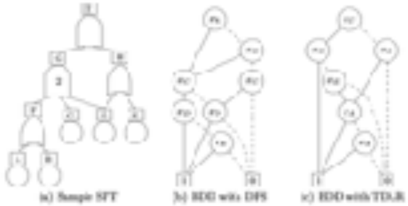
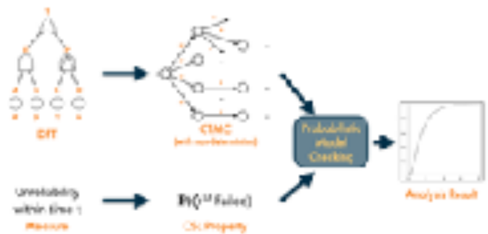
Scalable Fault Tree Analysis by Model Checking

Joost-Pieter Katoen and Falak Sher

August 23, 2022

Talk Overview

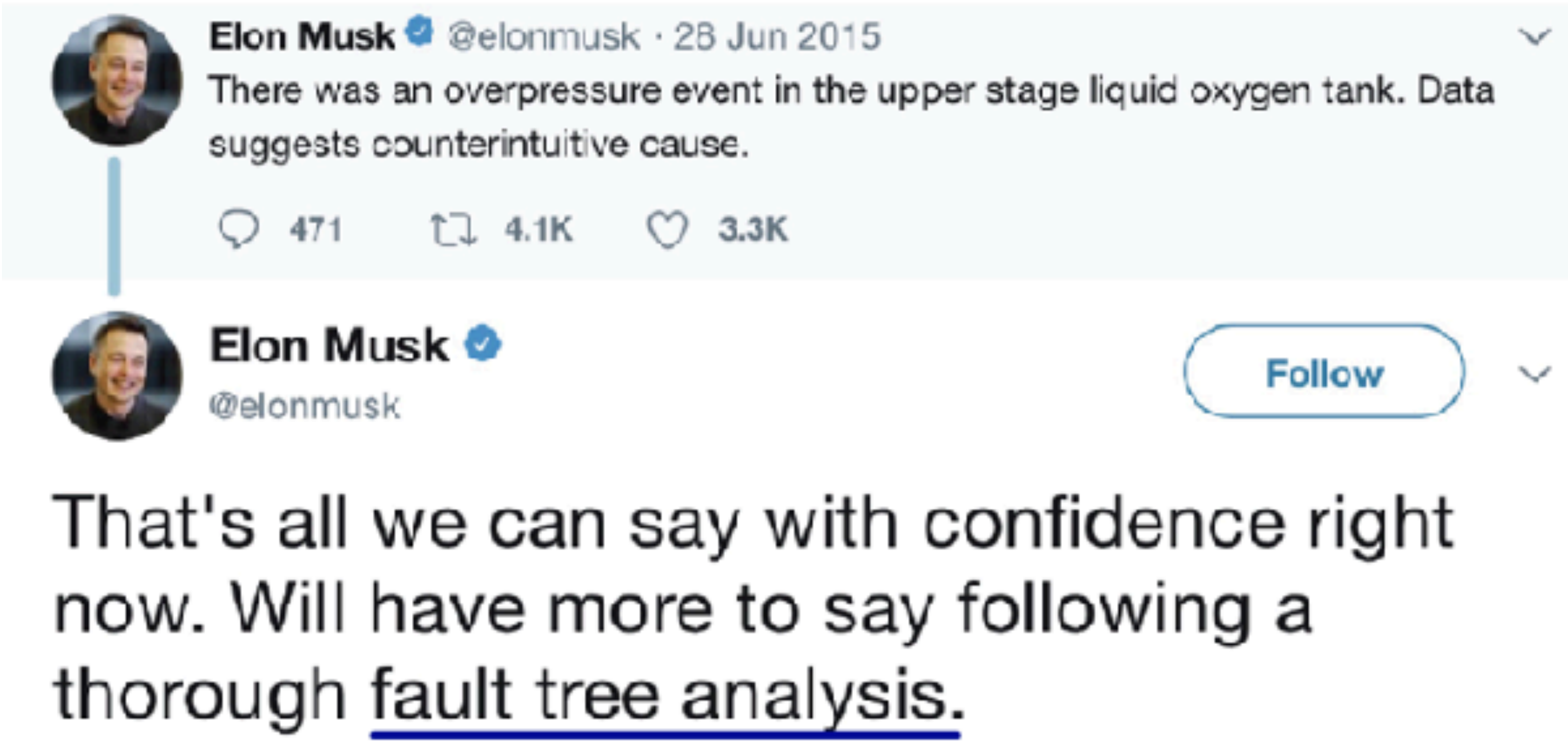
1.  Classical Static Fault Tree Analysis
2.  Dynamic Fault Trees
3.  Scaling Up DFT Analysis
4.    Industrial Case Studies
5. Storm — Storm Tool Demonstration

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


Elon Musk @elonmusk · 28 Jun 2015

There was an overpressure event in the upper stage liquid oxygen tank. Data suggests counterintuitive cause.

471 4.1K 3.3K

Elon Musk @elonmusk Follow

That's all we can say with confidence right now. Will have more to say following a thorough fault tree analysis.

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

Static Fault Trees

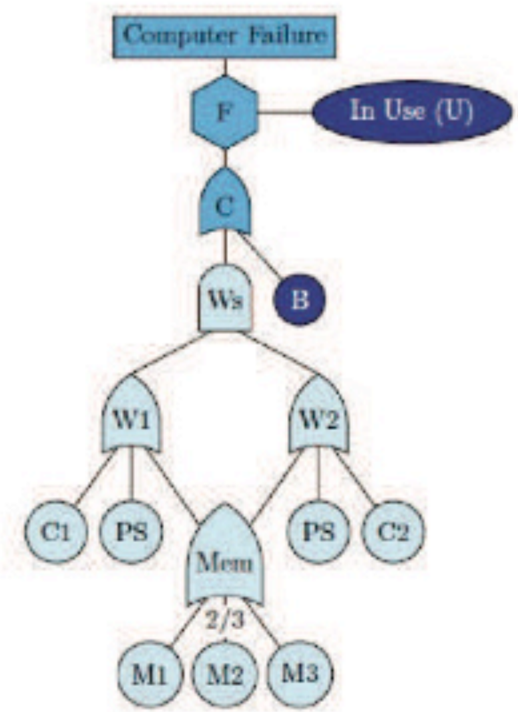
[Watson, 1961-62]

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

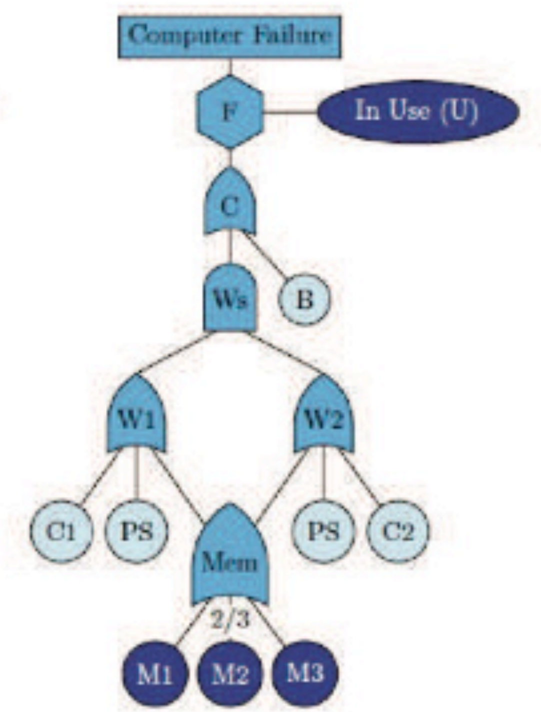
(a) AND gate (b) OR gate (c) k/N gate (d) INHIBIT gate

