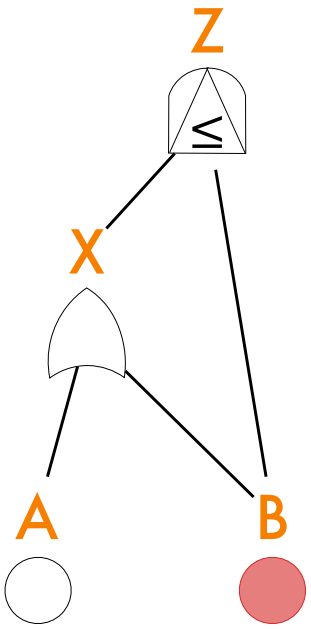


$\tau_1 = f_B$

$\tau_2 = f_B$

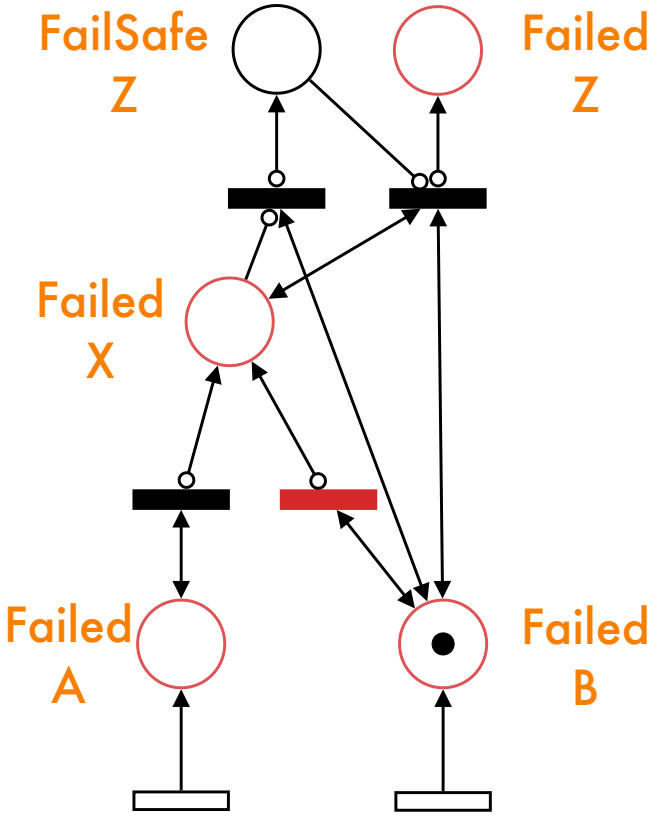


A Petri Net Approach

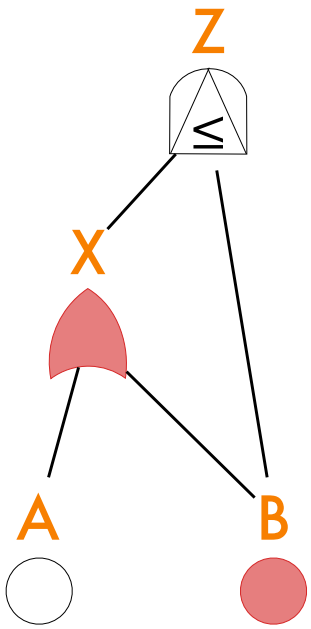


$\tau_1 = f_B$

$\tau_2 = f_B$

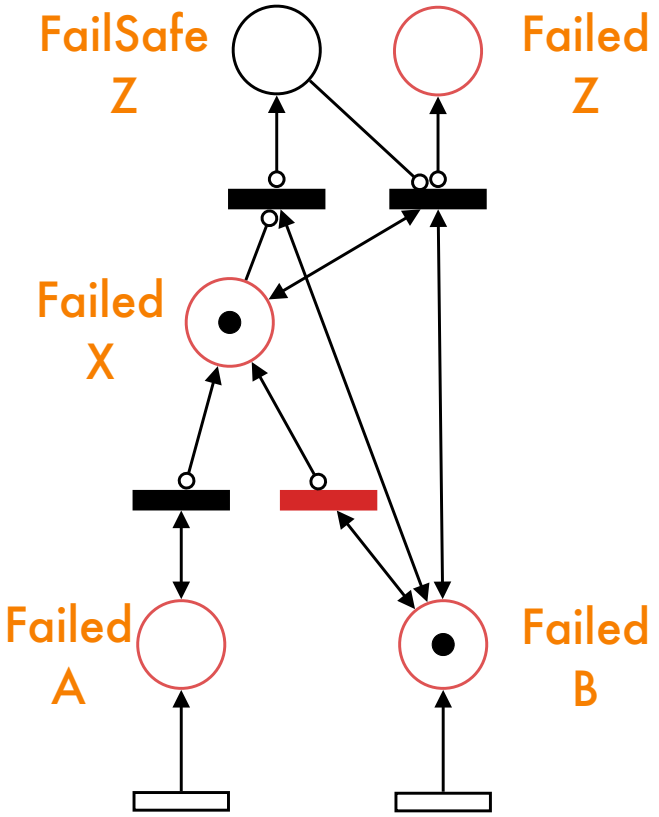


A Petri Net Approach

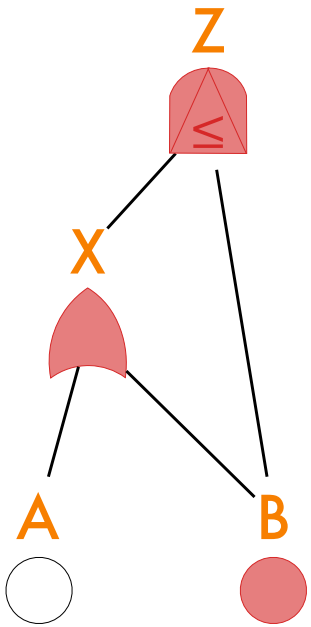


$\tau_1 = f_B\ f_X$

$\tau_2 = f_B$

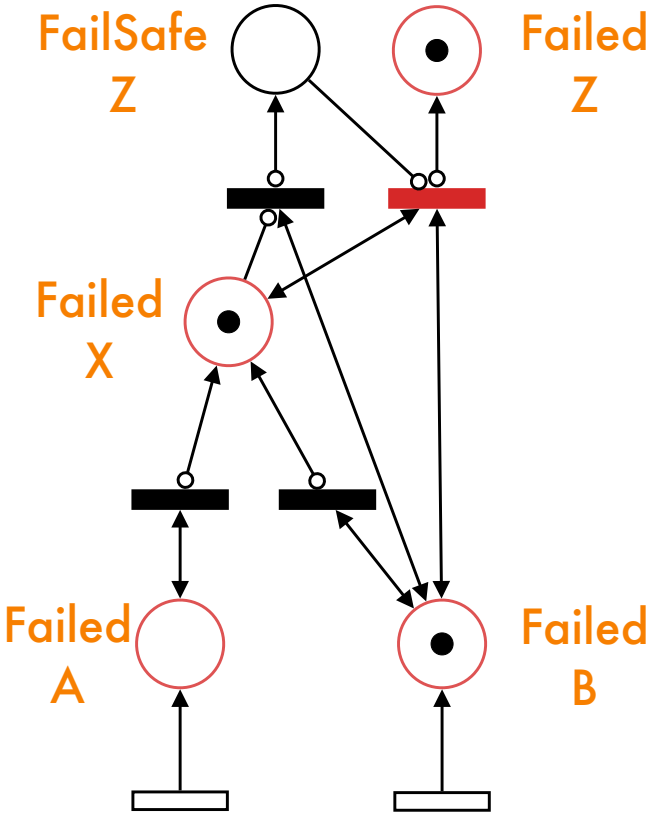


A Petri Net Approach

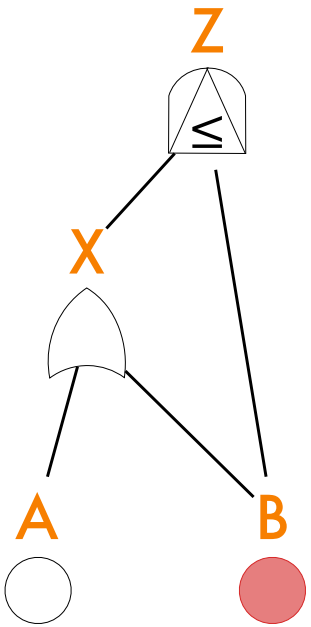


$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B$

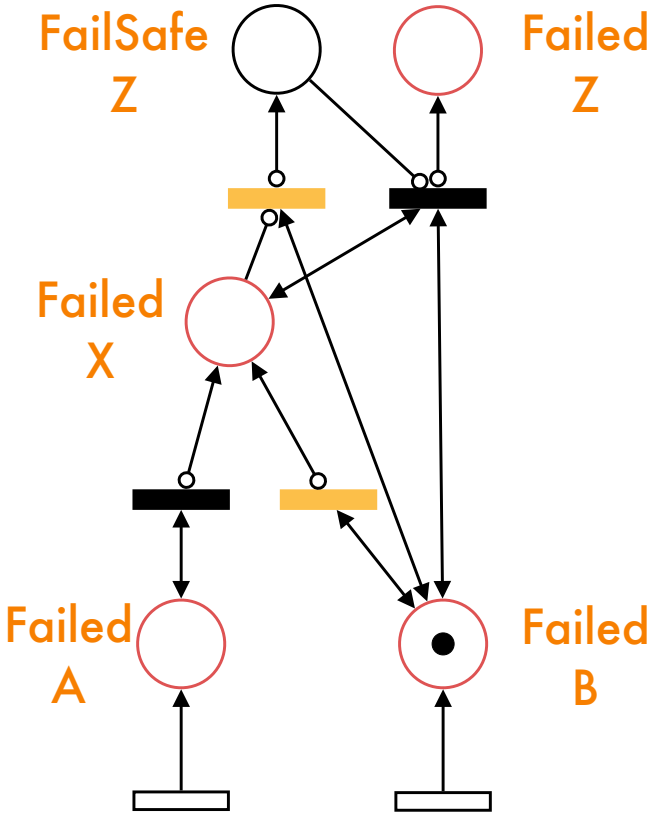


A Petri Net Approach

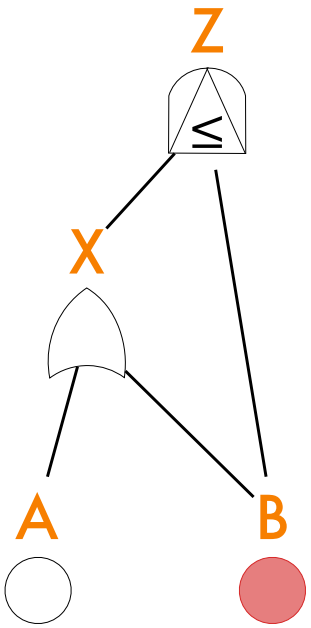


$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B$

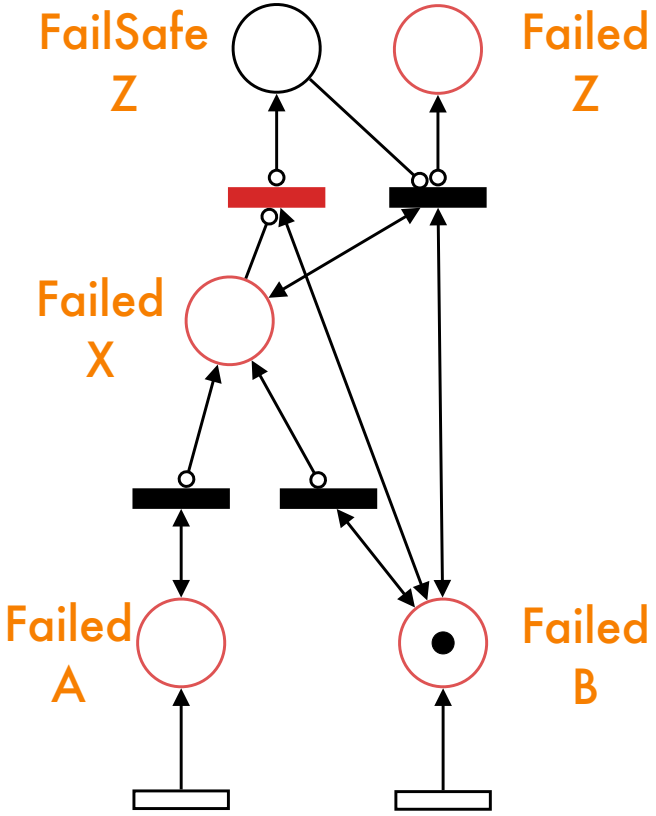


A Petri Net Approach

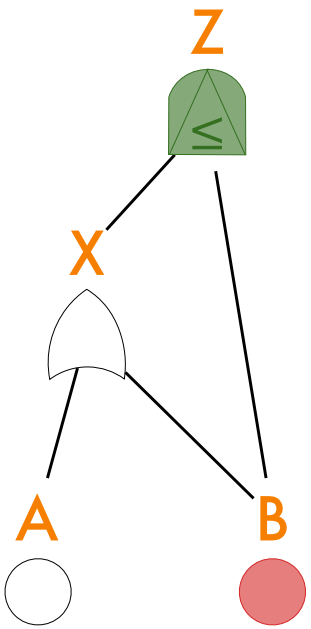


$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B$

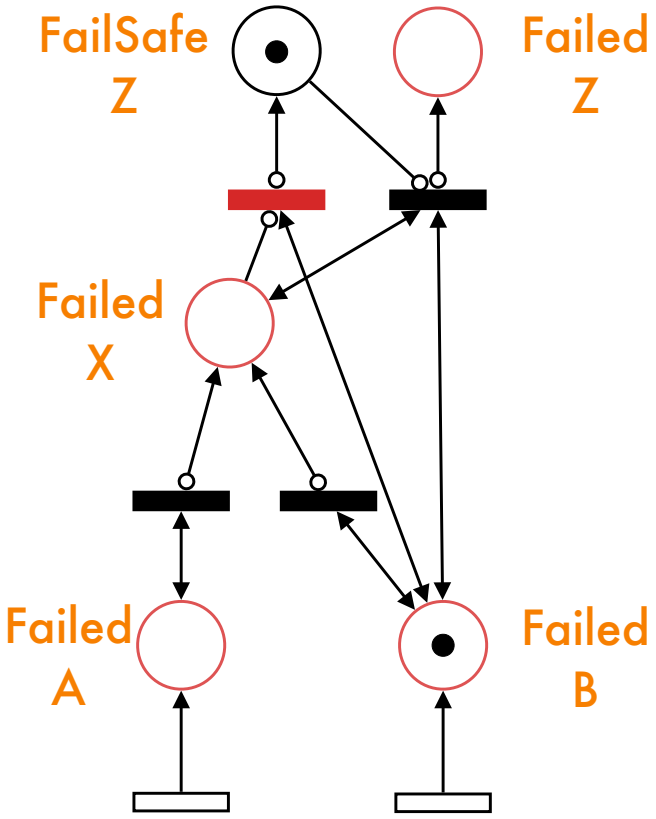


A Petri Net Approach

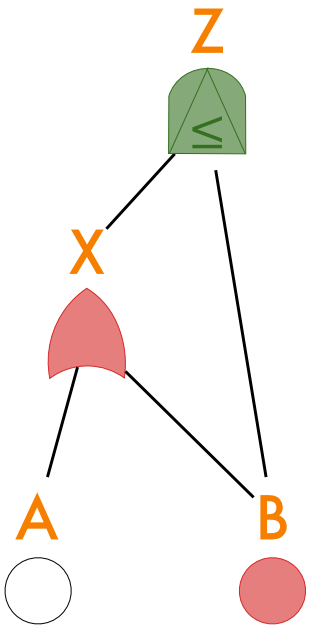


$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B\ \oslash_Z$

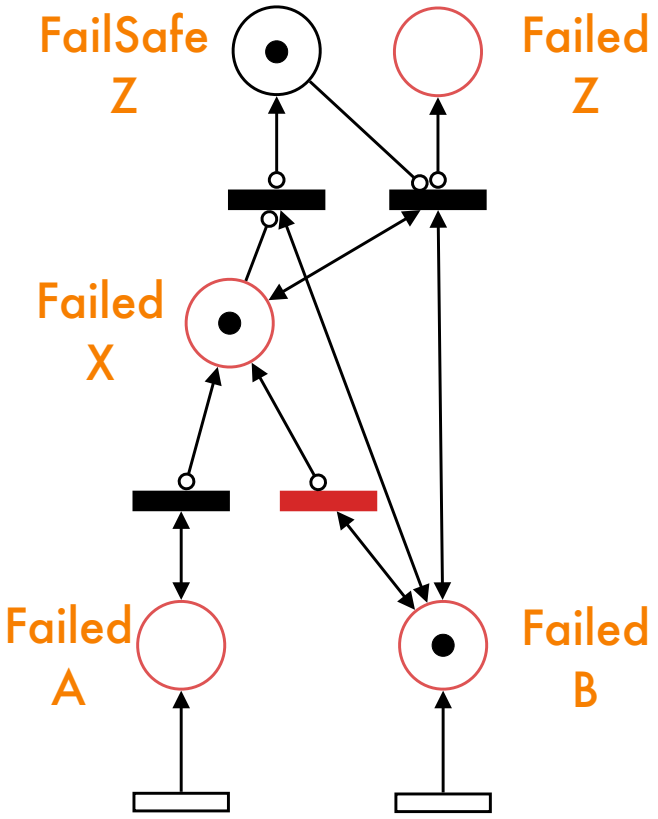


A Petri Net Approach

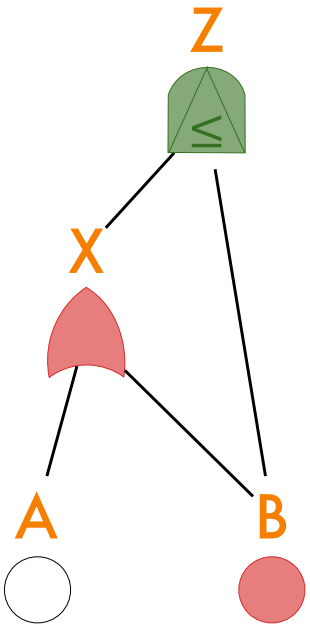


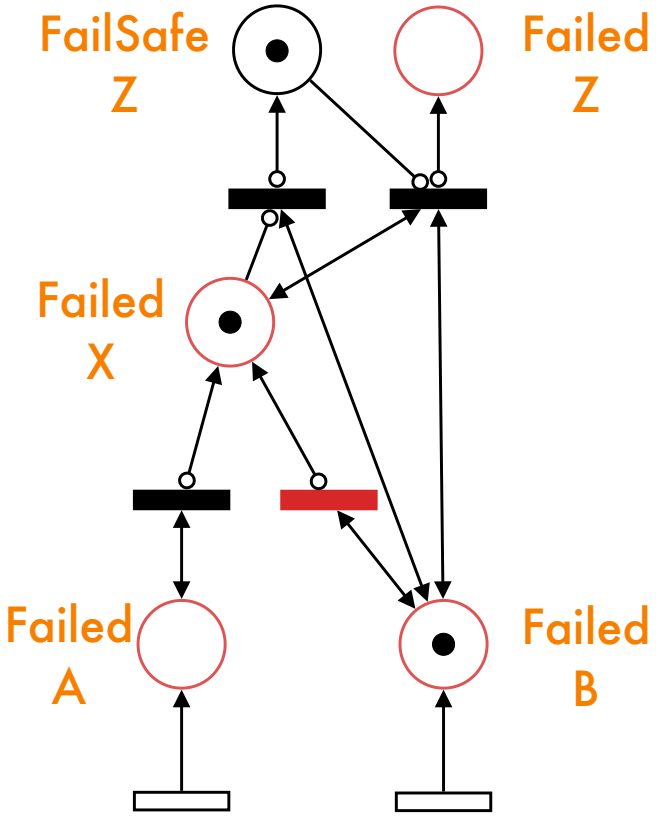
$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B\ \oslash_Z\ f_X$



A Petri Net Approach

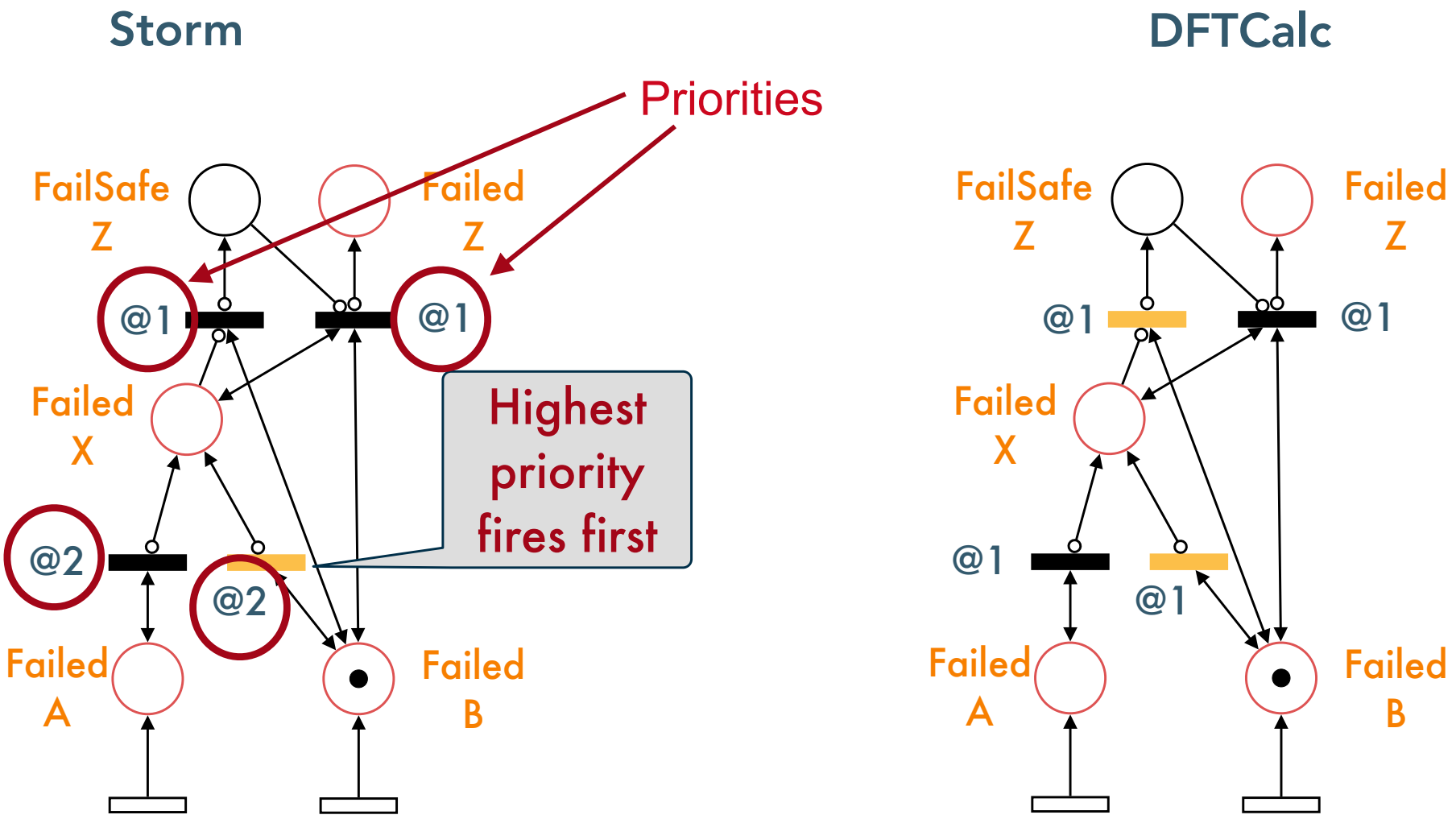




$\tau_1 = f_B\ f_X\ f_Z$

$\tau_2 = f_B\ \oslash_Z\ f_X\ \oslash_Z$

Priorities in Petri Nets



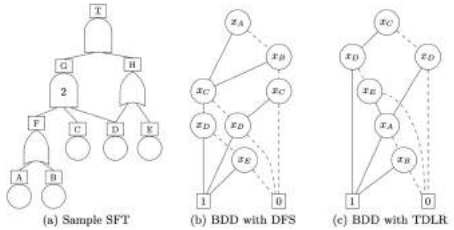
Petri net framework allows to pinpoint semantic differences

Petri Net Semantics Wrap-Up

- Compositional mapping of DFTs onto GSPNs
- Correctness
  - net semantics is equivalent to (intuitive) event trace semantics
- Petri net properties
  - the size of the net is linear in the size of the DFT
  - the resulting nets are bounded
- Our Petri net framework covers all existing DFT semantics
  - differences are in the priority assignment
  - spare races are non-deterministic or probabilistic

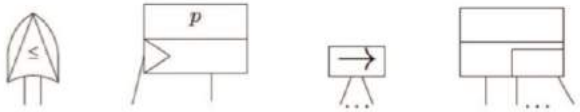
Talk Overview

1.



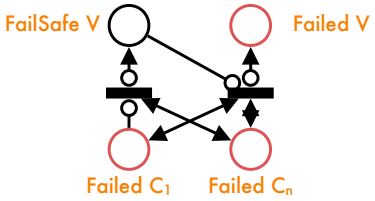
Classical Static Fault Trees

2.



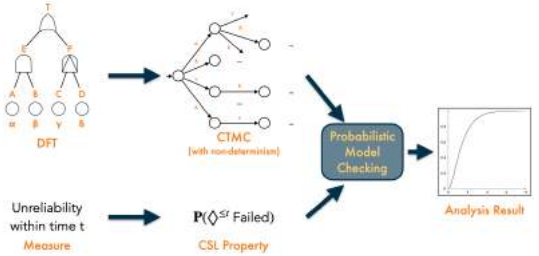
Dynamic Fault Trees

3.



Petri Net Semantics

4.



Scaling Up DFT Analysis

5.

Industrial Case Studies

6.

SAFEST
Dynamic Fault Tree Analysis

Commercialisation

Myths About Dynamic Fault Trees

"Although DFTs are powerful in modeling systems with dynamic failure behaviors,
their quantitative analyses are pretty much troublesome,
especially for large scale and complex DFTs."

[Ge *et al.*, Rel. Eng. Syst. Safe, 2015]

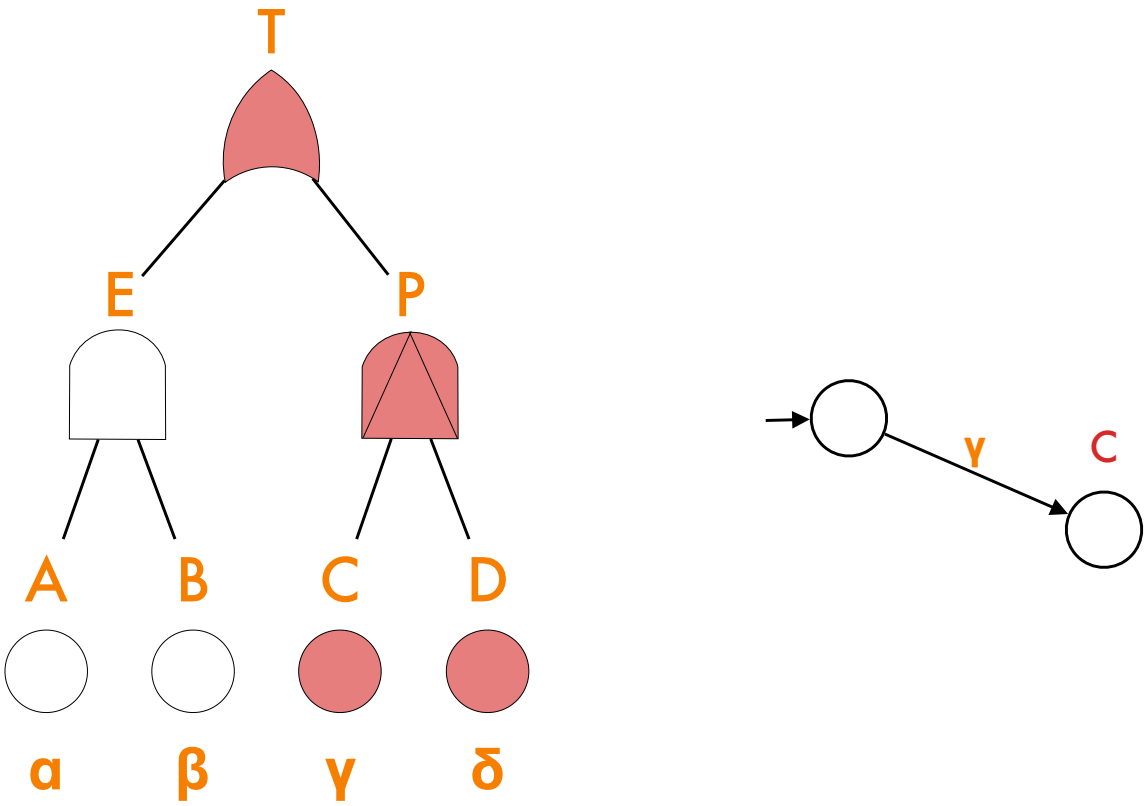
"Although many extensions of fault trees have been proposed,
they suffer from a variety of shortcomings.
In particular, even where software tool support exists,
these analyses require a lot of manual effort."

[Kabir, Expert Syst. Appl., 2017]

These are all myths. **Scalable** and **fully automated** DFT analysis is possible.

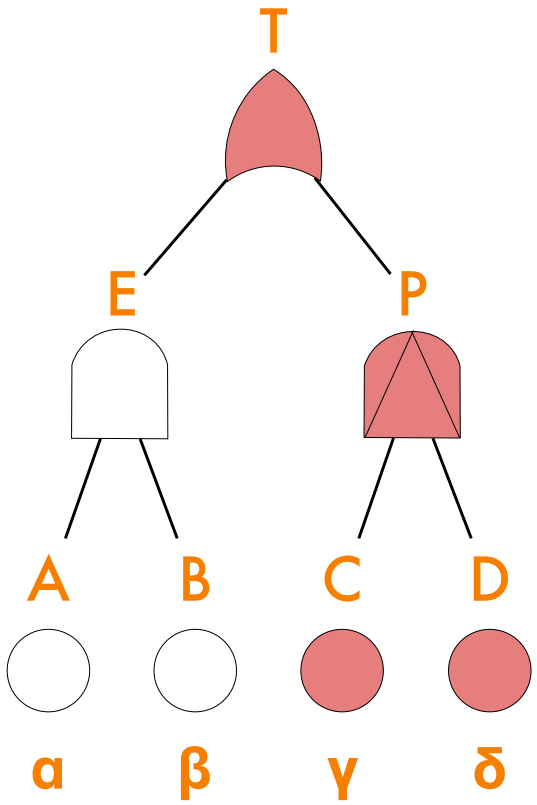
State Space Generation

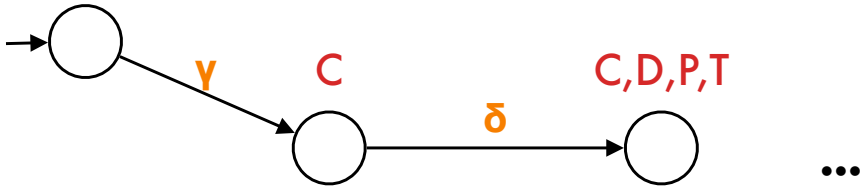
[Manian et al., 1999]



State Space Generation

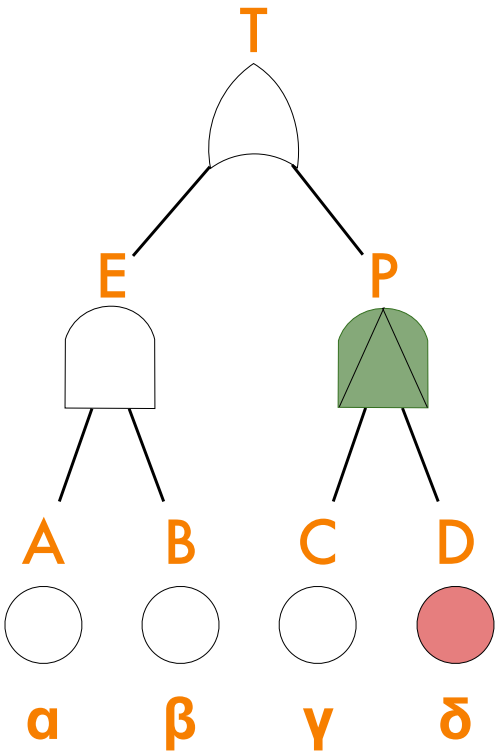
[Manian et al., 1999]

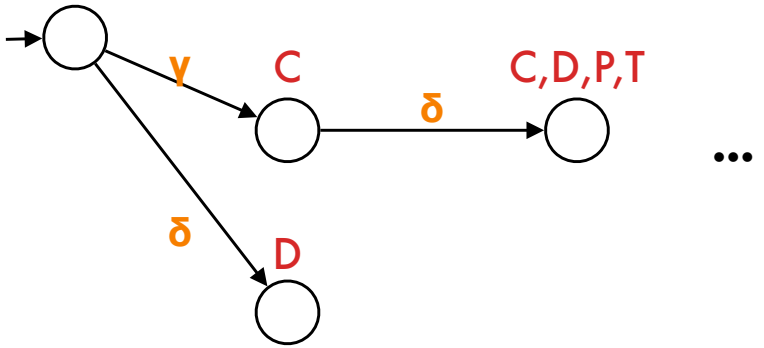




State Space Generation

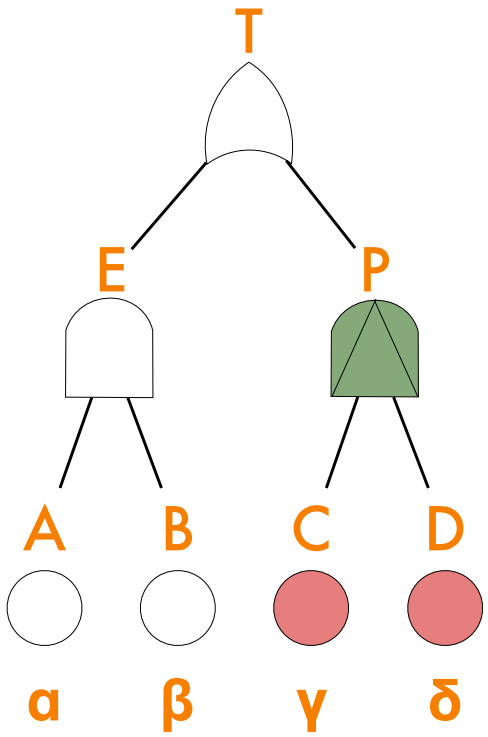
[Manian et al., 1999]

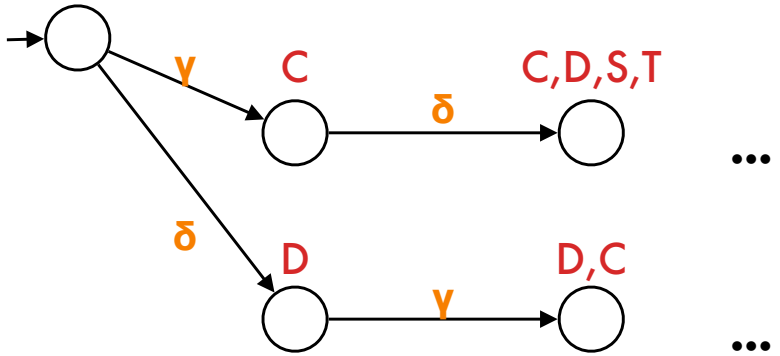




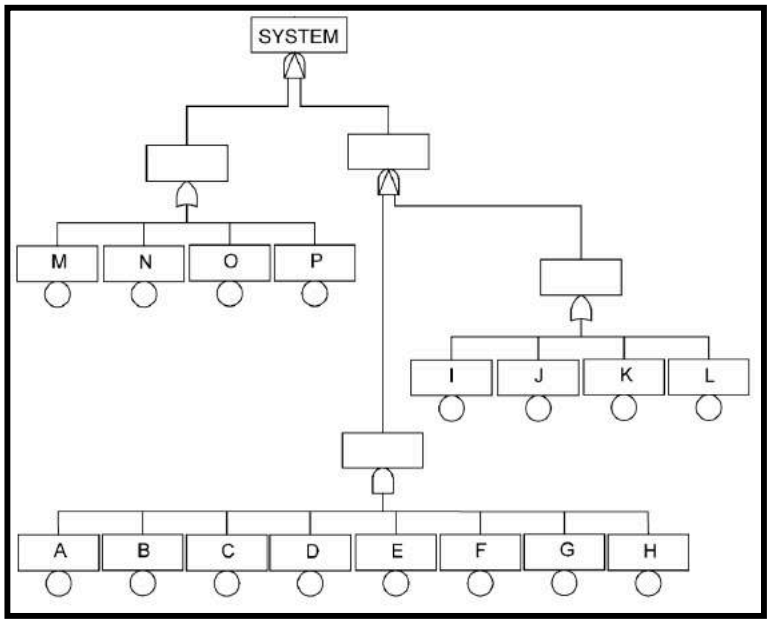
State Space Generation

[Manian et al., 1999]





State Space Explosion Problem?