- 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

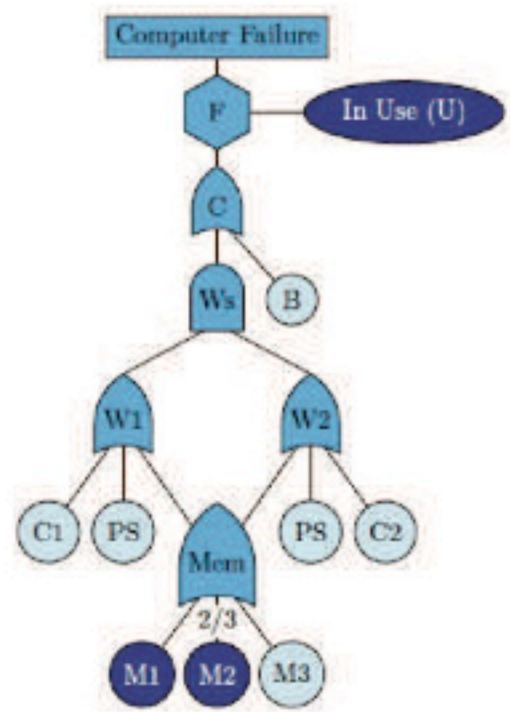
Minimal Cut Sets



minimal



not minimal

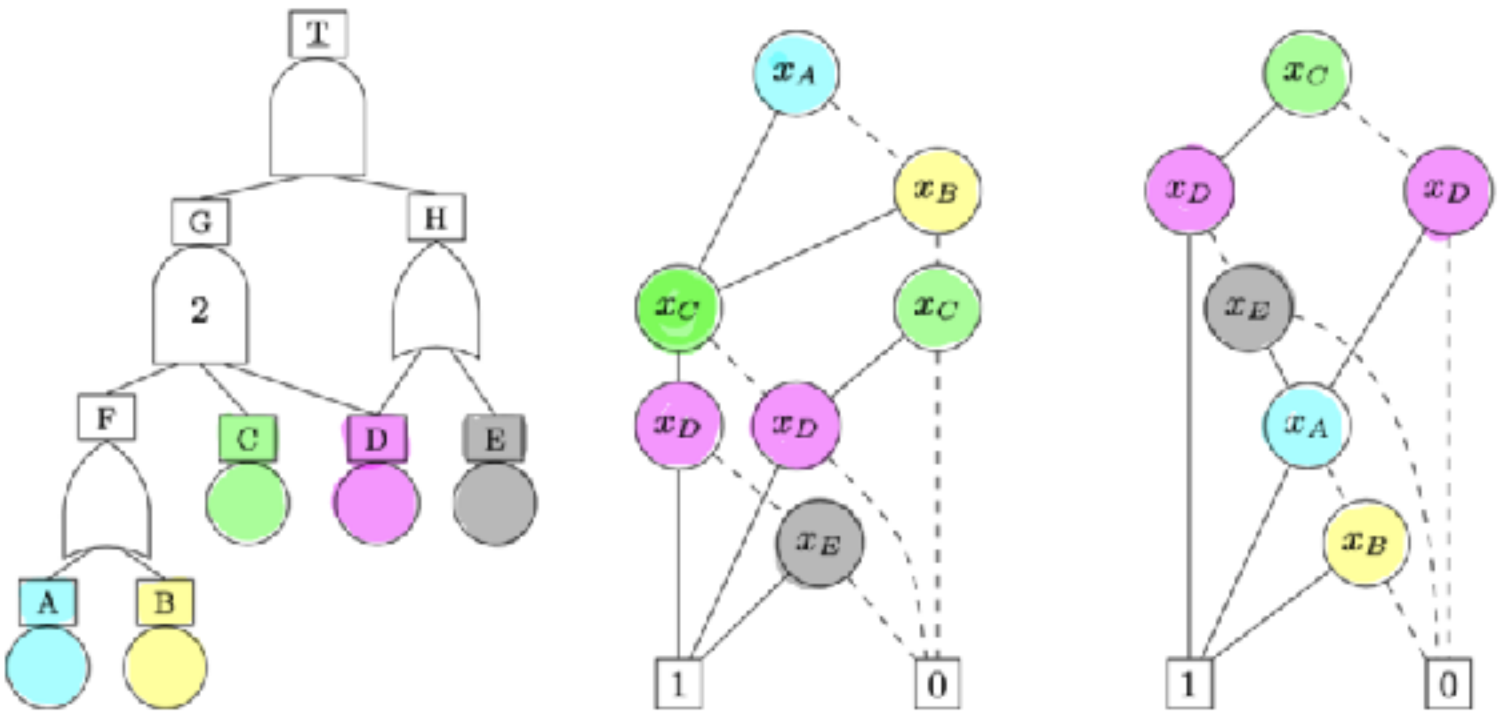


minimal

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

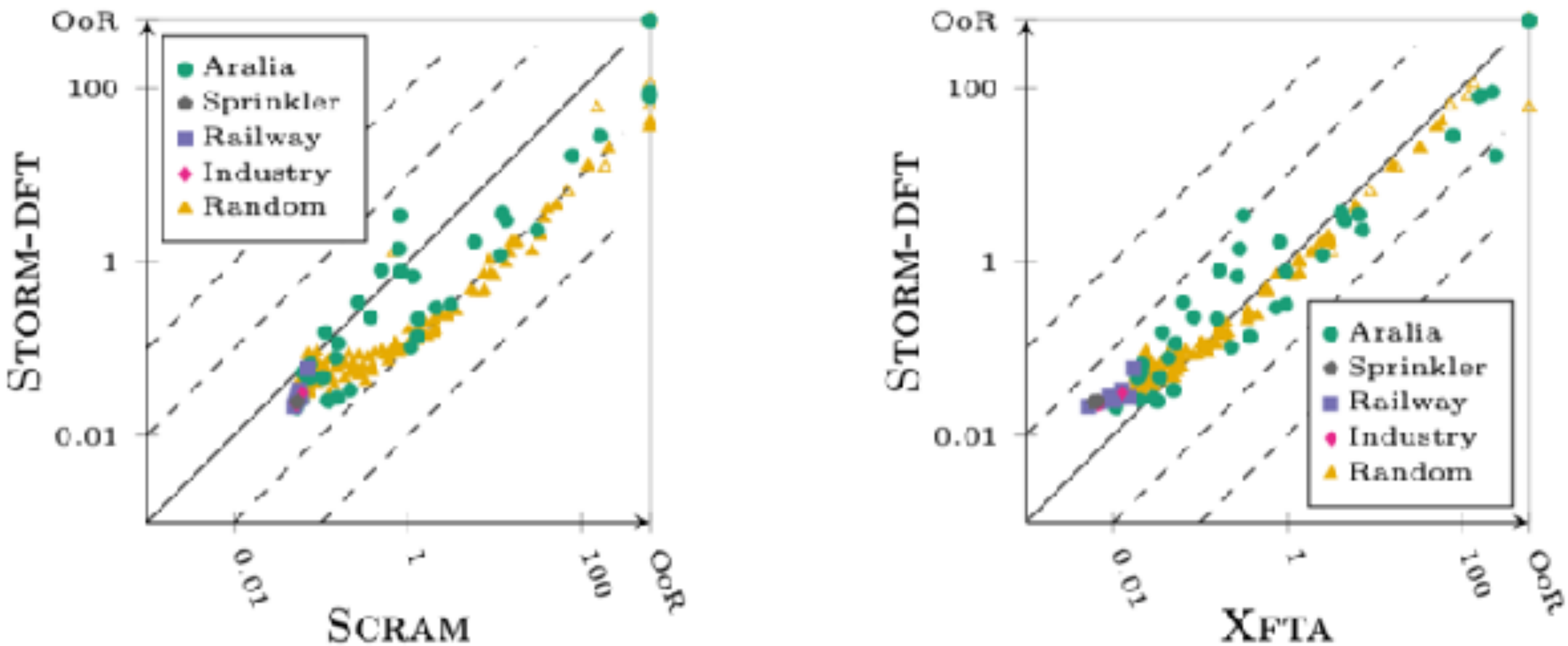
Experimental: Computing MCS

[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

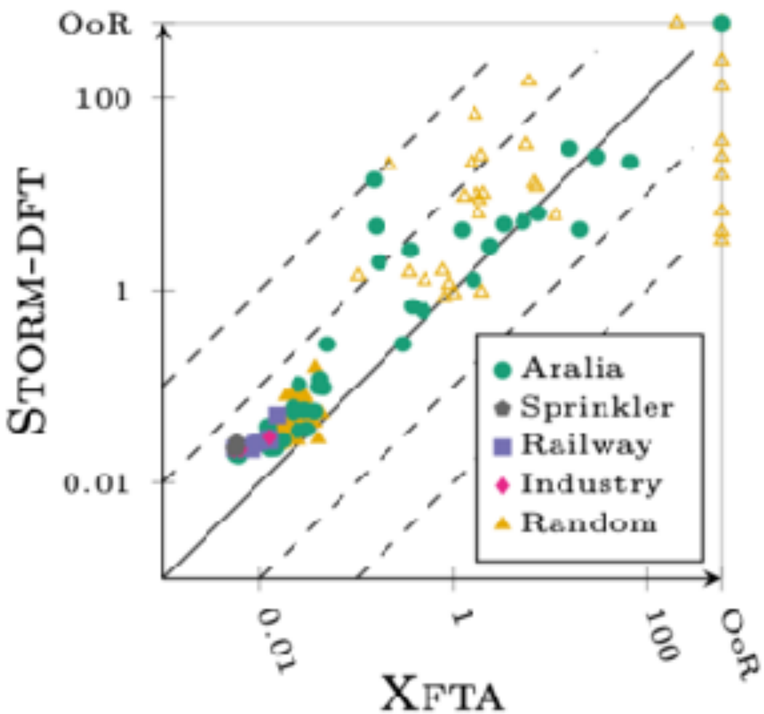


For computing MCS, Storm-DFT is faster than both XFTA and SCRAM for large SFTs

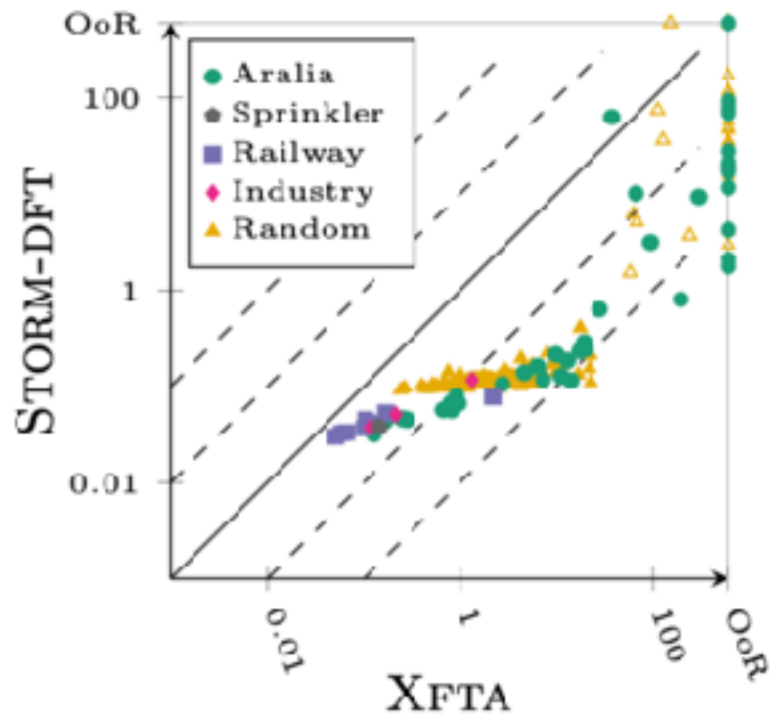
Experiments: Computing Birnbaum Index

[Basgöze et al., NASA FM 2022]