Fictitious system DFT

„[The example was created to] make the corresponding Markov chain of this tree drastically large and practically impossible to solve without resorting to simplifying assumptions and/or approximations"

[Boudali & Dugan 2005]

Storm

Naive state-space generation

✓ 66,001 states

✓ Analysis in 1.073 seconds

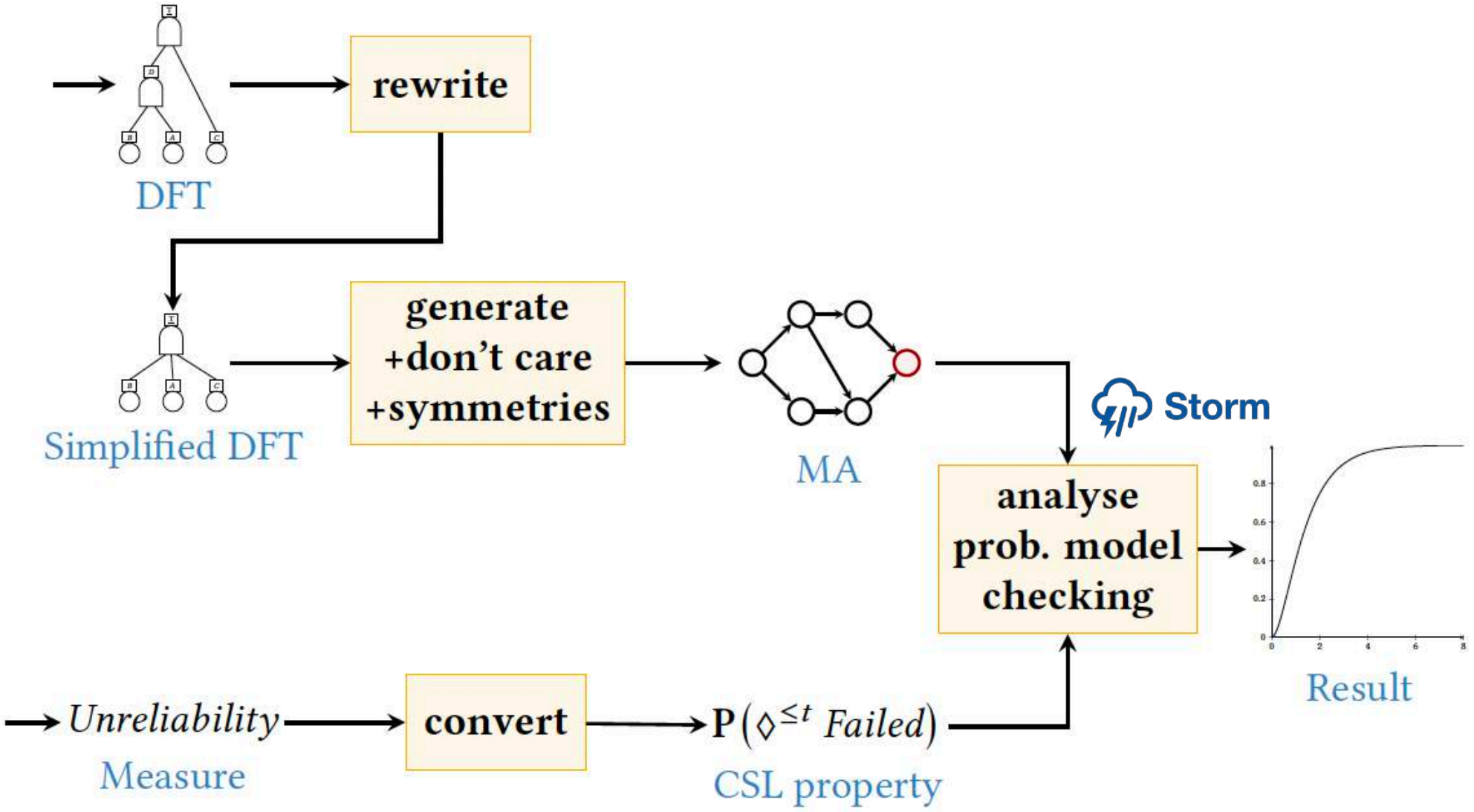
Optimised state-space generation

✓ 514 states

✓ Analysis in 0.015 seconds

Exact result

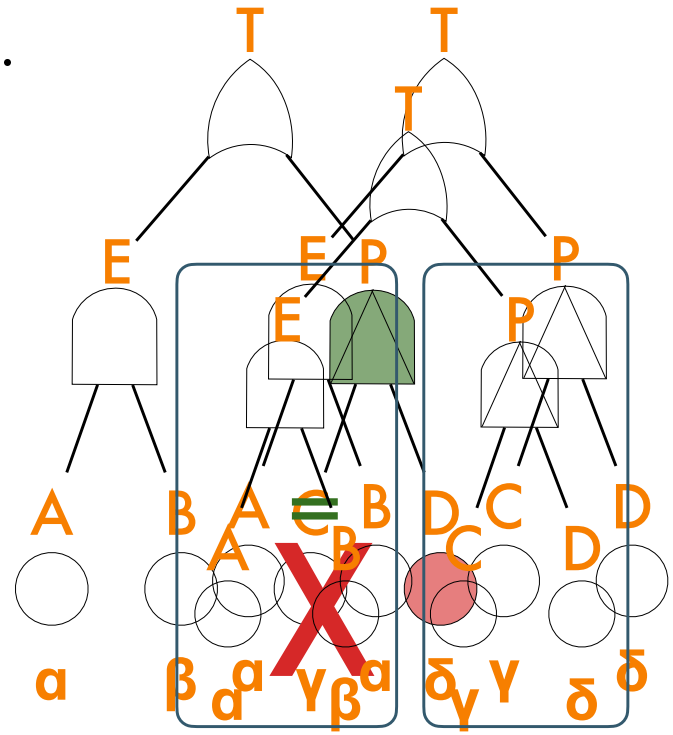
What's The Secret?



Optimisations

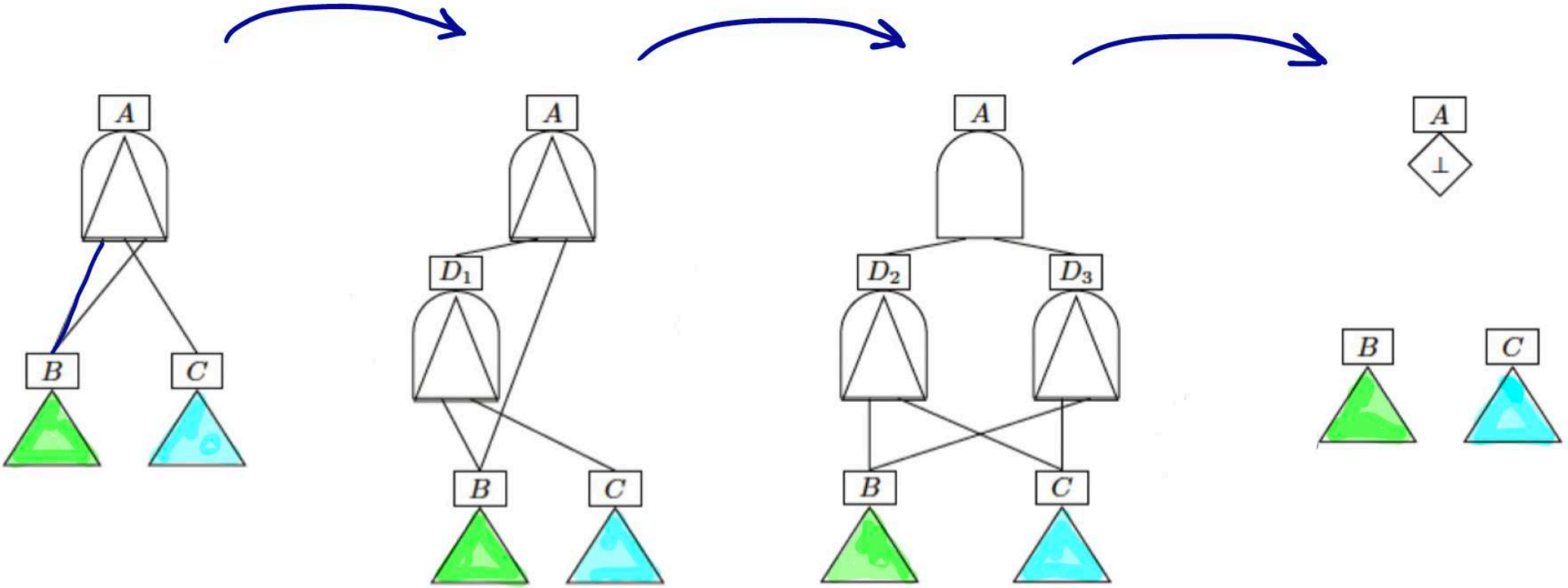
All these techniques were revised, improved and extended.

- **Don't Care** [Bouissou, Bon, 2003] for BDMP, [Yevkin, 2016]
  - exact status of element is irrelevant for further analysis
  - e.g., fail-safe, completely failed, etc.
- **Symmetries** [Bobbio, Codetta-Raiteri, 2004]
  - present through redundancies
  - merge states which are symmetric
- **Modularisation** [Gulati, Dugan, 1997]
  - analyse sub-parts independently
- Eliminate **spurious non-determinism**
- **Rewrite (simplify) DFTs** before analysis
- **Partial state-space generation**



Rewriting DFTs

[Junges et al., FAC 2017]



Rewriting DFTs

- Context-sensitive rewrite rules
- Interpreted as graph rewriting rules
- Catalogue of 29 rewrite rules
  - flattening of AND, OR and PAND
  - removal of conflicting PAND gates
  - pushing up OR and AND gates
- Correctness [Elderhalli et al., SEFM 2019]
  - 22 rules were proven correct using HOL4
  - 1,500 lines of code and about 80 hours effort
  - no formalisation of SPARE and FDEP