Single time point



Multiple (1,000) time points



Storm-DFT is slower than XFTA for one time point, but significantly faster for multiple time points

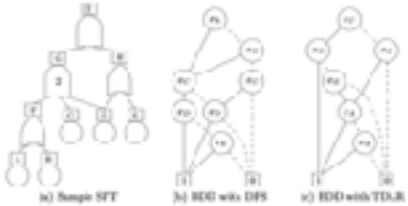
SFT Deficiencies

- 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

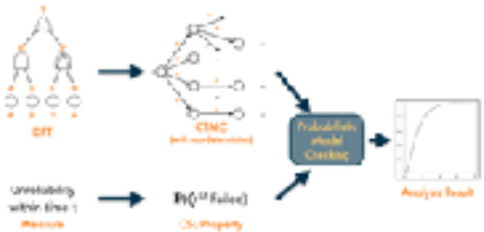
Many variants:

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 Tree Analysis

2.  Dynamic Fault Trees

3.  Scaling Up DFT Analysis

4.    Industrial Case Studies

5. Storm — Storm Tool Demonstration

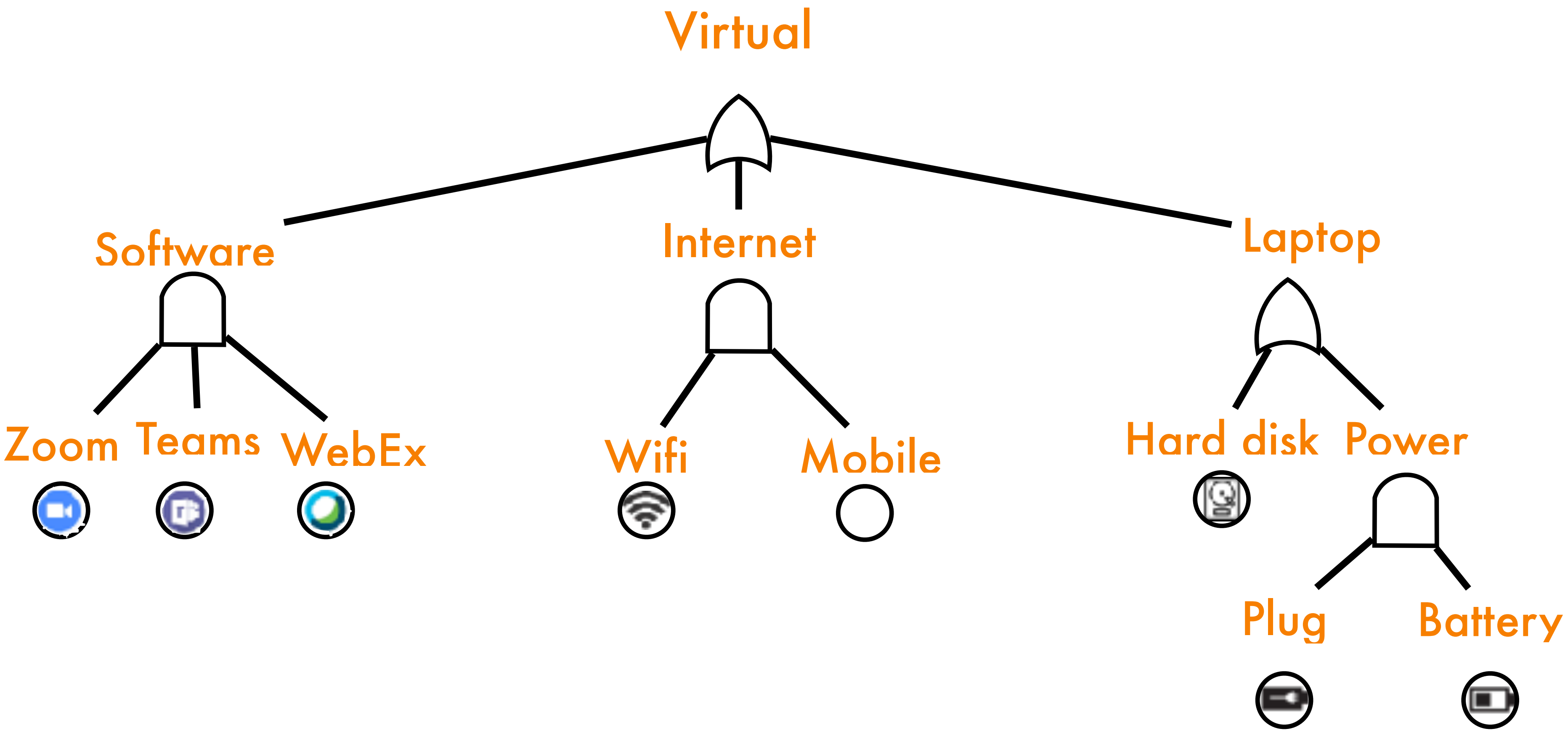
Dugan's Dynamic Fault Trees

"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."