GROOVE
GRaphs for Object-Oriented VErification

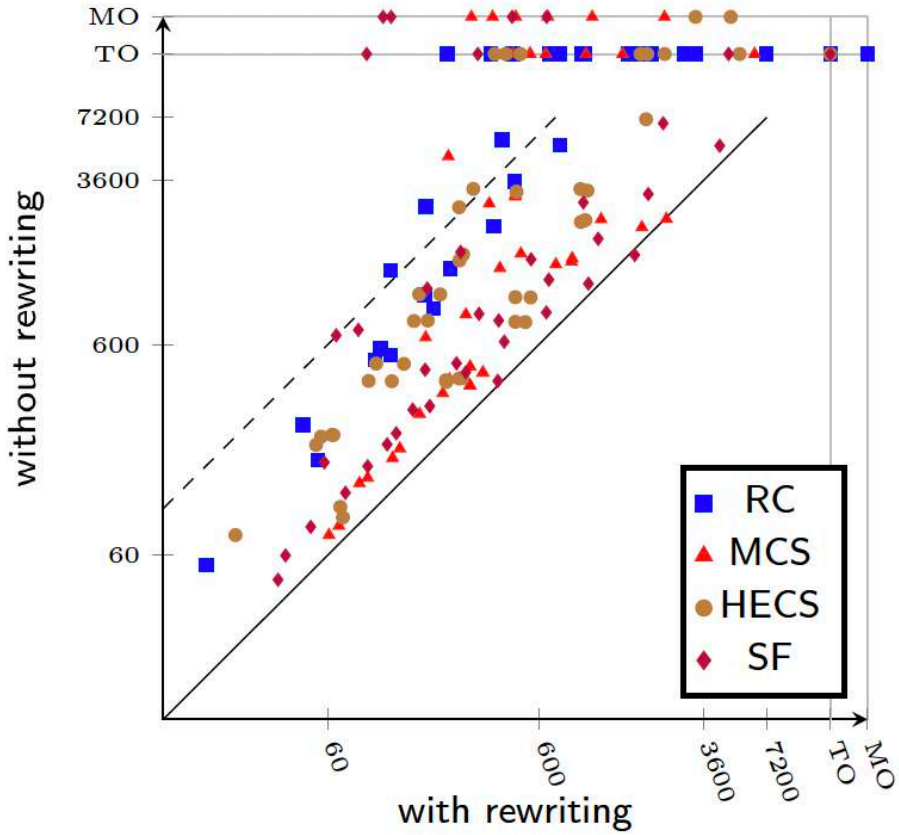
Experimental evaluation

- Analysis with DFTCalc

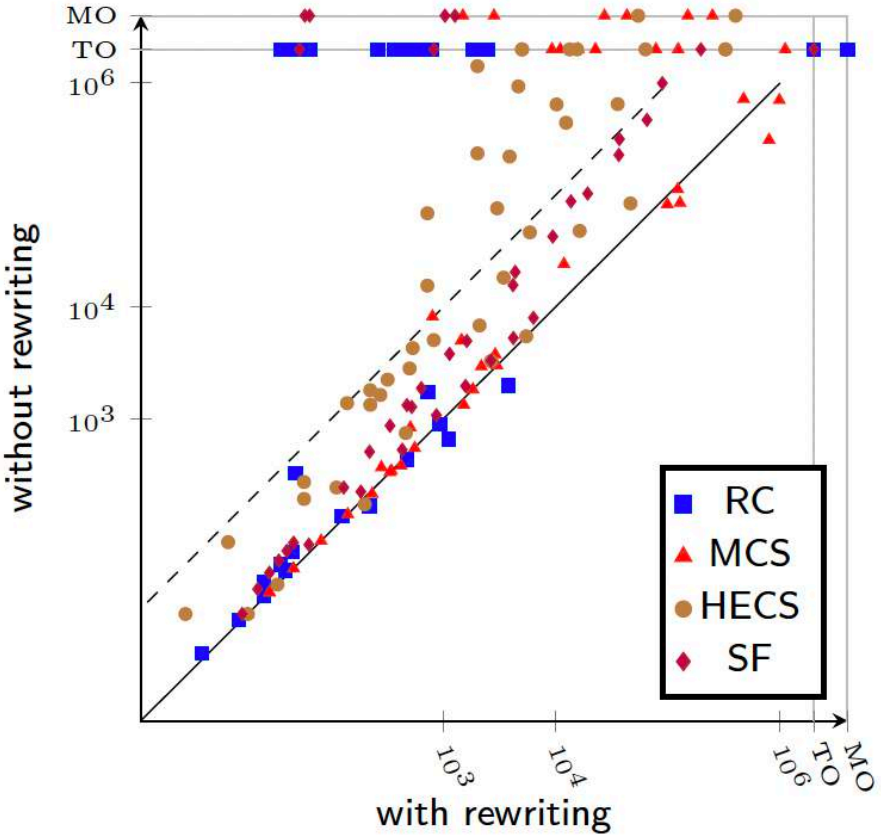
inside

- could solve 27% more examples with rewriting

Time

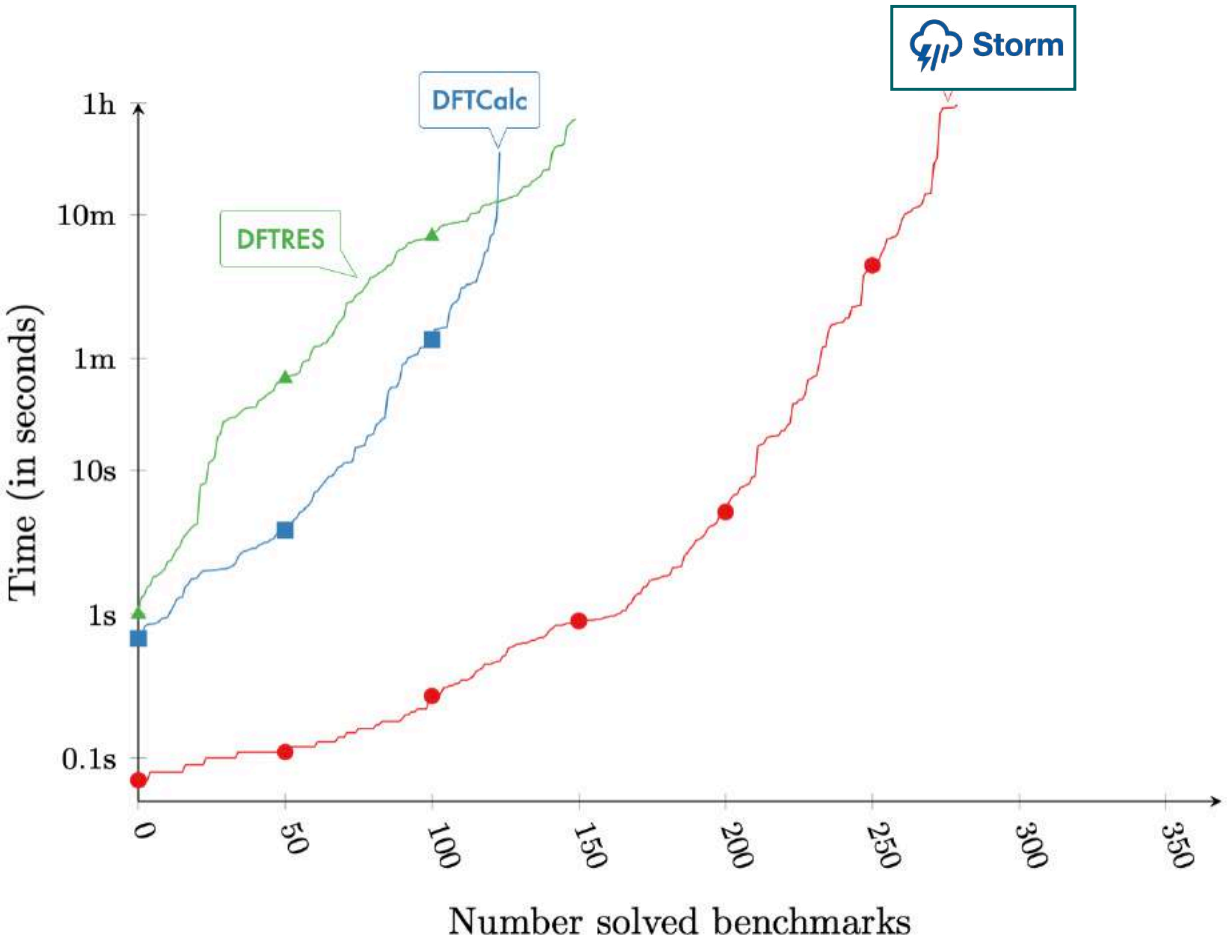


State space



Evaluation: DFT Analysis Times

https://dftbenchmarks.utwente.nl/

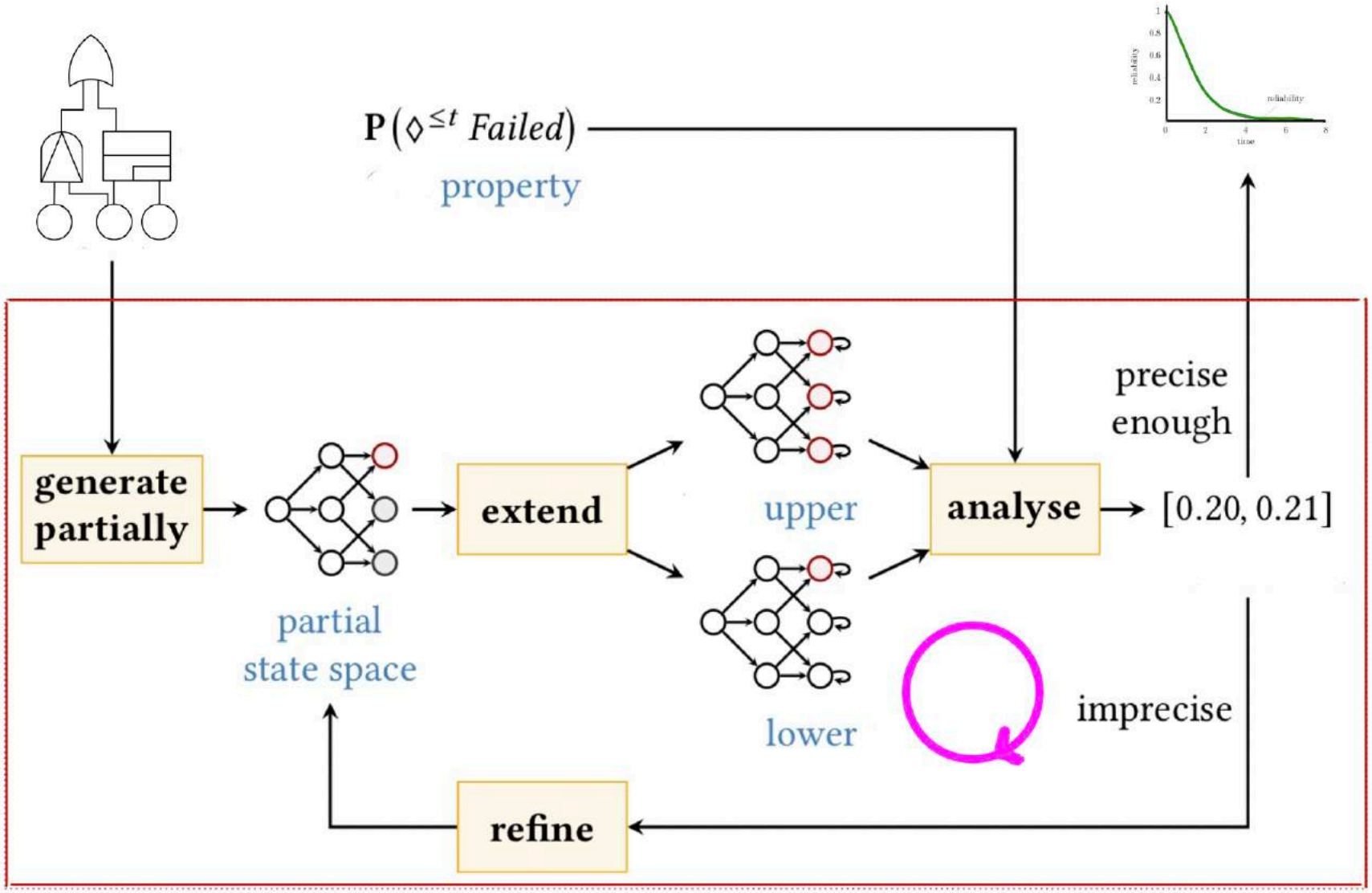


- ✓ Public FFORT benchmark suite
- ✓ Unreliability and MTTF
- ✓ 369 benchmarks
- ✓ Comparison to
  - ✓ DFTRes (2020, simulation)
  - ✓ DFTCalc (2013, compositional)
- ✓ 2.1 GHz, 16 GB RAM
- ✓ Error bound: $\mathbf{10^{-4}}$

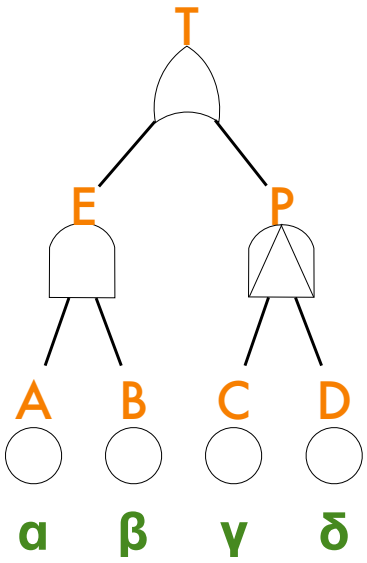
Storm solves more benchmarks in 1 second than others in 1 hour

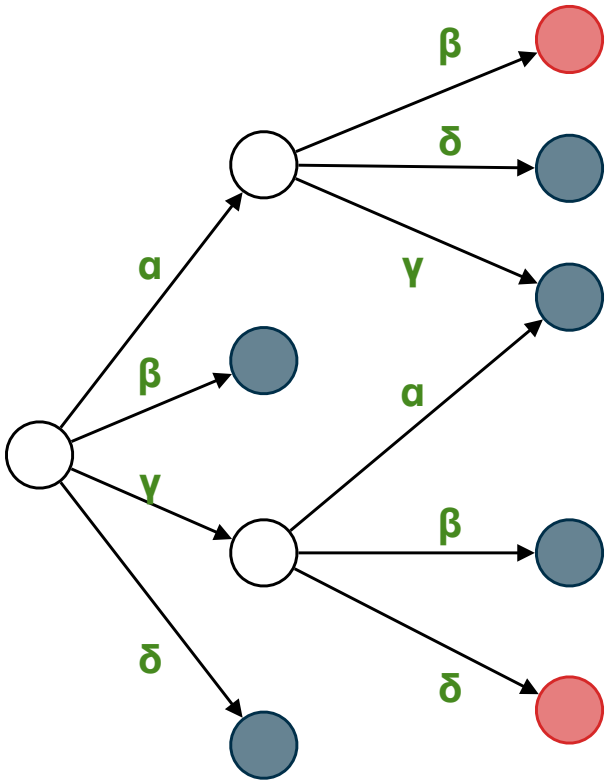
Partial State-Space Generation

[Volk, Junges, K., IEEE TII 2018]



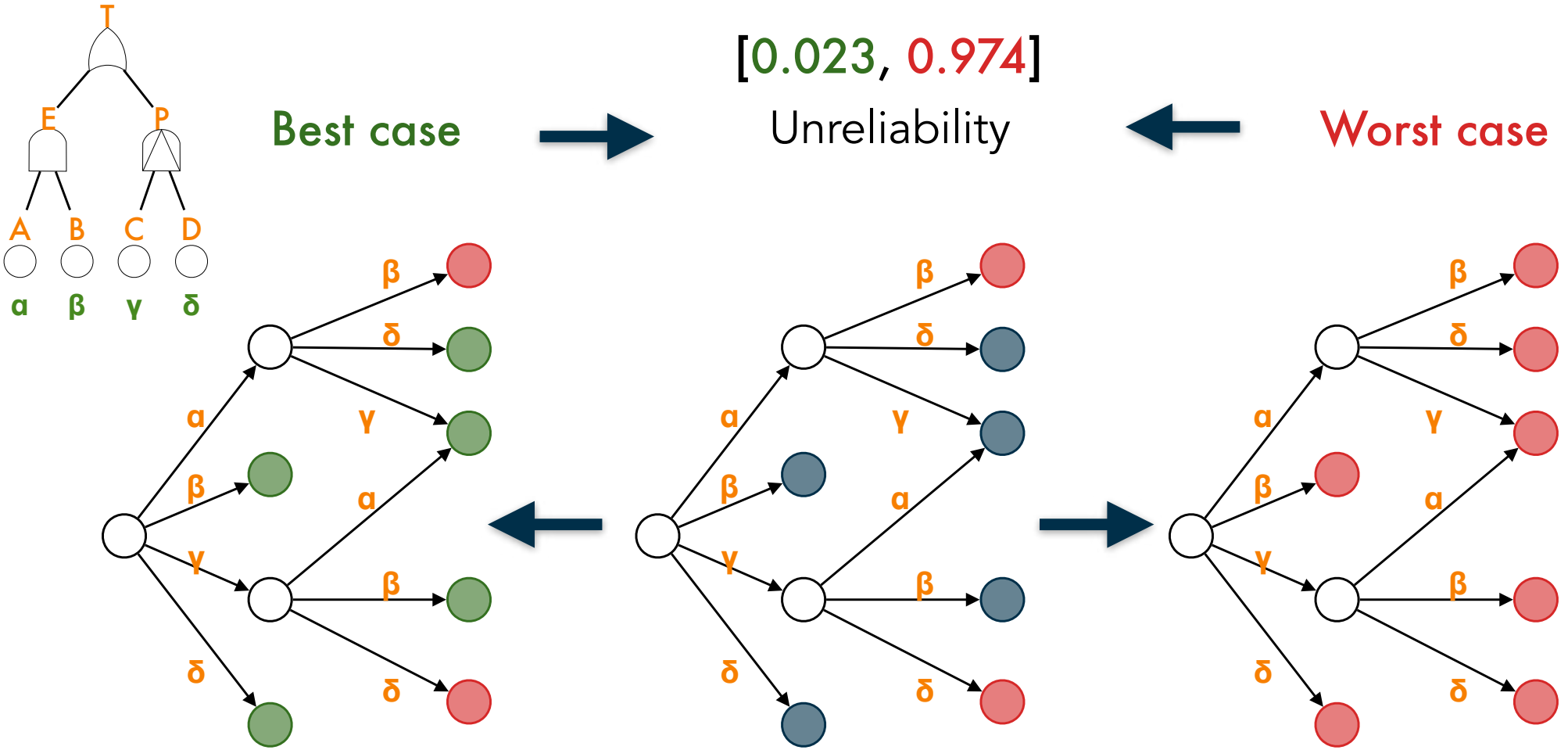
Partial State-Space Generation



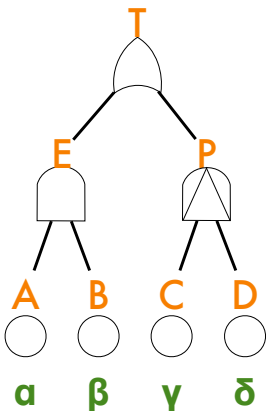


unexplored state

Partial State-Space Generation

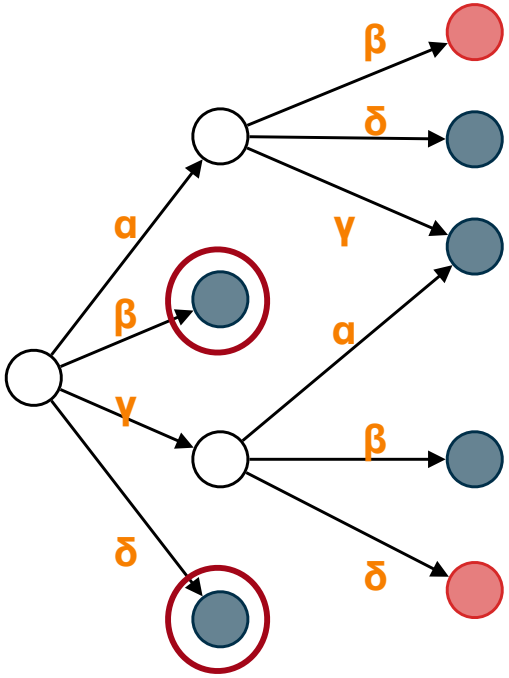


Partial State-Space Generation



[0.023, 0.974]

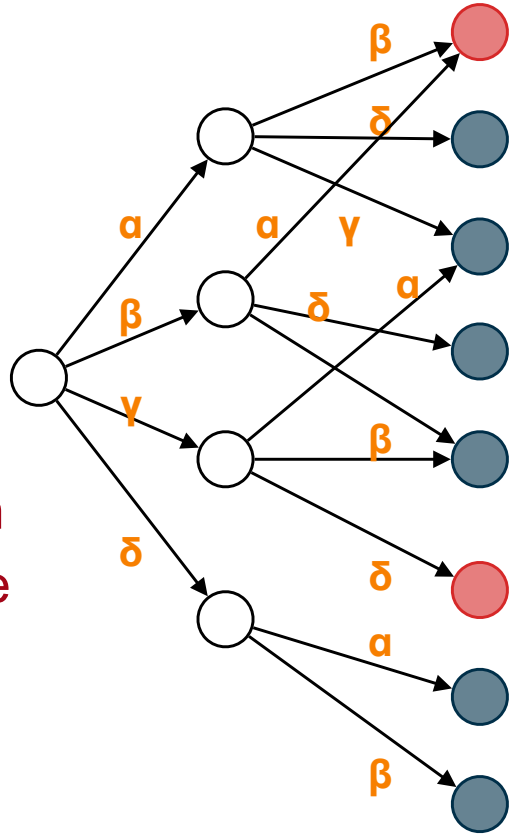
Unreliability



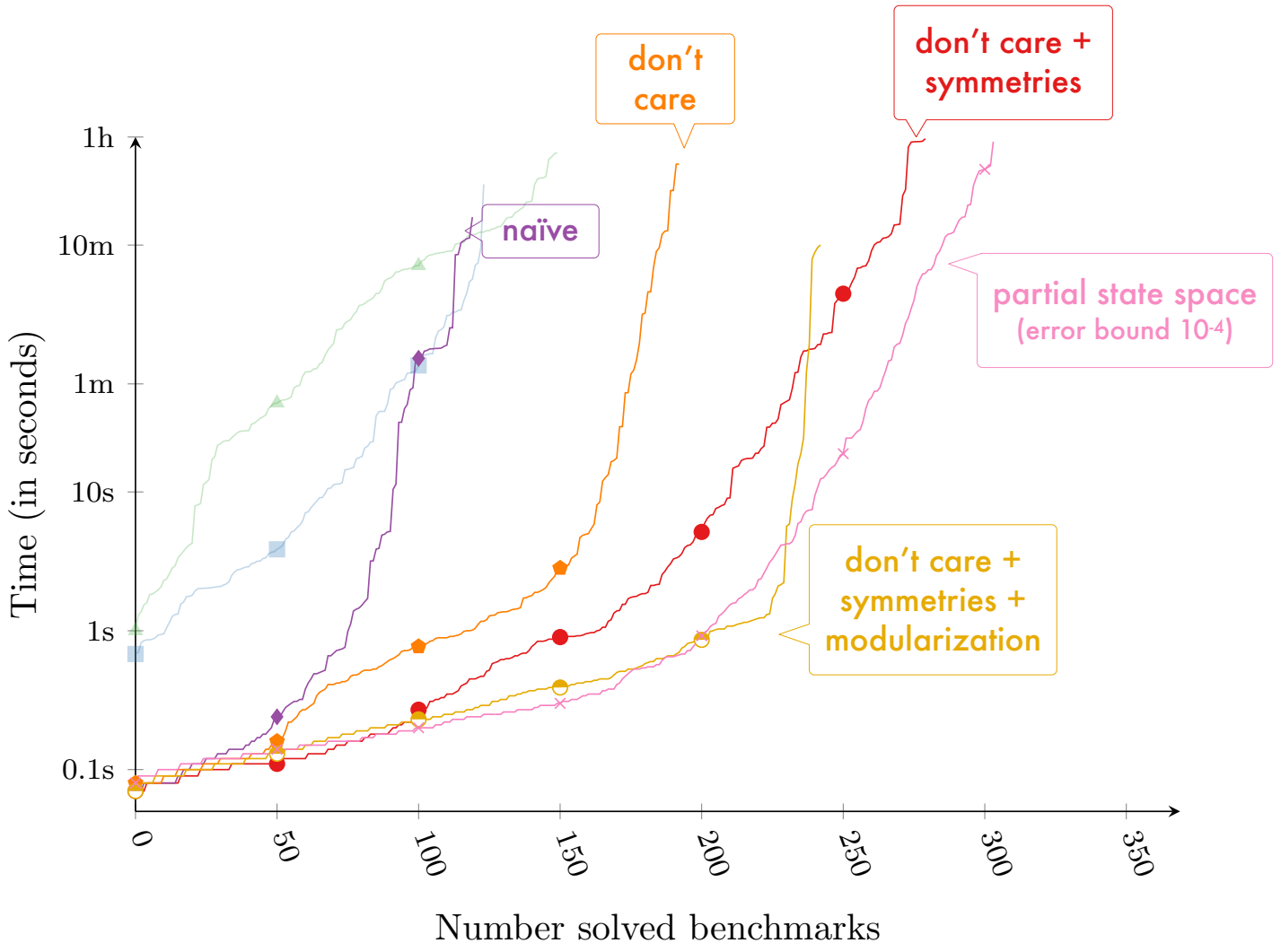
Heuristics which states to explore

[0.145, 0.535]