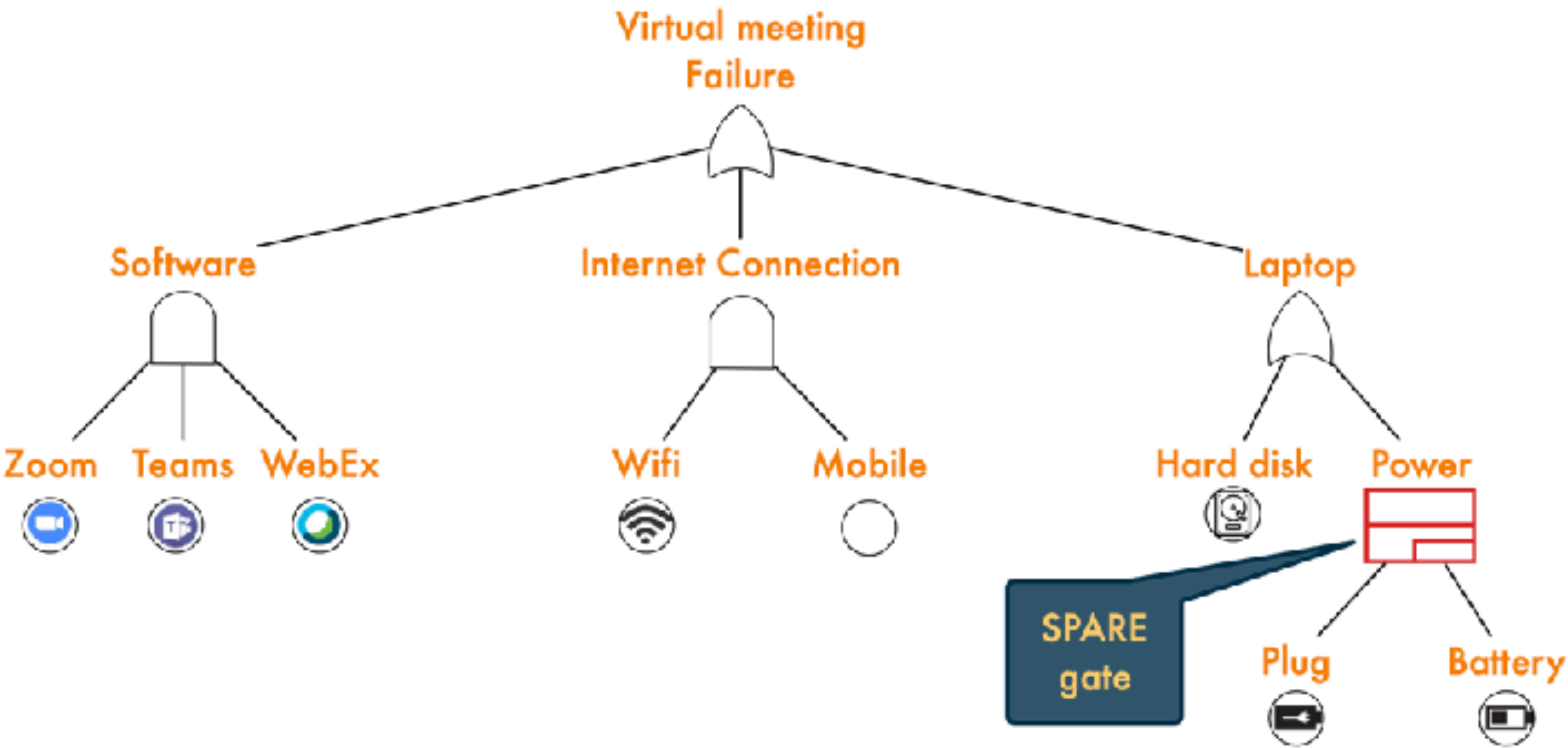
- JOANNE BECHTA DUGAN, PROFESSOR OF ELECTRICAL & COMPUTER ENGINEERING

Galileo User's Manual & Design Overview

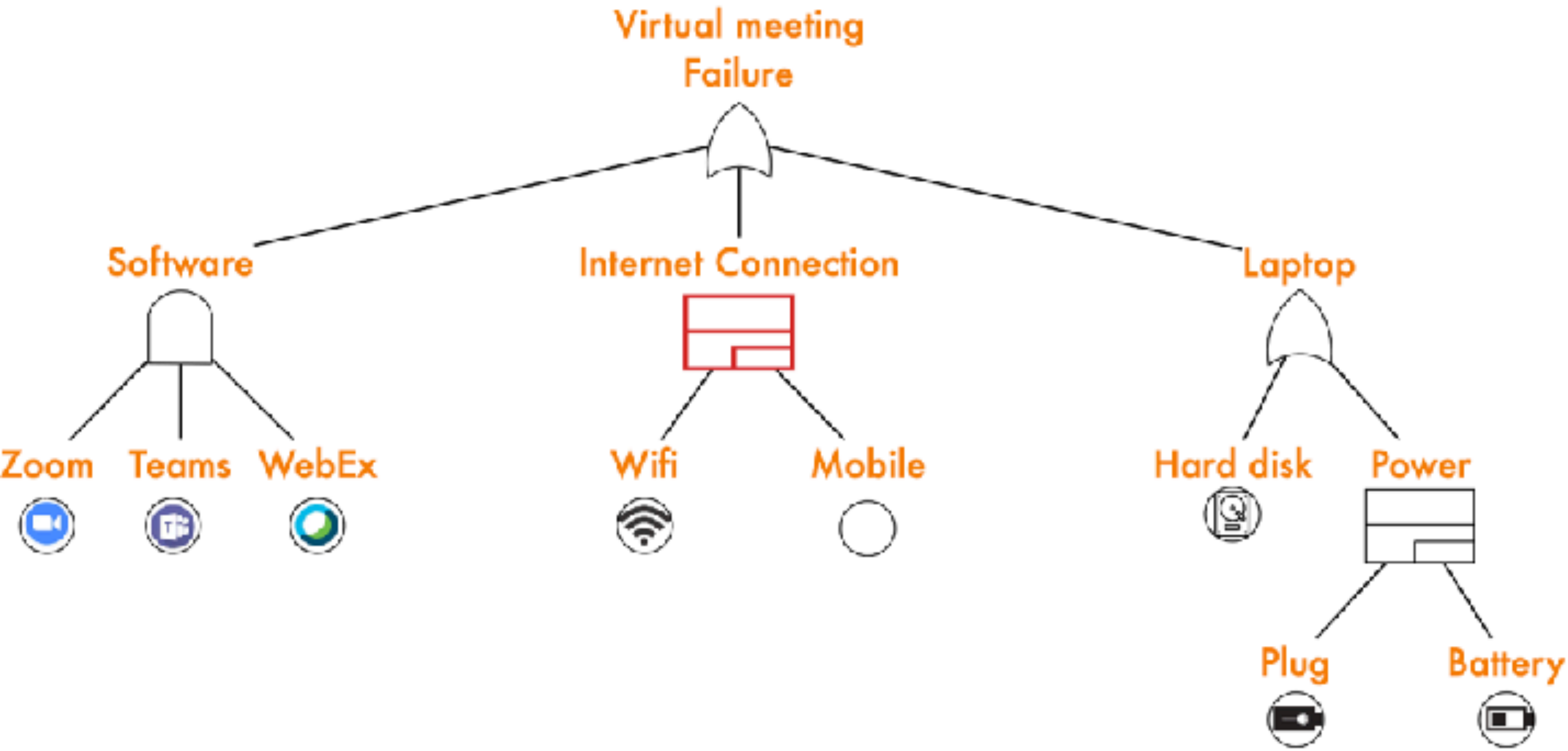
A Sample Dynamic Fault Tree



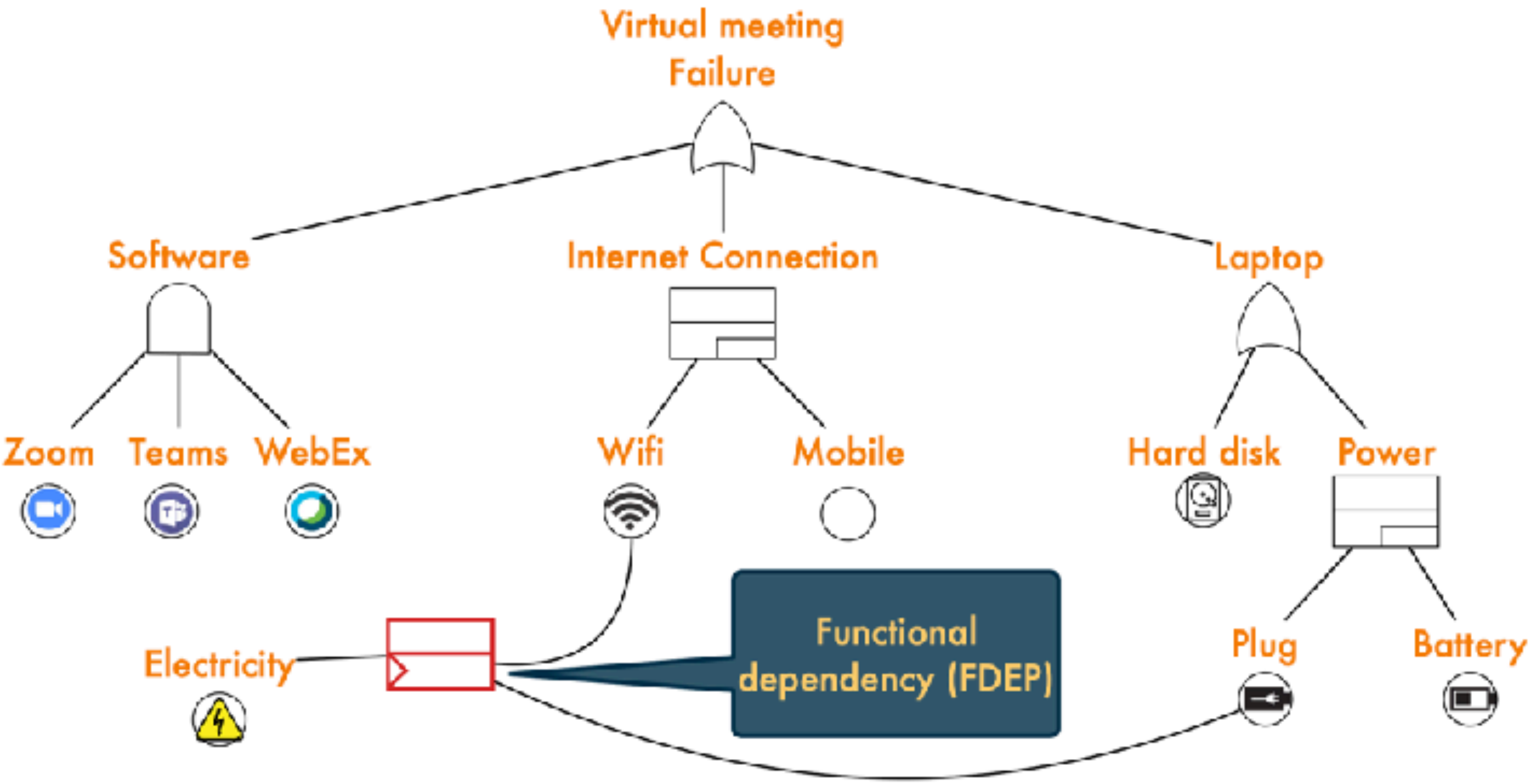
A Sample Dynamic Fault Tree



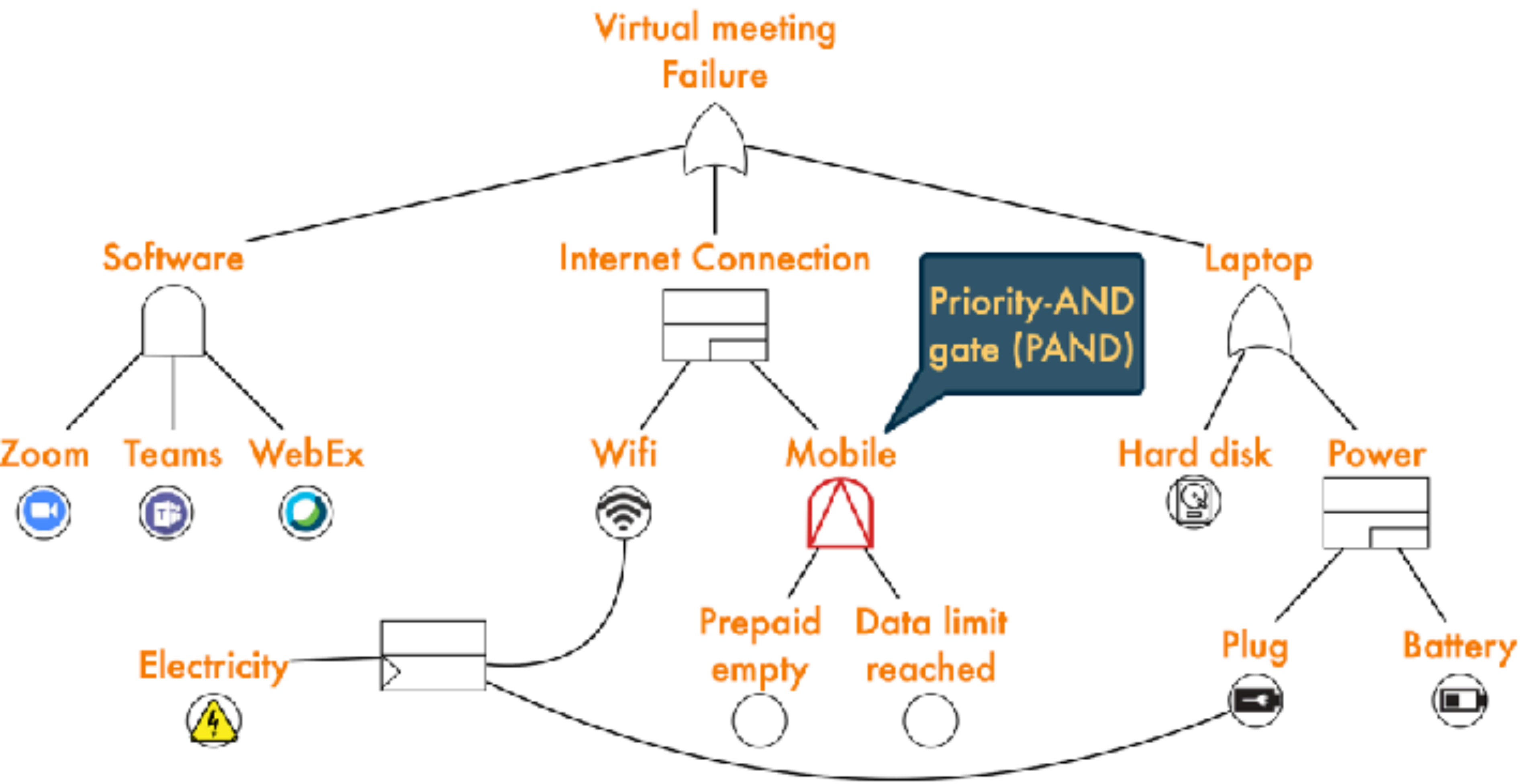
A Sample Dynamic Fault Tree



A Sample Dynamic Fault Tree



A Sample Dynamic Fault Tree



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."