Unreliability



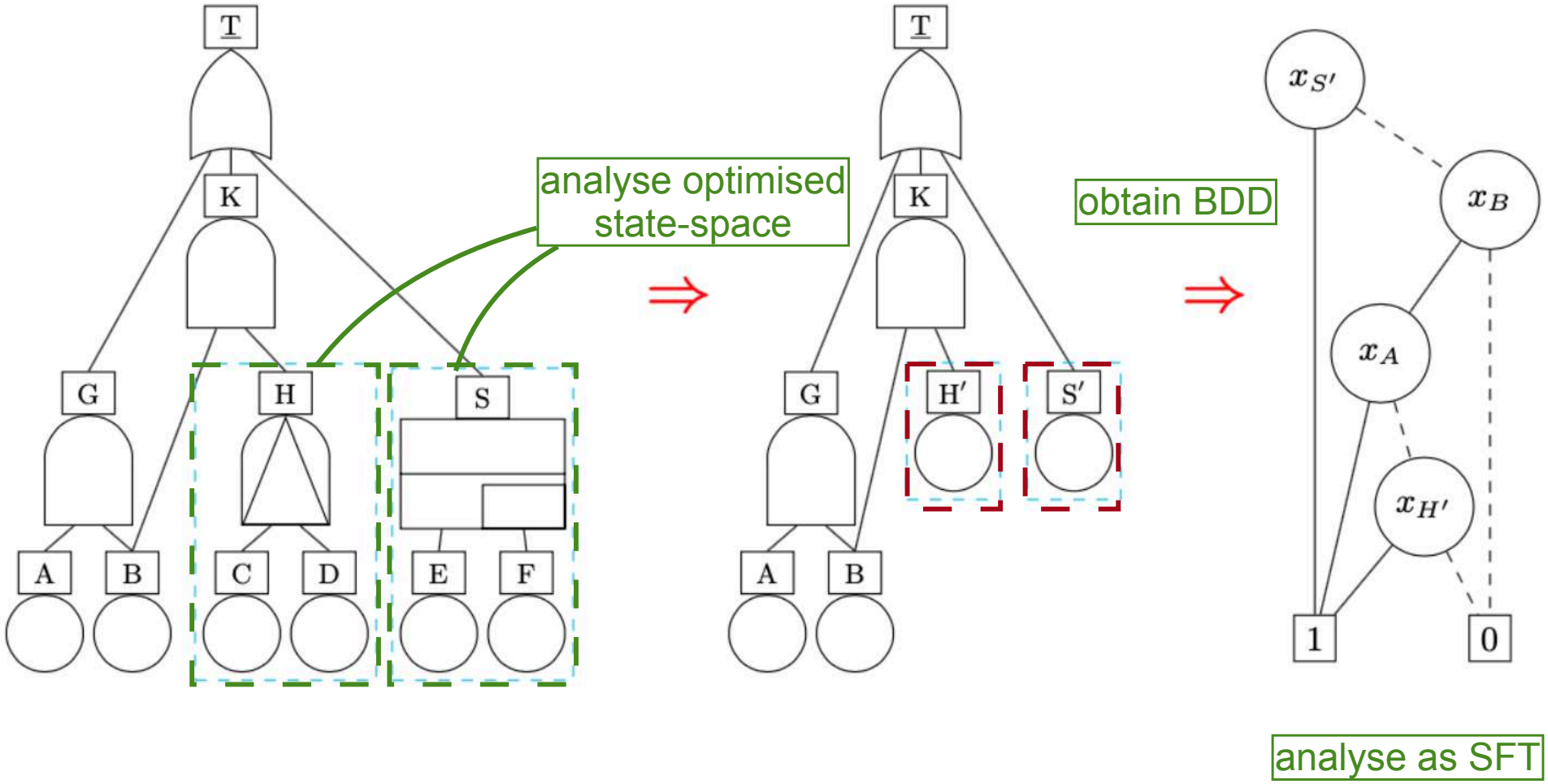
The Effect of the Optimisations



The combination of all optimization techniques pays off

What If DFTs Contain Large Static Parts?

[Basgöze et al., NASA FM 2022]



Experiments: DFTs with Static Parts

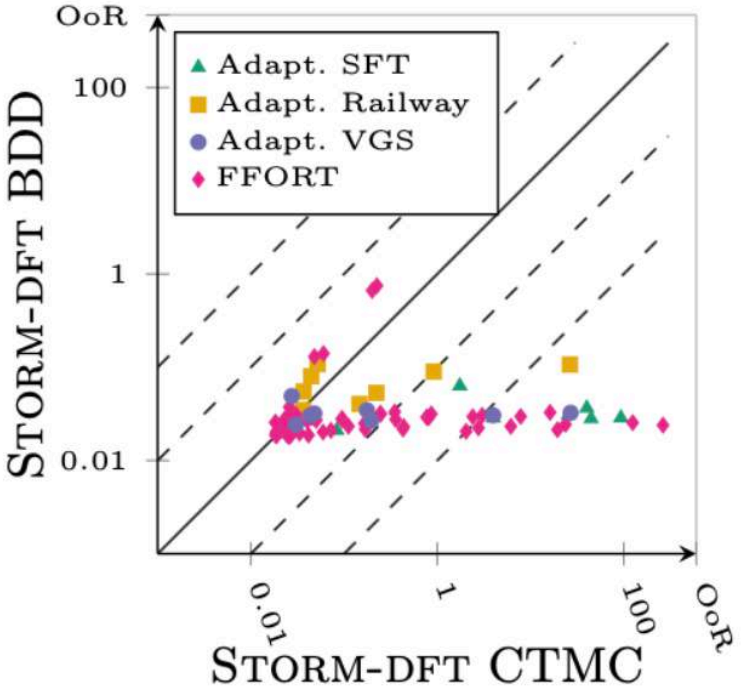
[Basgöze et al., NASA FM 2022]

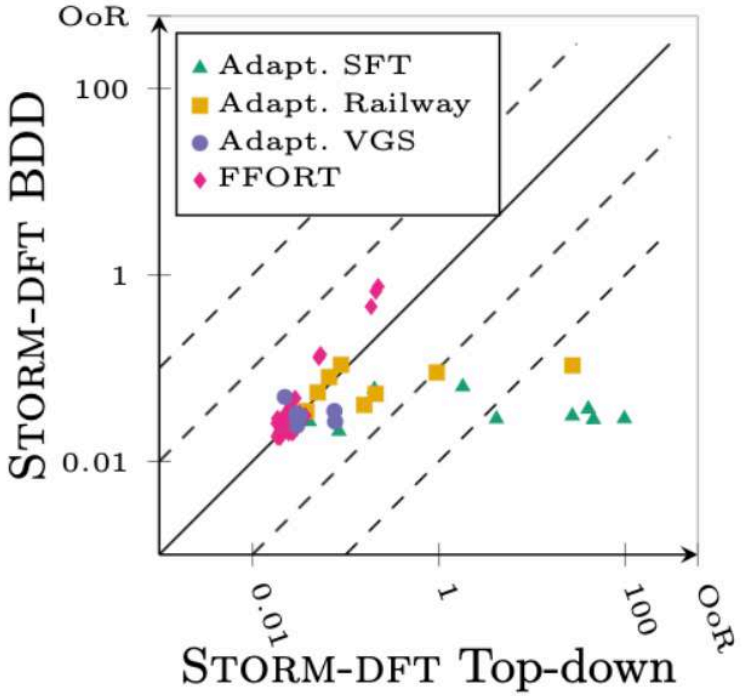
Benchmark set	#BEs	#Static gates	#Dyn. gates	#BEs mod.	#Static gates mod.
Adapt. SFT	32-1574	26-1628	3	25-1623	21-1623
Adapt. Railway	194-545	153-487	19-54	22-54	40-168
Adapt. VGS	54-99	31-59	6-20	1-79	0-39
FFORT	6-87	1-50	0-44	1-50	0-21

after modularisation

Storm

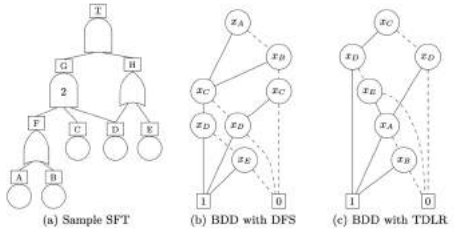
all run times in seconds



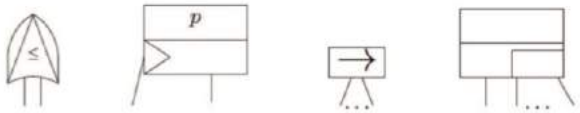


Outperforms Markov chain analysis and modularisation

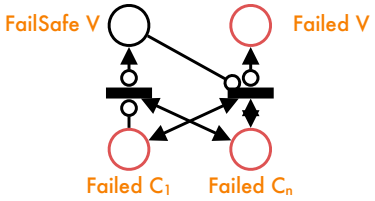
Talk Overview

1. 

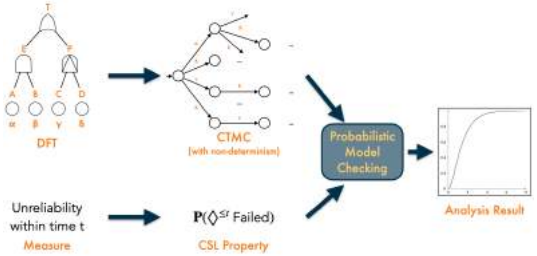
Classical Static Fault Trees

2. 

Dynamic Fault Trees

3. 

Petri Net Semantics

4. 

Scaling Up DFT Analysis

5. 

Industrial Case Studies

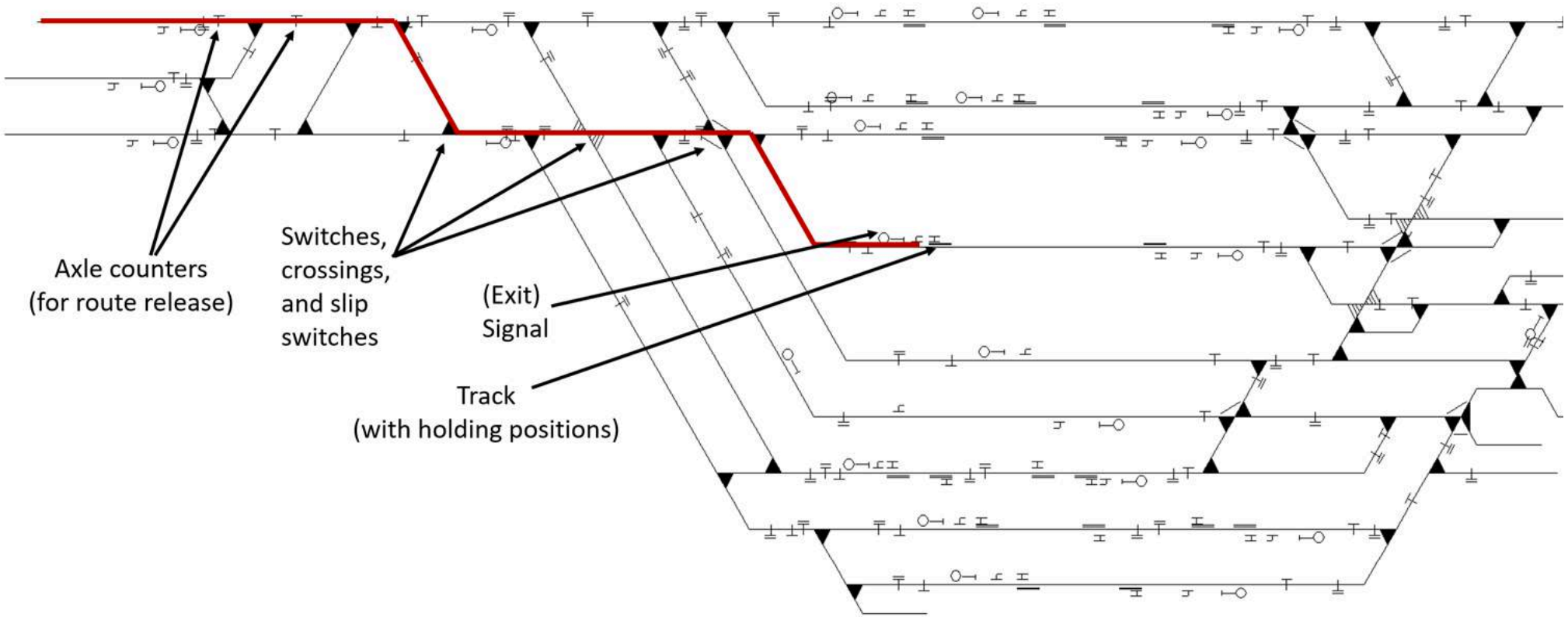
6. SAFEST
Dynamic Fault Tree Analysis

Commercialisation

Criticality Assessment of Railway Station Areas

[Weik et al., STTT 2022]

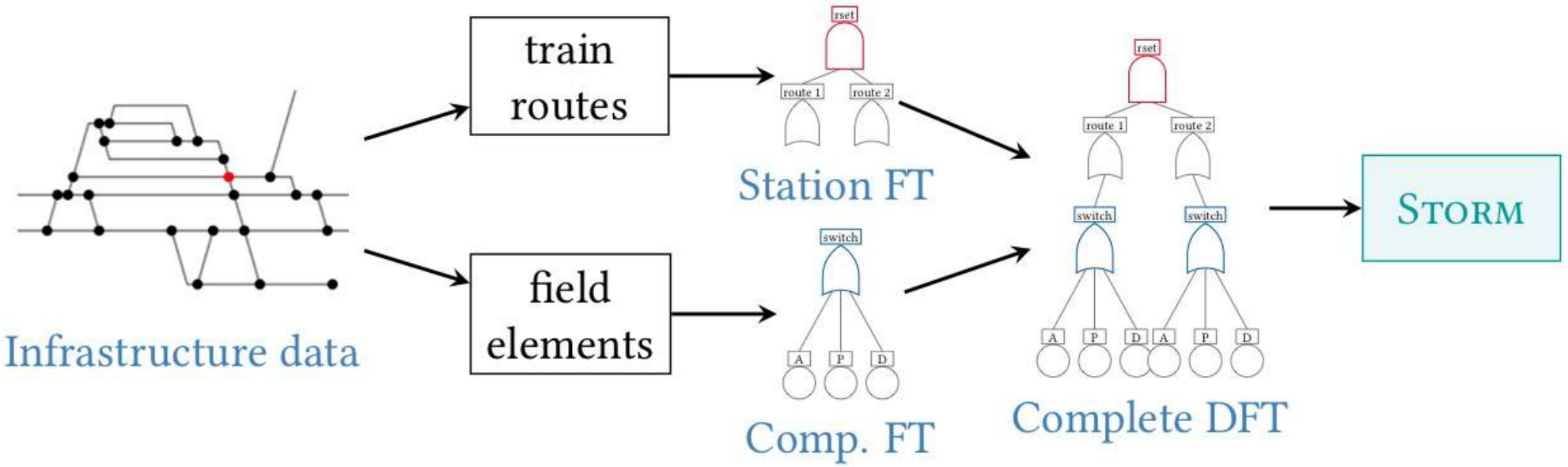
Train Routability



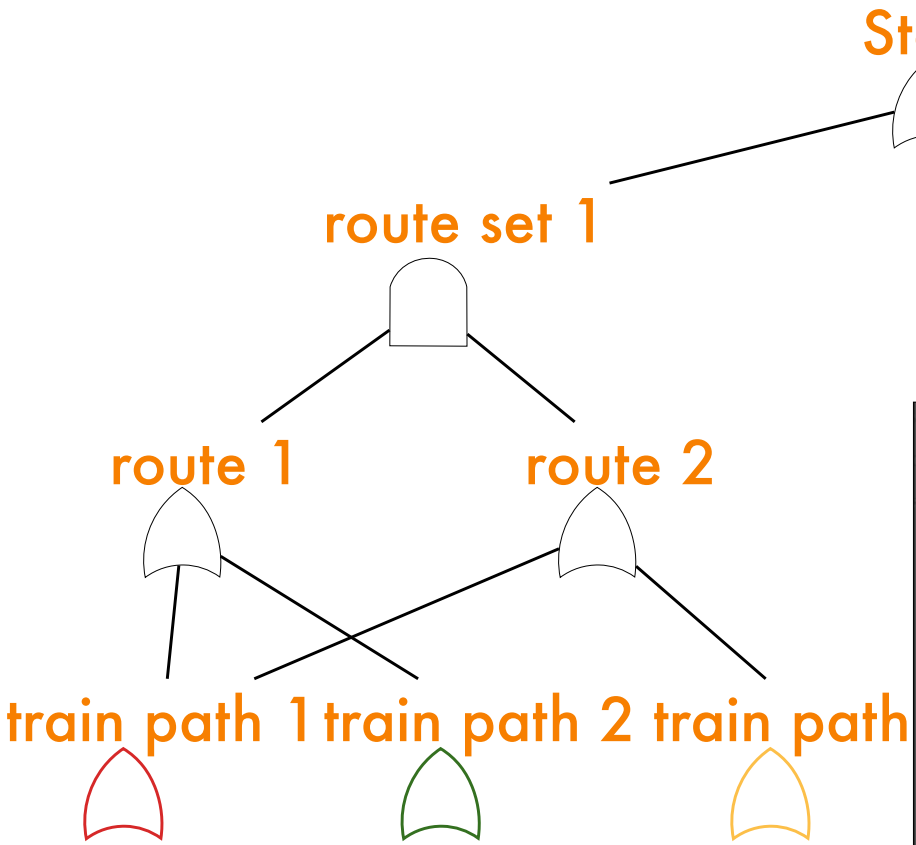
train path must be set to run train

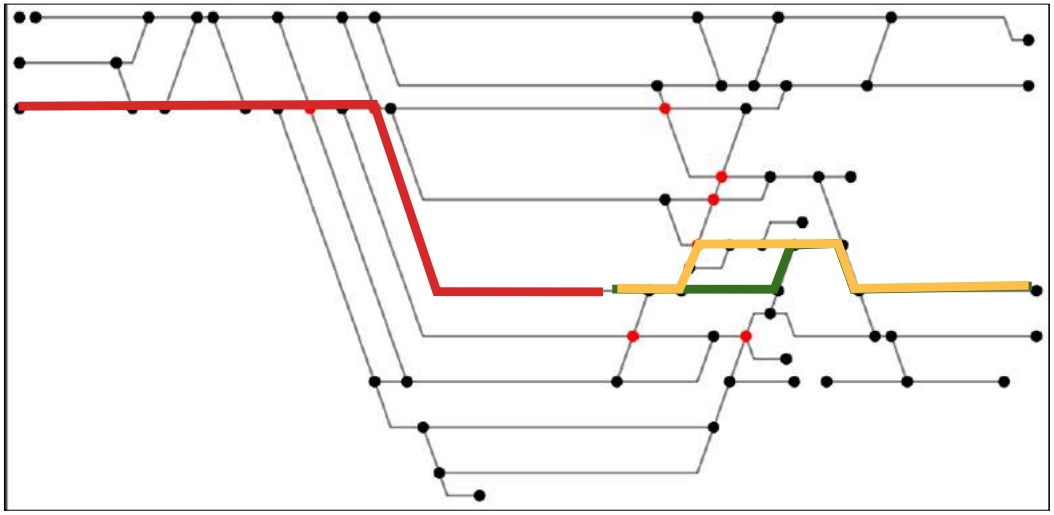
field elements must be operational and in correct position

Our Approach



Station Fault Tree





Field Elements

Other field elements

- ✓ Slip switch
  - ✓ modeled as two switches
- ✓ Crossing
- ✓ Track clearance detection
  - ✓ permanent and transient failure
- ✓ Signal

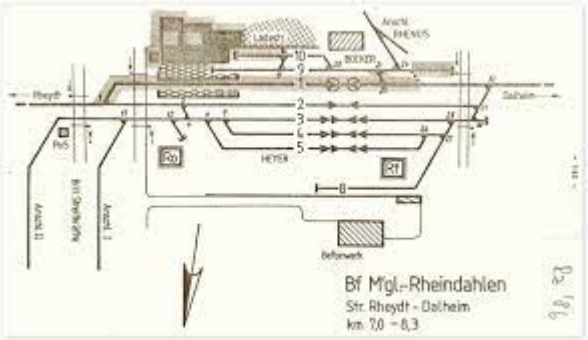
Failure rates

- ✓ Switches
  - ✓ data from UK railway network
  - ✓ failure types:
    - ✓ Actuation
    - ✓ Control/Power
    - ✓ Detection
    - ✓ Locking
    - ✓ Permanent Way
- ✓ Other field elements
  - ✓ use data from NL, N, etc.