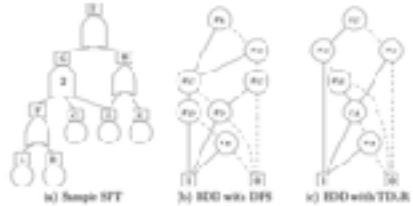
[Ge *et al.*, Rel. Eng. Syst. Safe, 2015]

"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."

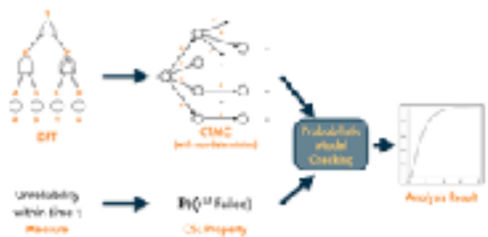
[Kabir, Expert Syst. Appl., 2017]

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

Talk Overview

1. 
Classical Static Fault Tree Analysis

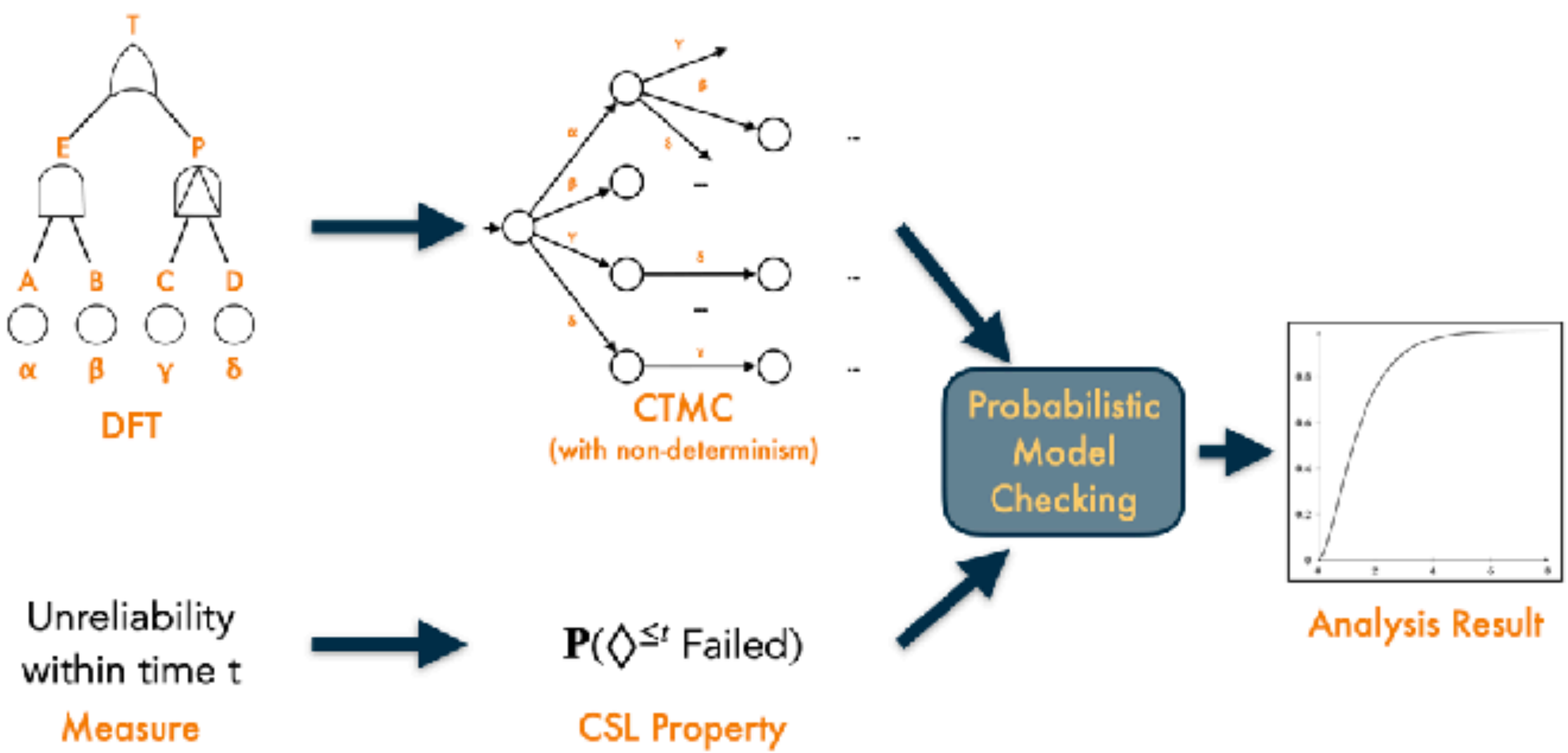
2. 
Dynamic Fault Trees

3. 
Scaling Up DFT Analysis

4.   
Industrial Case Studies

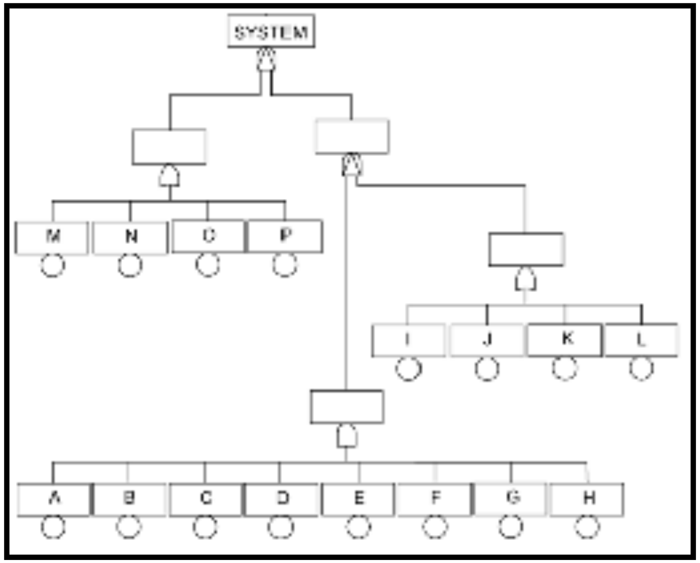
5. Storm
Storm Tool Demonstration

Analysis Workflow



https://www.stormchecker.org

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

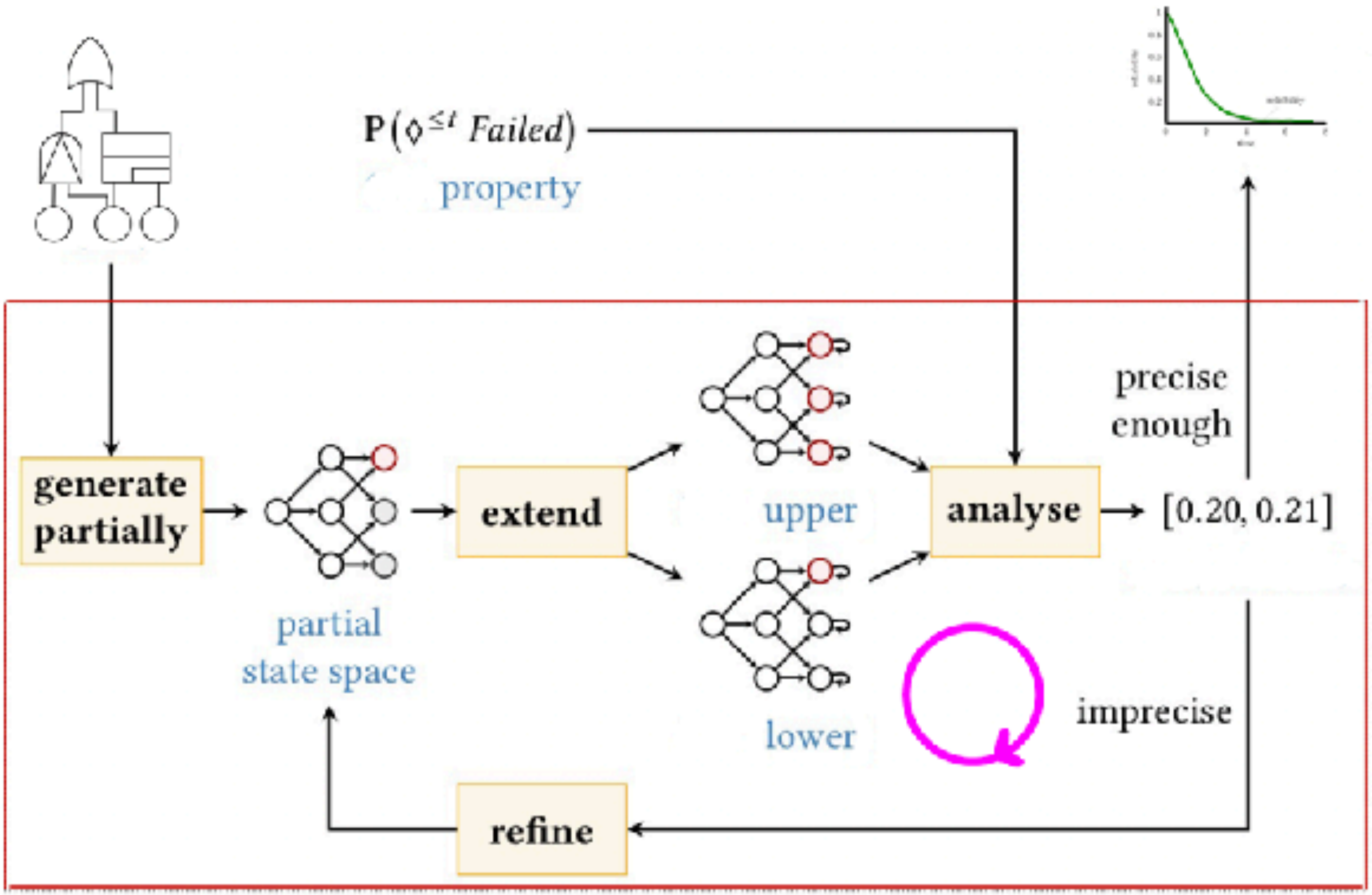
Exact result

Optimisations

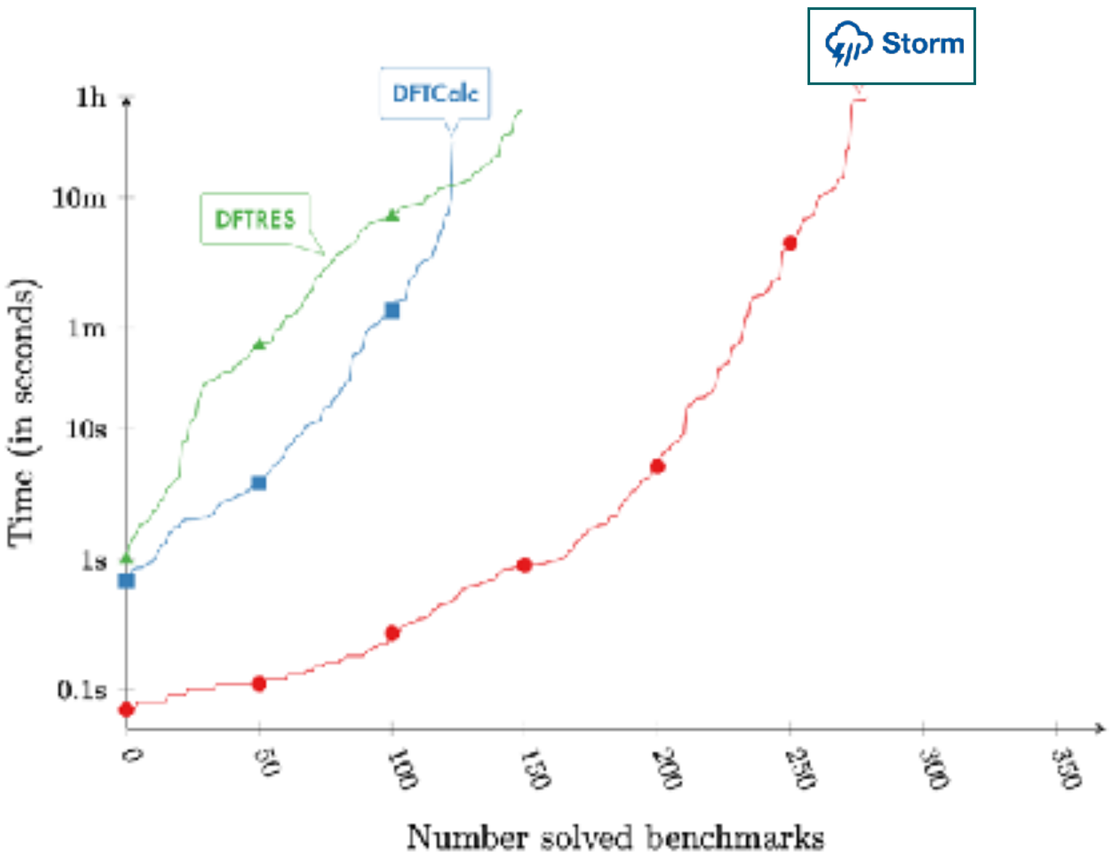
All these techniques were revised, improved and combined.

- **Don't Care** [Bouissou, Bon, 2003] for BDMP, [Yevkin, 2016]
  - exact status of element is irrelevant for further analysis
  - Example: 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, adapted also to MTTF
- Eliminate **spurious non-determinism**
- **Rewrite (simplify) DFTs** before analysis [Junges et al., 2017]
- **Partial state-space generation** [Volk et al., 2018]

Analysis by Partial State-Space Generation



Evaluation: DFT Analysis Times

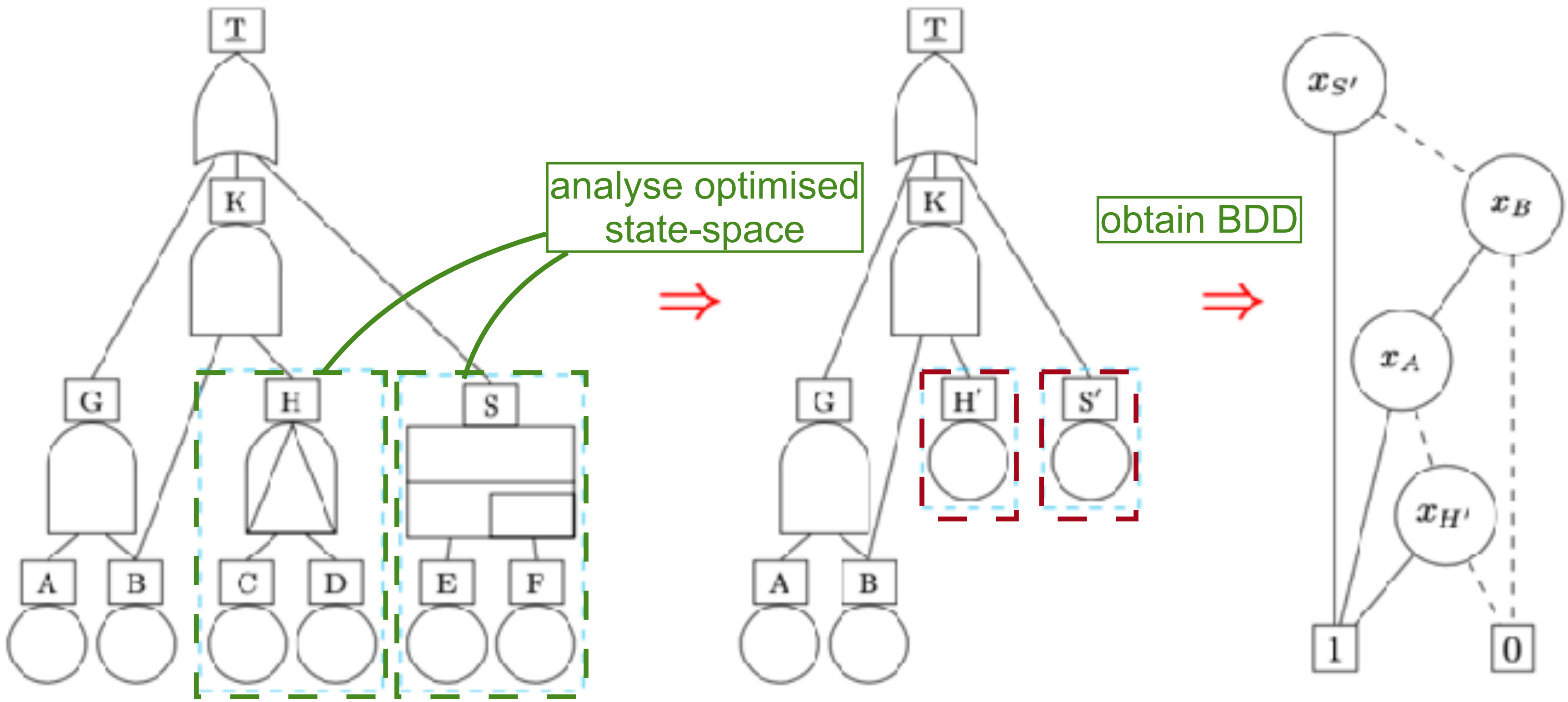


- ✓ Public FFORT benchmark suite
- ✓ Unreliability and MTTF
- ✓ 369 benchmarks
- ✓ Comparison to
  - ✓ DFTRes (2020, simulation)
  - ✓ DFTCalc (2013, compositional)
- ✓ 2.1 GHz, 16 GB RAM
- ✓ Error bound: $\mathbf{10^{-4}}$

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

What If DFTs Contain Large Static Parts?