Criticality Assessment of Railway Infrastructures

Storm

Scenario				Railway			
Id	Station	Variant	Max fail	#Route sets	#Routes	#Train paths	#Components
1	Aachen	std	∞	61	61	62	53
2		alt 5	4	23	115	41	54
3	Herzog.	std	∞	11	11	15	22
4		alt 5	4	9	19	15	24
5		alt 5	6	9	19	15	24
6	M'gladb.	std	∞	26	26	32	40
7		alt 5	4	11	43	25	41

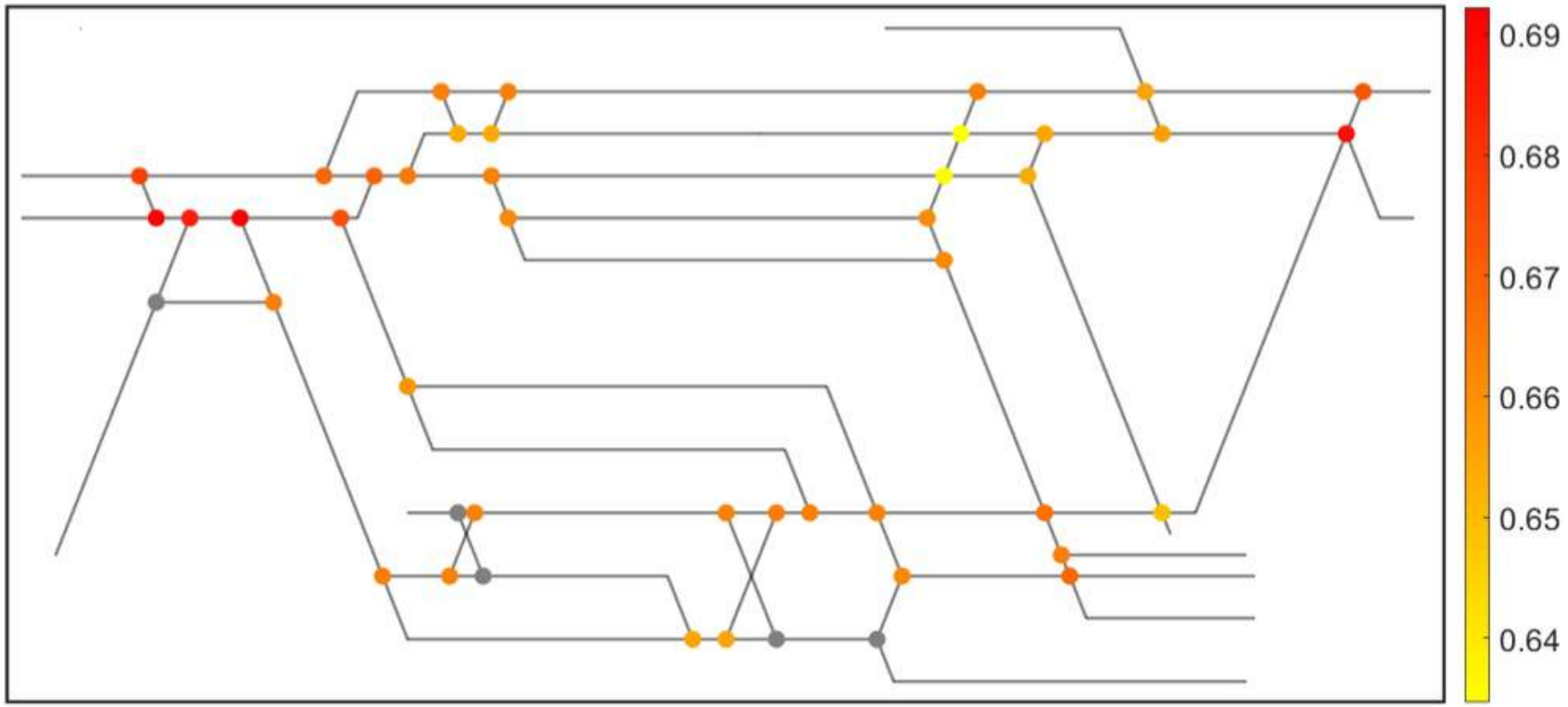


Id	DFT			CTMC		
	#BE	#Static	#Dynamic	#States	#Transitions	Build time [s]
1	544	459	54	2 049	13 313	0.11
2	536	451	53	11 371 990	45 946 651	2 006.16
3	194	137	19	257	1 281	0.04
4	214	153	21	275 073	1 109 037	12.33
5	214	153	21	17 592 280	106 375 167	1 110.48
6	480	325	48	8 193	61 441	27.79
7	490	325	49	6 224 521	24 798 158	645.51

automatically generated

automatically generated

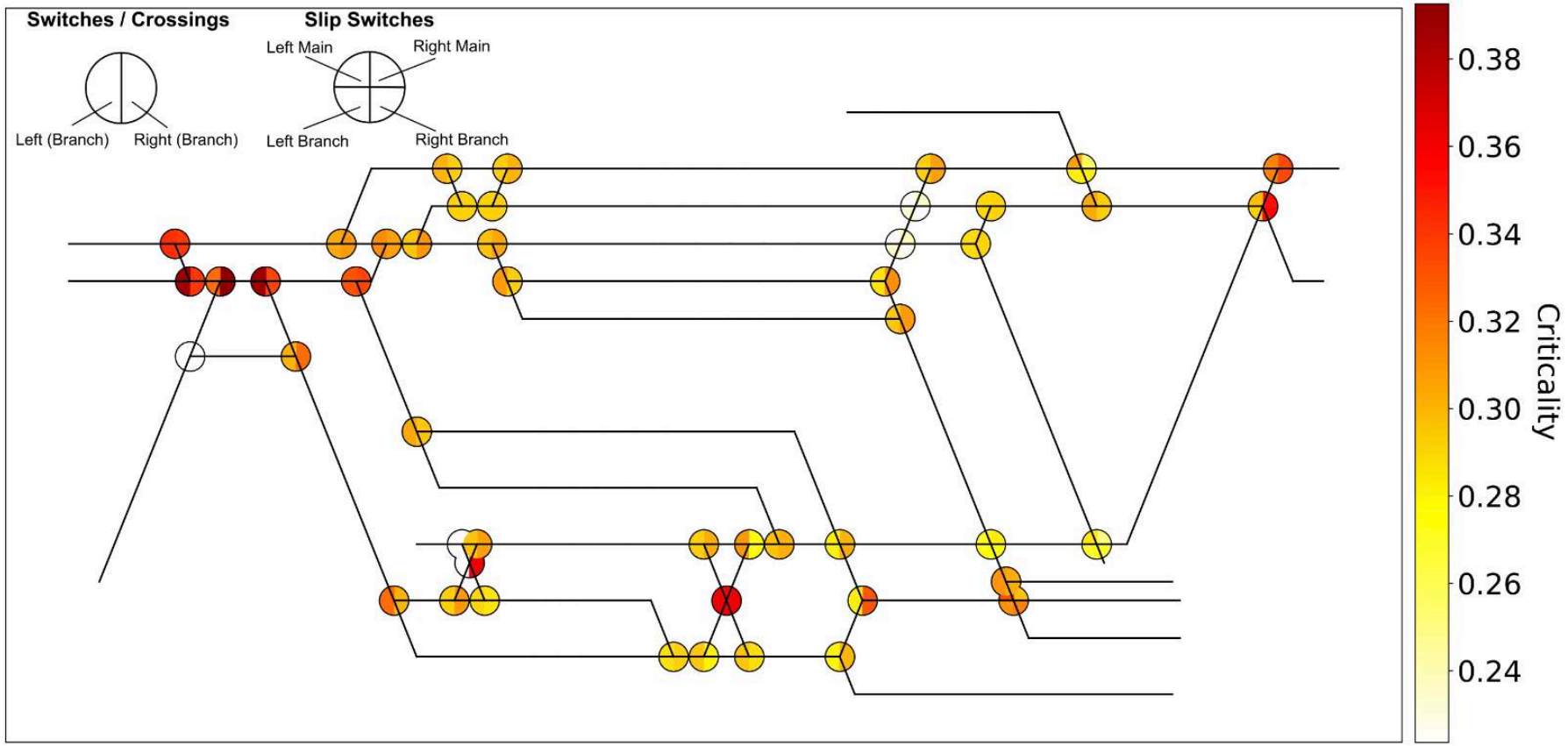
Criticality Assessment of Railway Station Areas



Criticality of Mönchengladbach Hbf

Criticality Assessment of Railway Station Areas

[Weik et al., STTT 2022]



$$I_v^t = \frac{\partial \text{Unreliability}_{\text{TLE}}^t}{\partial \text{Unreliability}_v^t}$$

Birnbaum importance index for switch branches

Mönchengladbach Hbf

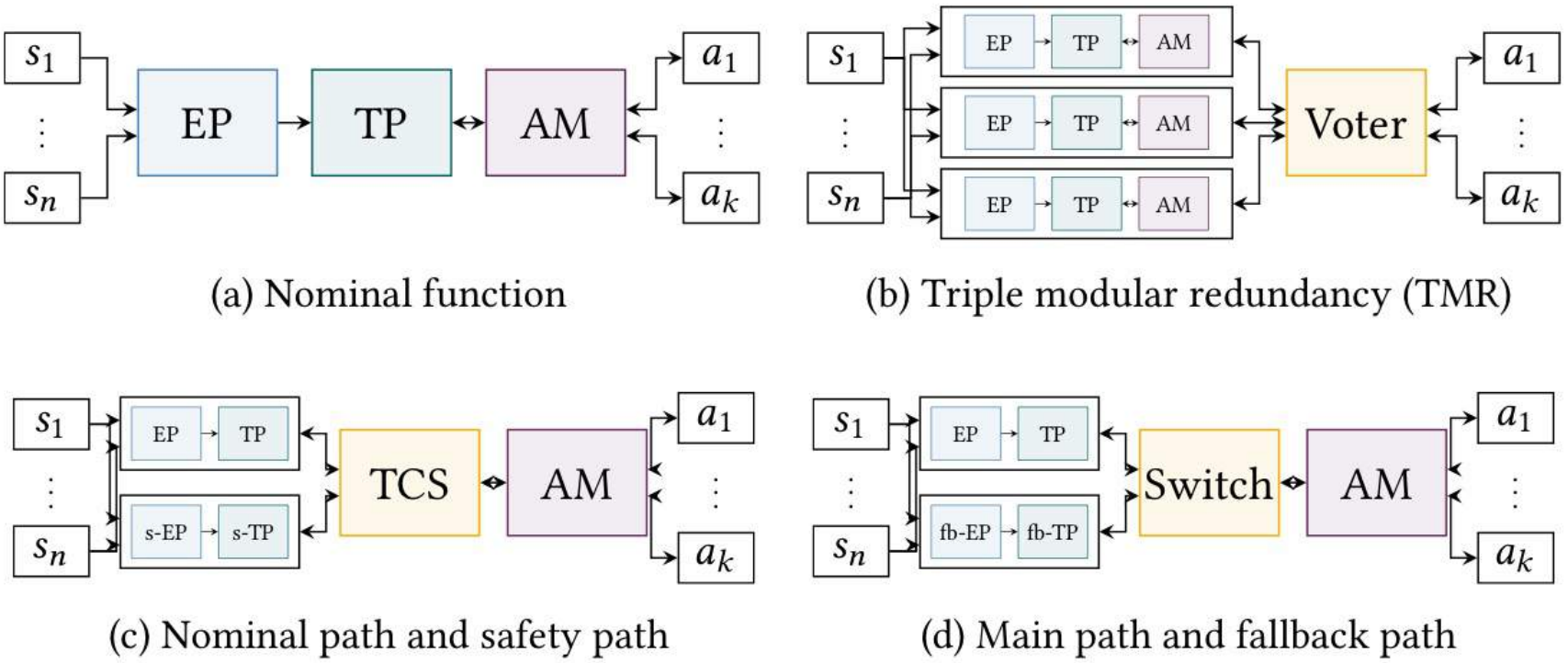
Autonomous Vehicle Guidance

[Ghadhab et al., RESS 2019]

Major safety goal: avoid wrong vehicle guidance.

Automotive Safety Integrity Level D, i.e., 10^{-8} residual hardware failures per hour

Functional Safety Blocks



(a) Nominal function

(b) Triple modular redundancy (TMR)

(c) Nominal path and safety path

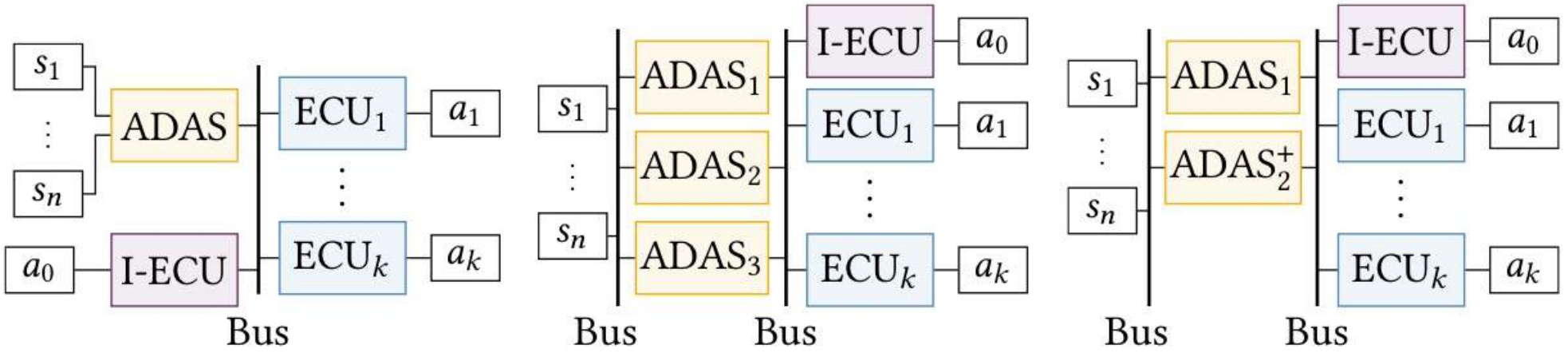
(d) Main path and fallback path

Fail-operational design patterns for autonomous driving.

EP = Environment Perception, TP = Trajectory Planning

AM = Actuator Mgt, TCS = Trajectory Checking and Selection

Sample Car Architectures



(a) E/E architecture A (b) E/E architecture B (c) E/E architecture C

(a) nominal, (b) "TMR", and (c) ADAS+ architecture.