[Basgöze et al., NASA FM 2022]



Experiments: DFTs with Static Parts

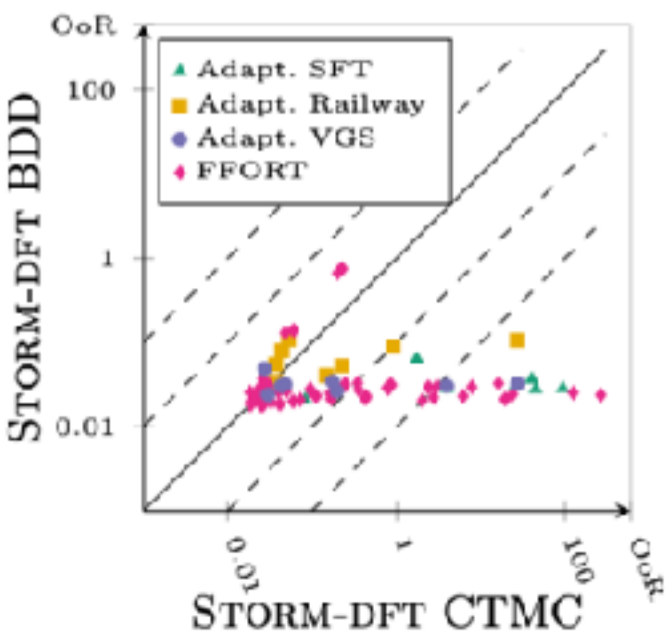
[Basgöze et al., NASA FM 2022]

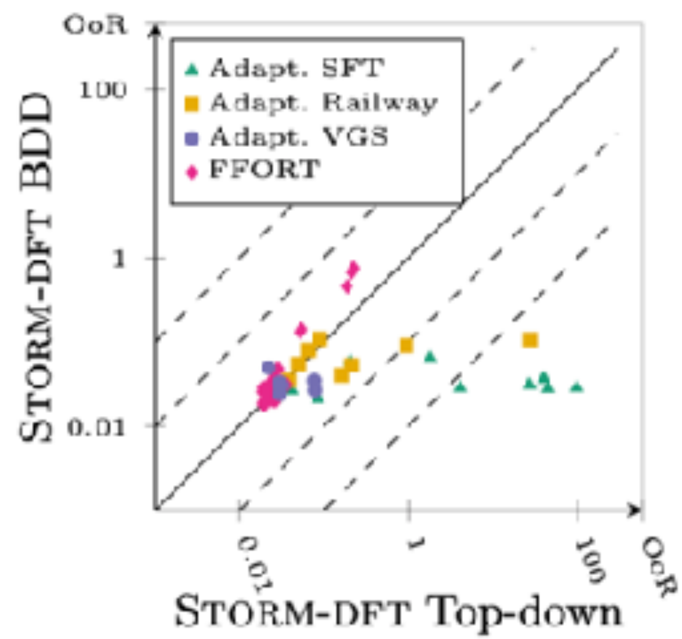
after modularisation

Storm

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

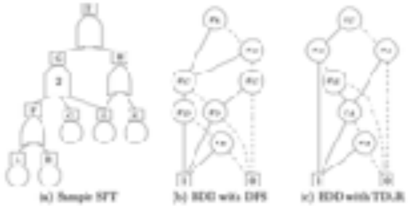
all run times in seconds



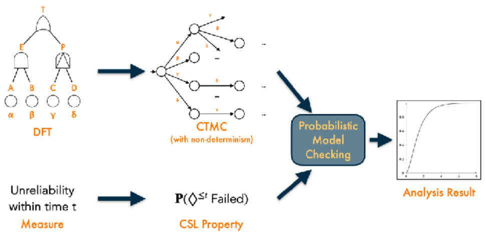


Storm-DFT outperforms Markov chain analysis and modularisation

Talk Overview

1. 
Classical Static Fault Tree Analysis

2. 
Dynamic Fault Trees

3. 

Scaling Up DFT Analysis

4. 
Industrial Case Studies

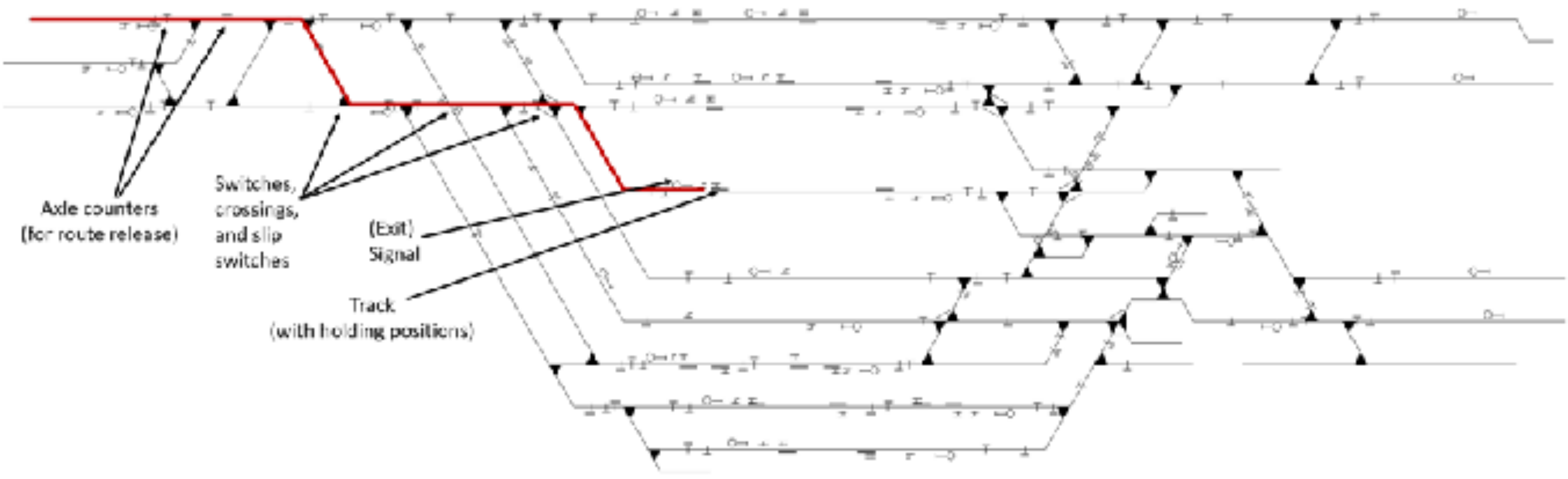
5. Storm
Storm Tool Demonstration

Criticality Assessment of Railway Infrastructures

[Weik et al., STTT 2022]

Criticality Assessment of Railway Stations

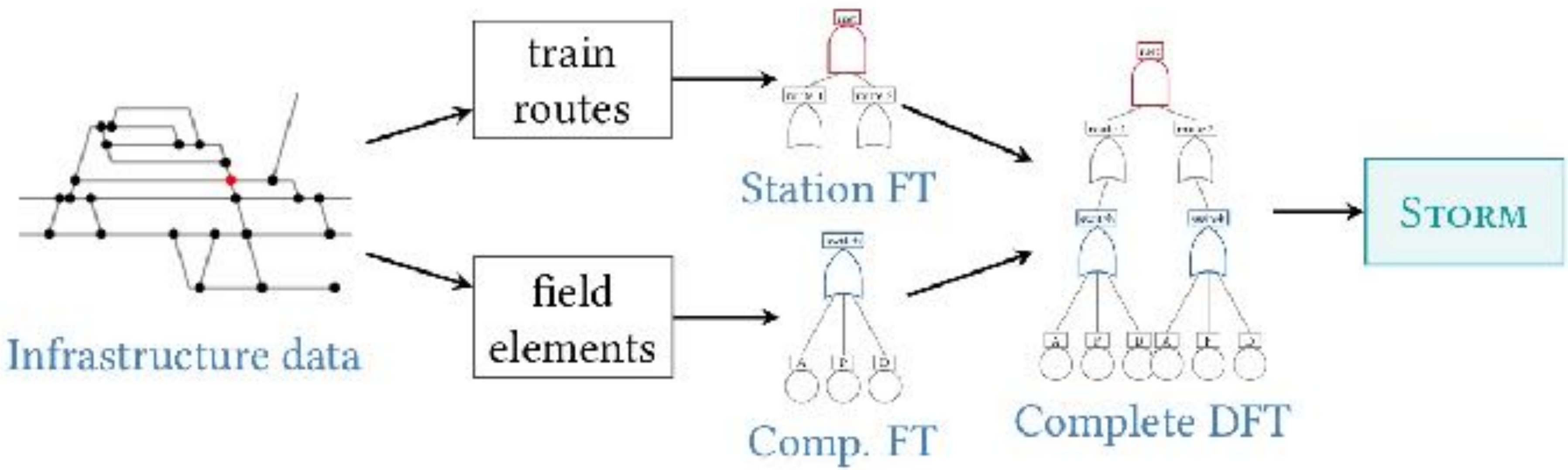
[Weik et al., STTT 2022]



train path must be set to run train

field elements must be operational and in correct position

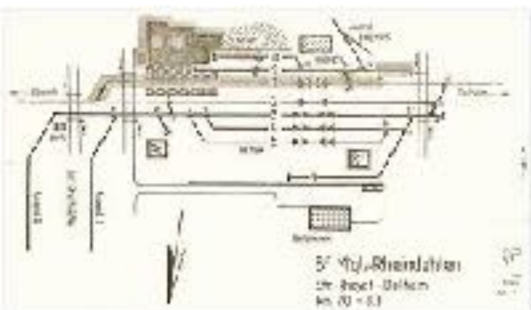
Criticality Assessment of Railway Infrastructures