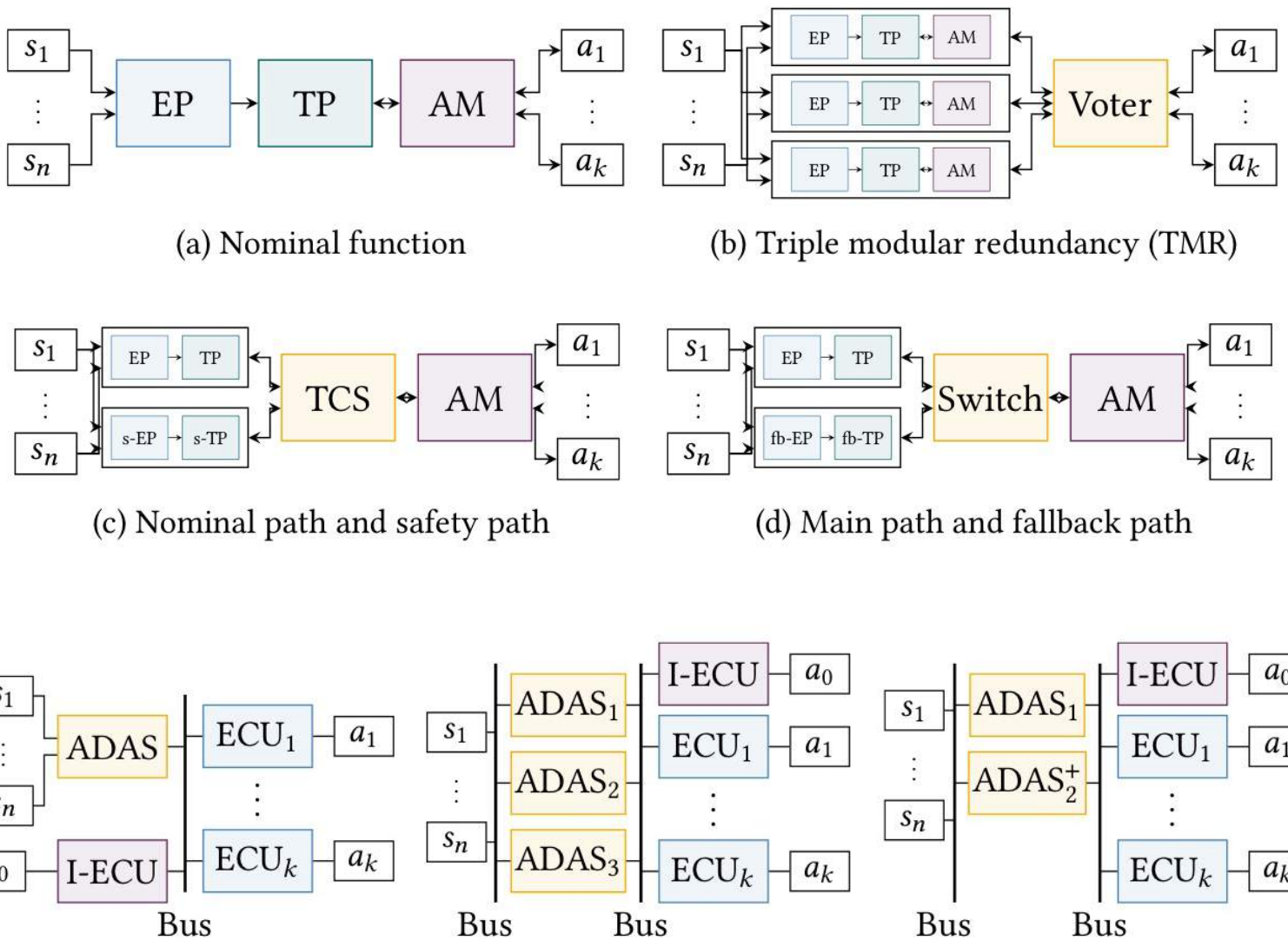
Assumption: during a transient fault, no other faults occur (conform ISO 26262)

ADAS = Advanced Driver Assistance System, I-ECU = Integration ECU

Autonomous Vehicle Guidance

Software

Hardware



Reliability Metrics Beyond Reliability and MTTF

System integrity $\approx$ probability of safe operation during operational lifetime

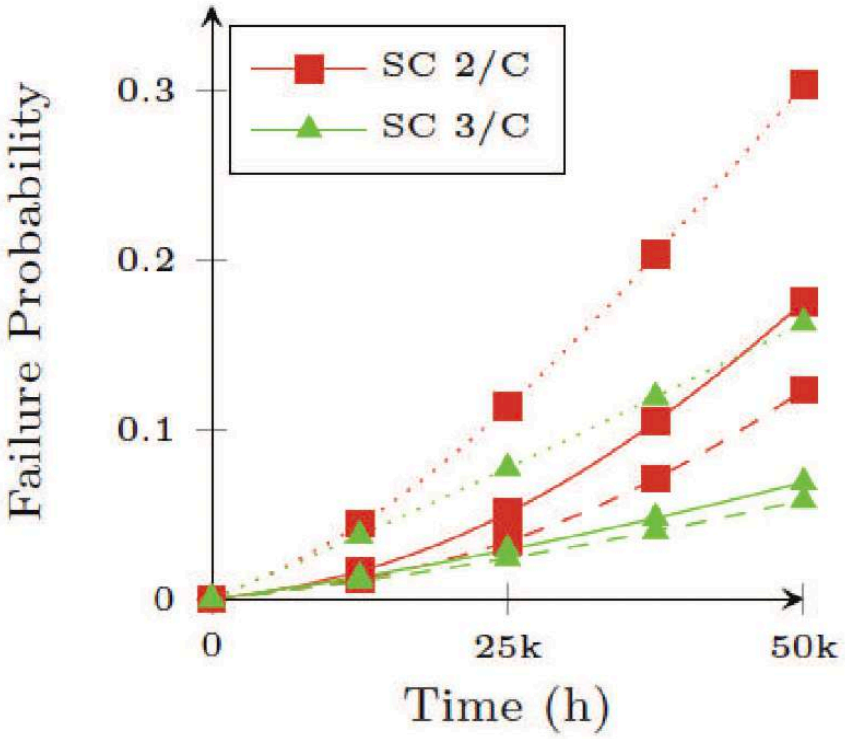
1. How probable is it that the system is fully functional at time t?
2. What is the fraction of system failures w/o being degraded first?
3. The expected time to failure upon becoming degraded?
4. Criticality: how likely is it to fail within a drive cycle once degraded?
5. System integrity when limiting operational time after degradation?

DFT Modeling Statistics

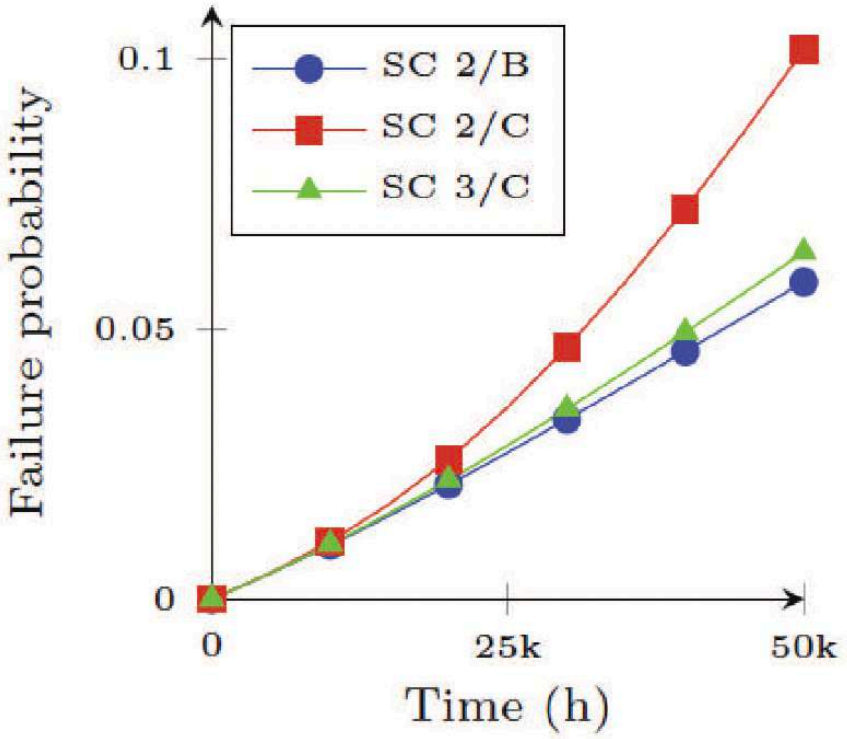
	Scenario					DFT			CTMC		
	SC	Arch.	Adap.	Sens.	Act.	#BE	#Dyn.	#Elem.	#States	#Trans.	Degrad.
I	SC1	B	—	2/4	4/4	76	25	233	5,377	42,753	—
II	SC2	B	—	2/4	4/4	70	23	211	5,953	50,049	19.35%
III	SC2	C	ADAS+	2/4	4/4	57	19	168	1,153	7,681	16.65%
IV	SC3	C	—	2/4	4/4	57	21	170	385	1,985	12.47%
V	SC2	A	—	2/4	4/4	58	19	185	193	897	0.00%
VI	SC2	B	w/o I-ECU	2/4	4/4	65	21	199	1,201	8,241	19.98%
VII	SC2	B	5 ADAS	2/8	7/7	96	30	266	$2\ 10^5$	$2\ 10^6$	19.35%
VIII	SC2	B	8 ADAS	6/8	7/7	114	36	305	$4\ 10^6$	$66\ 10^6$	10.90%

Analysis Results

[Ghadhab et al., RESS 2019]



Sensitivity



System integrity
after degradation

Nuclear Power Plant

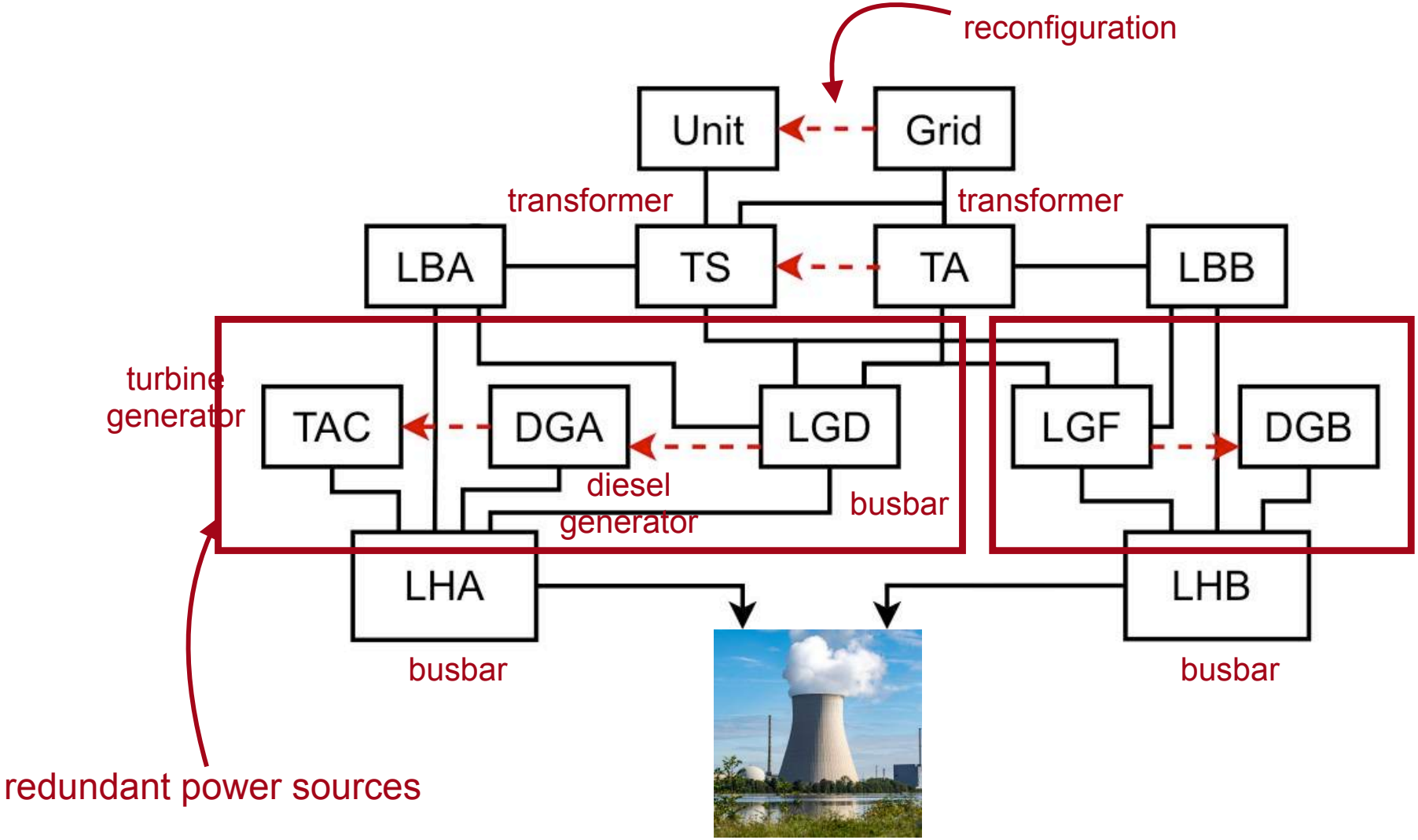
NPPS Benchmark

[Bouissou, MARS 2017]

- Nuclear Reactor managed by EDF – largest energy provider in France
- EDF challenged world reliability community to:
  - Faithfully model "Emergency Power Supply" and verify metrics like reliability, MTTF, (
- It is a highly complex and safety-critical system
  - Multiple power sources (high redundancy)
  - Large difference between failure rates of components
  - Components may fail:
    - Due to common cause failures (CCF)
    - While providing some functionality, e.g., generators fail while operating
    - When they are demanded for some service (on-demand failure)
  - Circular dependencies of components
  - Multi-directional interactions of components

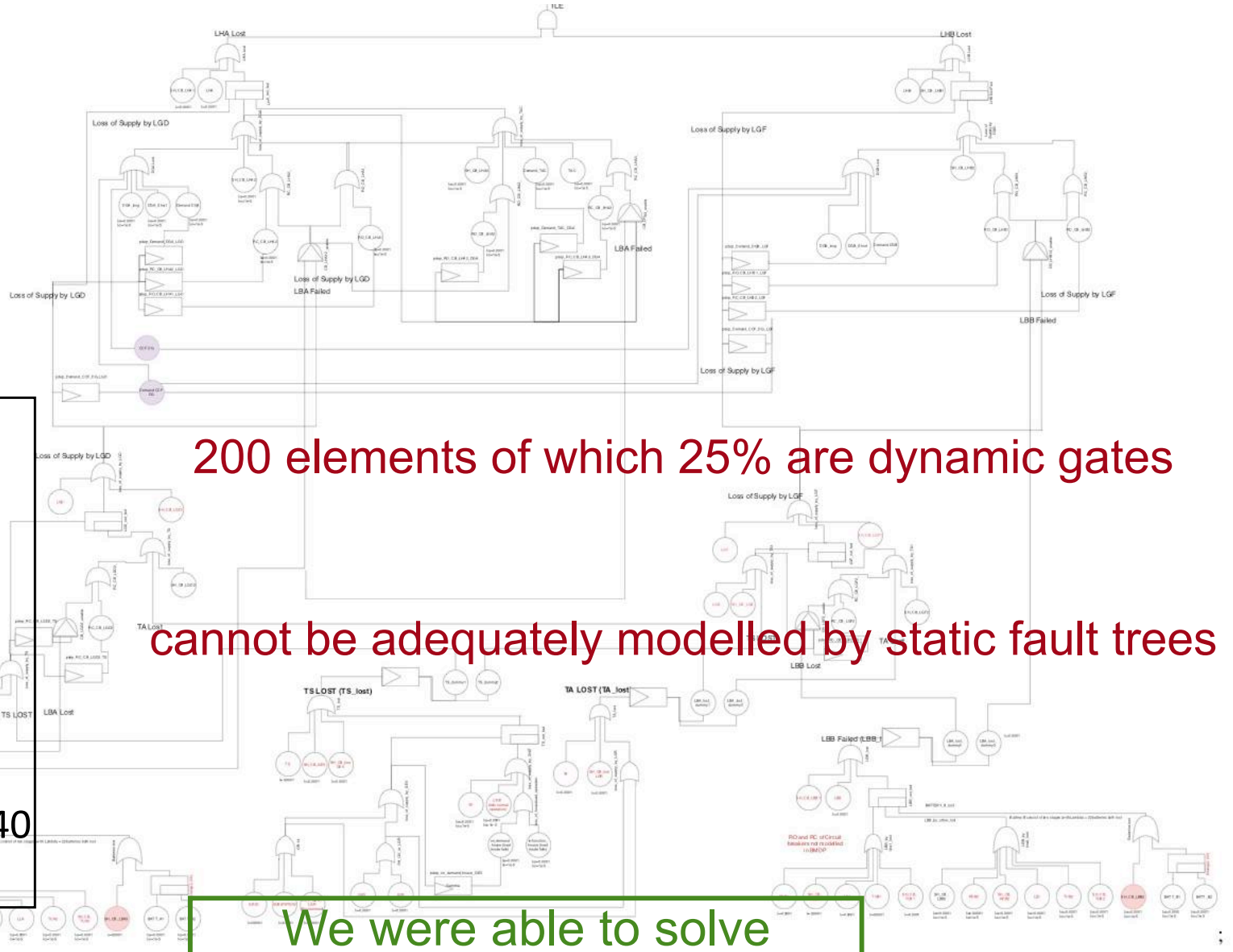
Nuclear Power Plant Power Supply

[Bouissou, MARS 2017]



DFT Model of NPPS

[Khan et al., PRDC 2019]



BEs: 107

Static gates:

- AND: 2
- OR: 36

Dynamic gates:

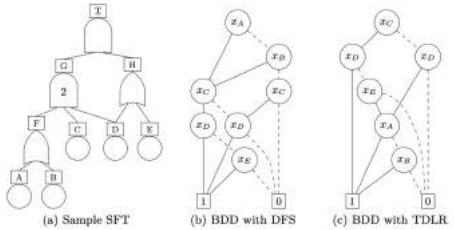
- PAND: 5
- SPARE: 8
- PDEP/FDEP: 40
- SEQ: 2

200 elements of which 25% are dynamic gates

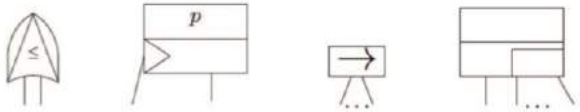
cannot be adequately modelled by static fault trees

We were able to solve this industrial challenge in 14s

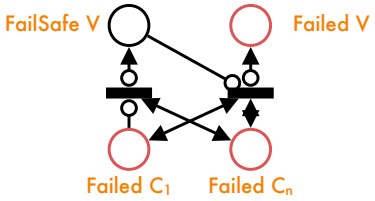
Talk Overview

1. 

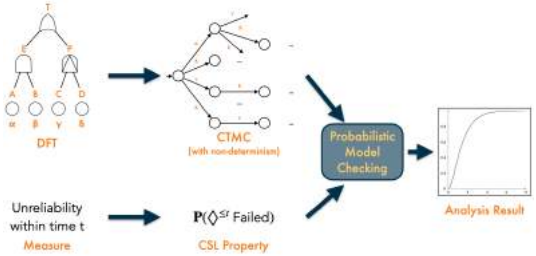
Classical Static Fault Trees

2. 

Dynamic Fault Trees

3. 

Petri Net Semantics

4. 

Scaling Up DFT Analysis

5. 

Industrial Case Studies

6. SAFEST
Dynamic Fault Tree Analysis

Commercialisation

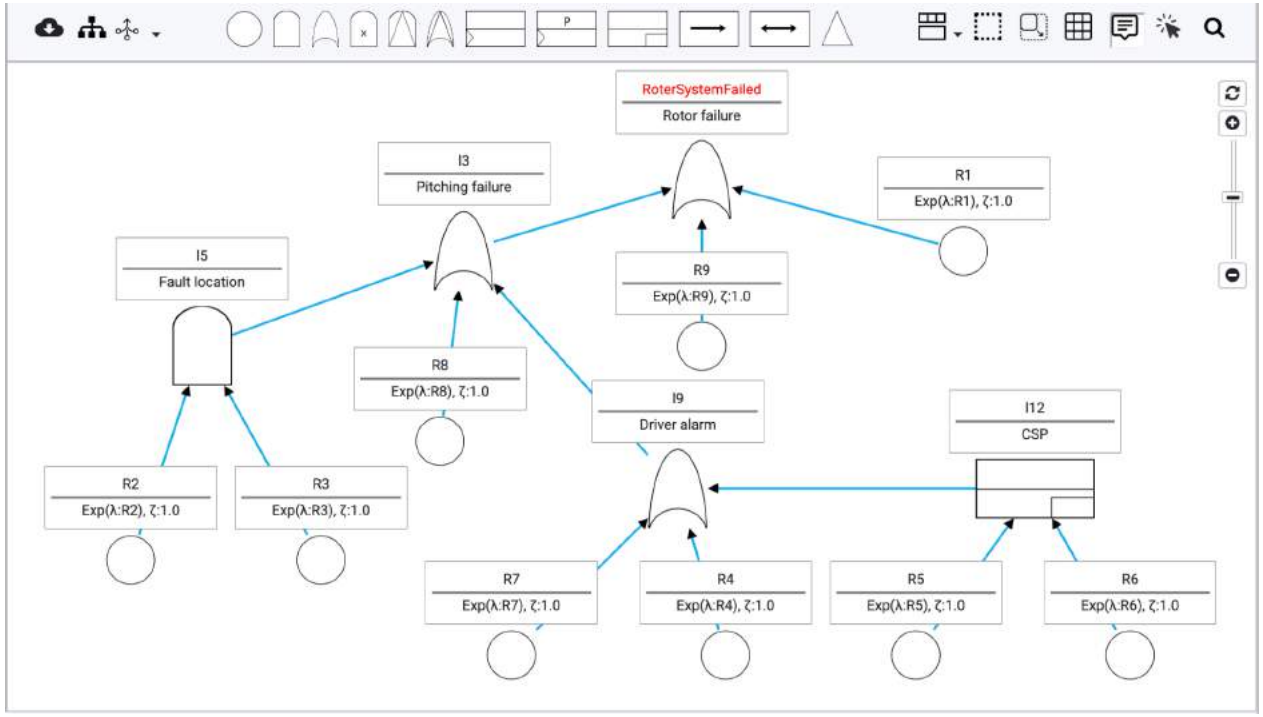
SAFEST: Static And dynamic Fault trEe analySis Tool

Modelling

Analysis

https://www.safest.dgbtek.com

Modelling with SAFEST

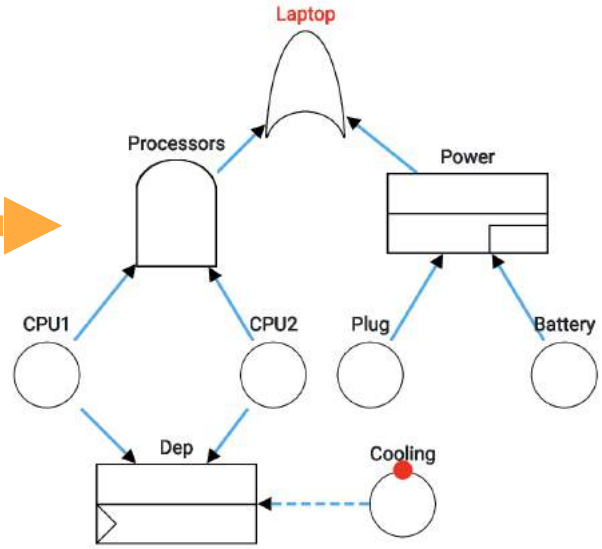


Modelling

01 Graphical fault tree

Modelling with SAFEST


```
package LaptopPackage {
    part Laptop{
        part CPU1{↔}
        part CPU2{↔}
        part cooling{↔}
        part plug{↔}
        part battery{↔}
    }
    metadata power:REDUNDANCY about
    Laptop::plug,noSwitch, Laptop::battery;
    metadata processors:AND about
    Laptop::CPU1, Laptop::CPU2;
    metadata laptop:OR about
    power, processors;
    metadata Dep:FDEP about
    Laptop::CPU1, Laptop::CPU2{
        trigger_occurrence = Laptop::cooling;}
    metadata TLE:TOP_LEVEL about laptop;
}
```

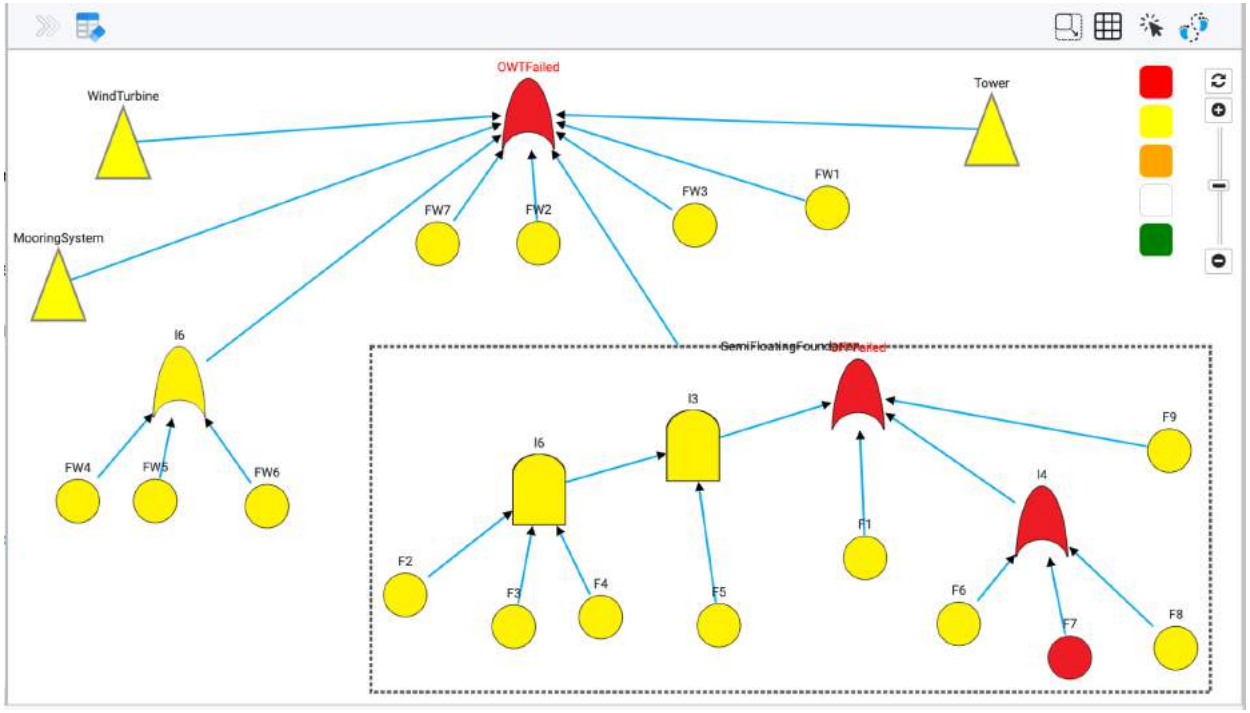


Modelling

02 SysML 2.0 to DFT

Modelling with SAFEST

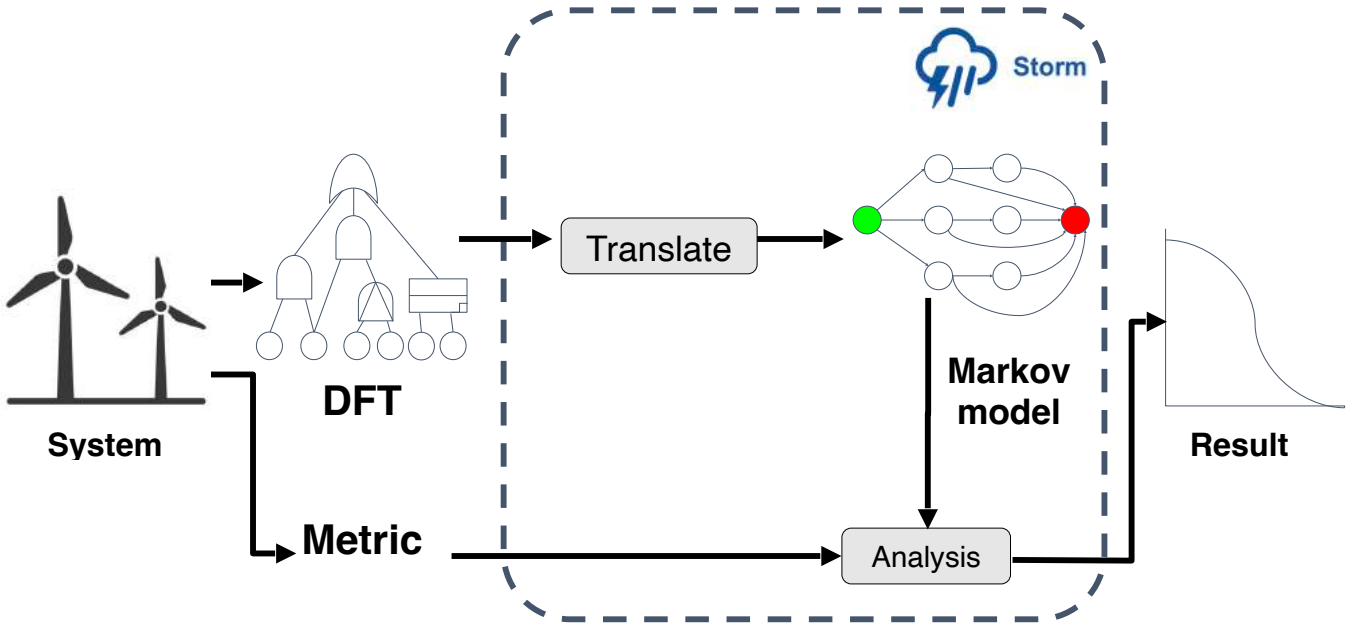
SAFEST
Dynamic Fault Tree Analysis



Modelling

03 Interactive Simulation

Analysis with SAFEST

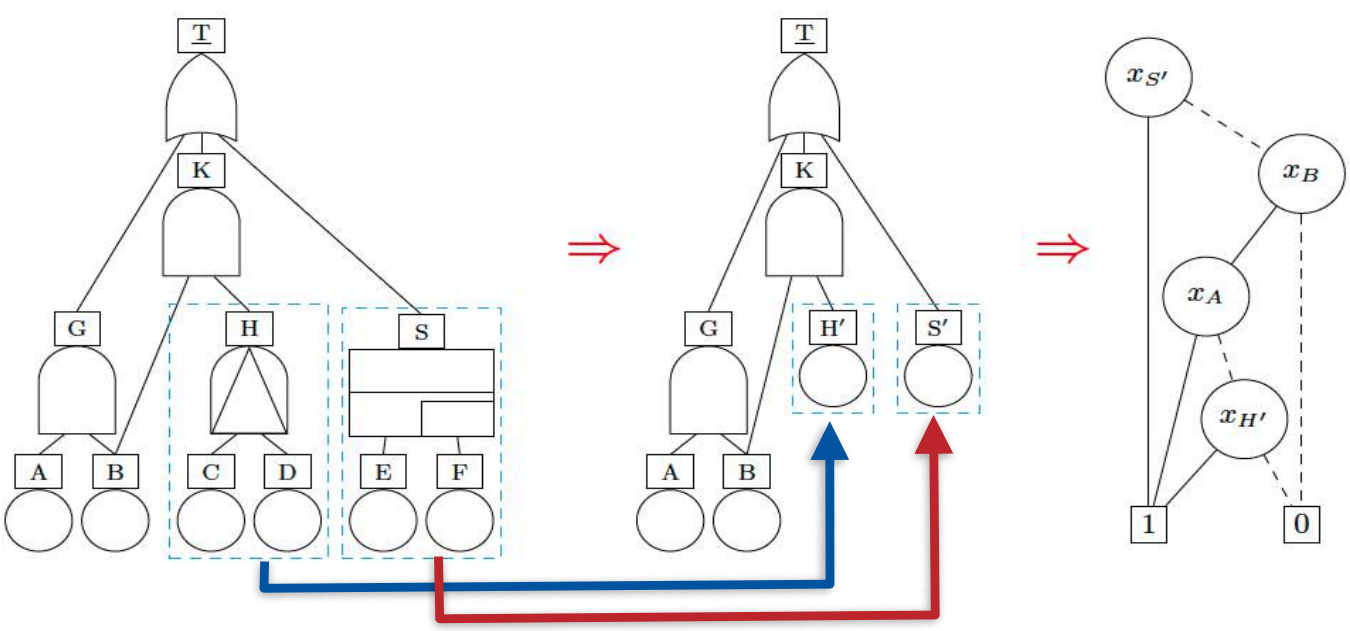


SAFEST
Dynamic Fault Tree Analysis

04 Probabilistic model checking

Analysis

Analysis with SAFEST

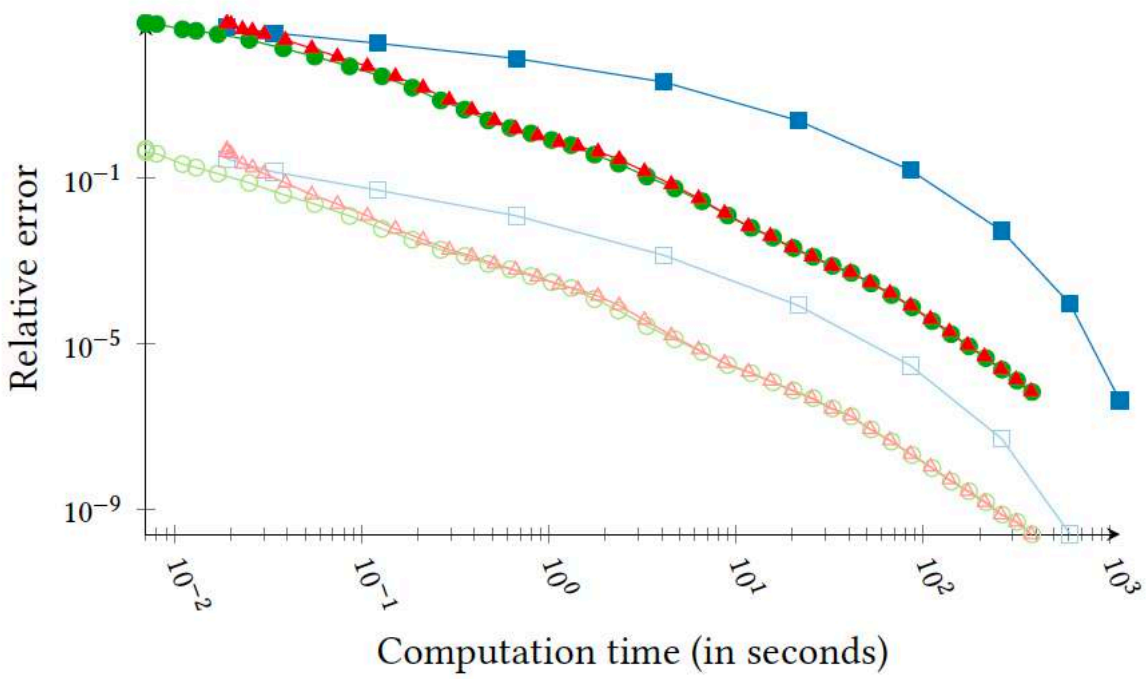


SAFEST
Dynamic Fault Tree Analysis

05 Modularisation

Analysis

Analysis with SAFEST



SAFEST
Dynamic Fault Tree Analysis

06 Bounded approximation

Analysis

Take-Home Messages

What?

- Analysis of the largest dynamic fault trees ever
- Metrics beyond standard reliability measures
- Full automation: Storm-DFT --> SAFEST
- Validated by various industrial case studies

How?

- DFT rewriting +
- Slim state-space generation +
- Tailored Markov chain model checking

Try it out

We applied this principle also
to BDMPs, an EDF fault tree dialect

Literature

- Semantic Intricacies of DFTs [Junges et al, DSN 2016]
- Simplifying DFTs by Graph Rewriting [Junges et al, Form. Asp. Comp., 2017]
- Fast DFT Analysis by Model Checking [Volk et al, IEEE Trans. Ind. Inf, 2018]
- One Net Fits All: Unifying Semantics of DFTs [Volk et al, Petri Nets 2018]
- Analysing DFTs with Static Parts [Basgöze et al, NFM 2022]
- Railway Station Areas Application [Weik et al, STTT 2022/FMICS 2019]
- Autonomous Car Application [Ghadhab et al, Reliab. Eng. Syst. Saf. 2019]
- Reliability Analysis of EDF's Fault Trees [Khan et al, DSN 2021/IEEE TDSC 2023]
- SAFEST: Static and Dynamic Fault Tree Aanalysis Tool [Volk et al., ESREL 2023]

Big Thanks to my Co-Workers!

Matthias Volk

Sebastian Junges

Marielle Stoelinga

Nils Nießen

Mattias Kuntz

Falak Sher

Arend Rensink

Enno Ruijters

Norman Weik

Shahid Kahn

Daniel Basgöze

Dennis Guck

Madji Ghadhab

Yasmeen Elderhalli

Marc Bouissou