Criticality Assessment of Railway Infrastructures

Storm

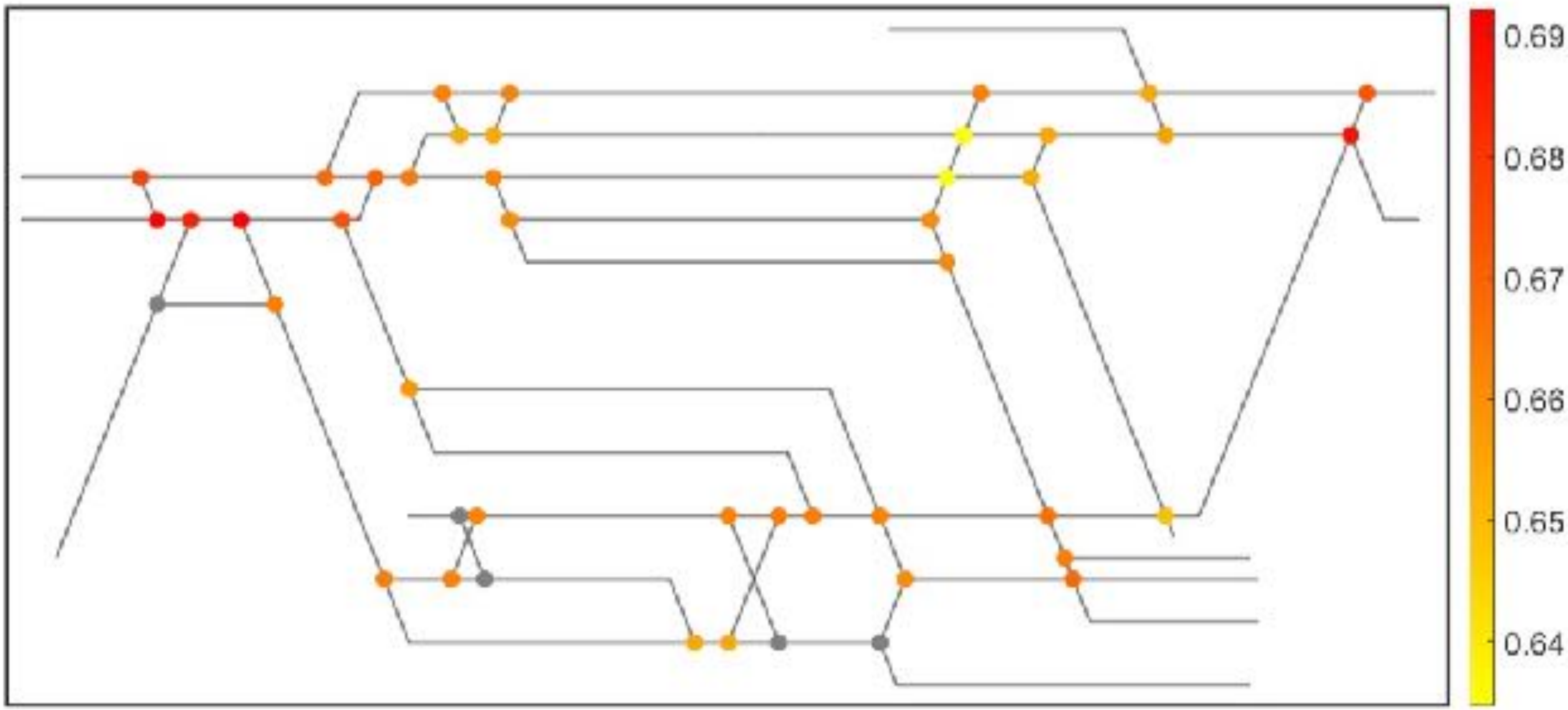
Scenario				Railway			
Id	Station	Variant	Max fail	#Route sets	#Routes	#Train paths	#Components
1	Aachen	std	∞	61	61	62	53
2		alt 5	4	23	115	41	54
3	Herzog.	std	∞	11	11	15	22
4		alt 5	4	9	19	15	24
5		alt 5	6	9	19	15	24
6	M'gladb	std	∞	26	26	32	40
7		alt 5	4	11	43	25	41



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 (DFT) — automatically generated (CTMC)

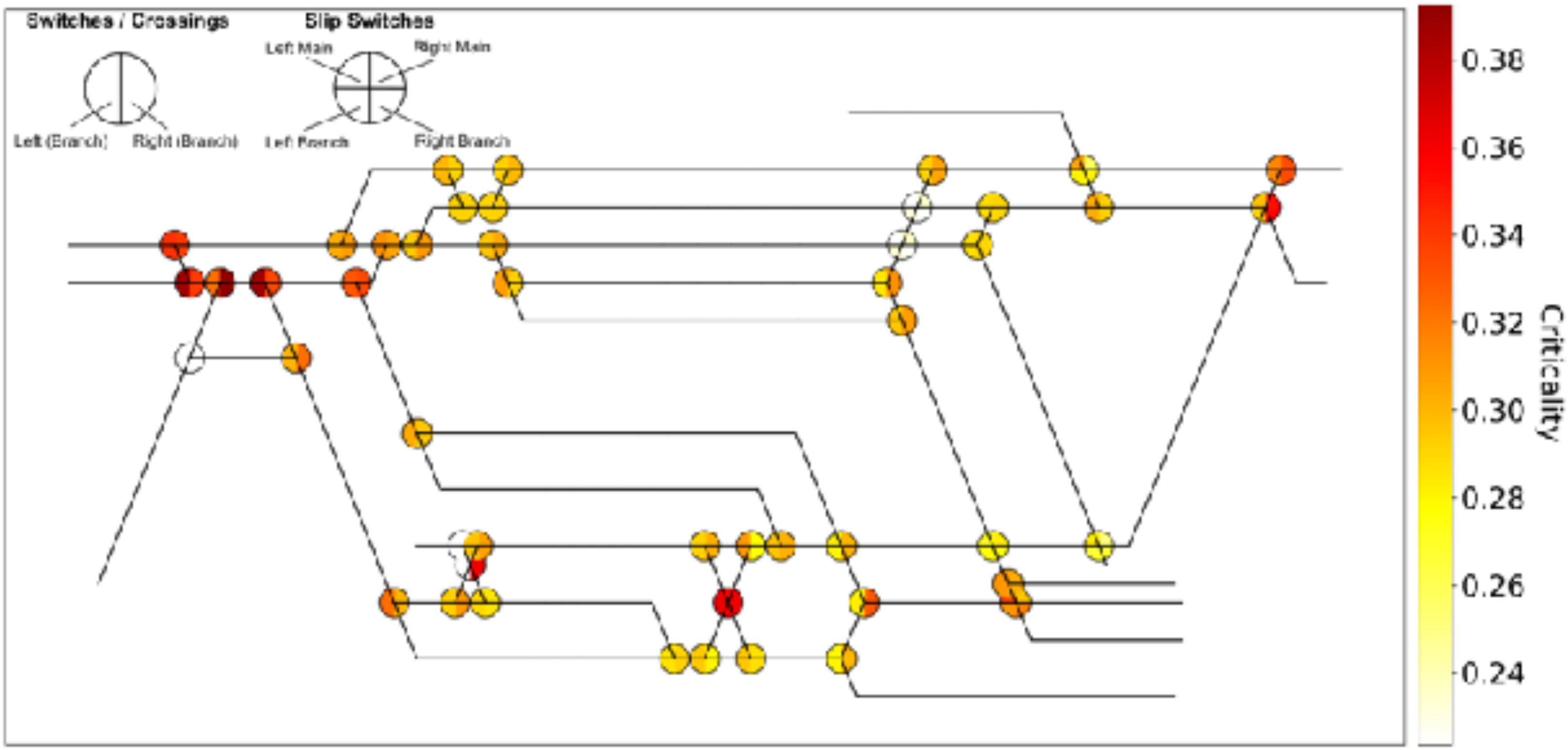
Criticality Assessment of Railway Infrastructures



Criticality of Mönchengladbach Hbf

Criticality Assessment of Railway Infrastructures

[Weik et al., STTT 2022]



Birnbaum importance index for switch branches
Mönchengladbach Hbf

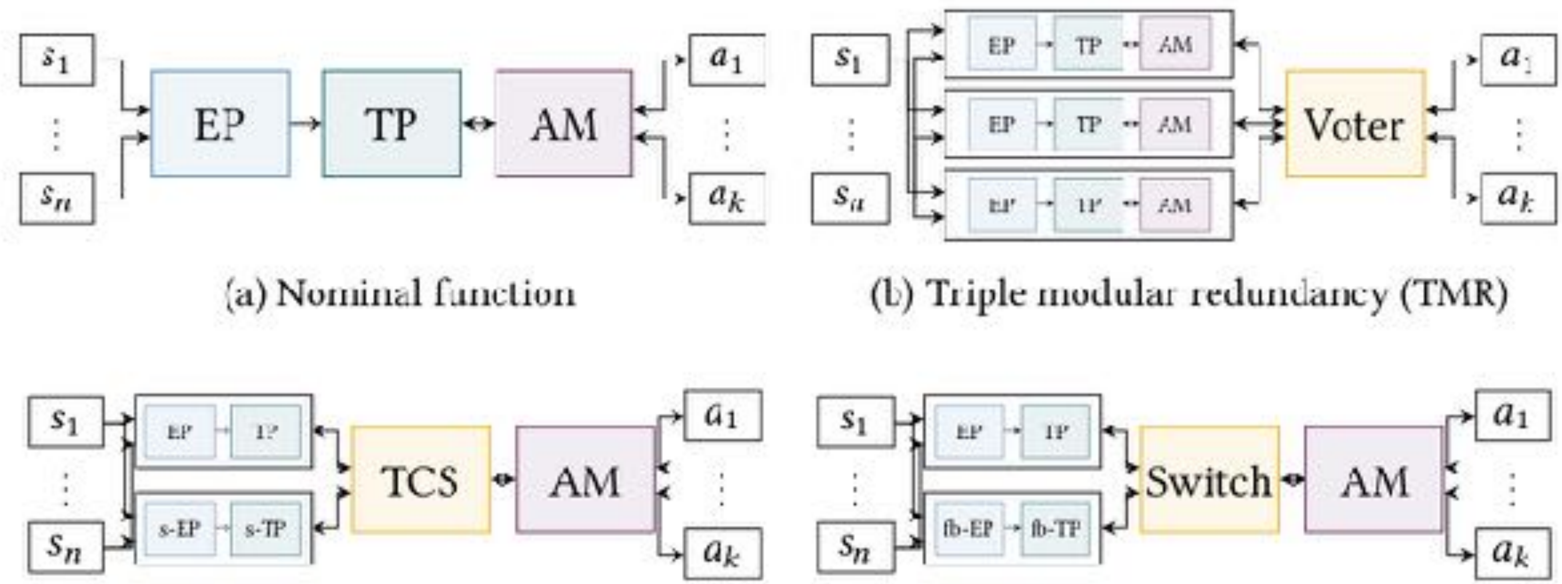
Autonomous Vehicle Guidance

[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

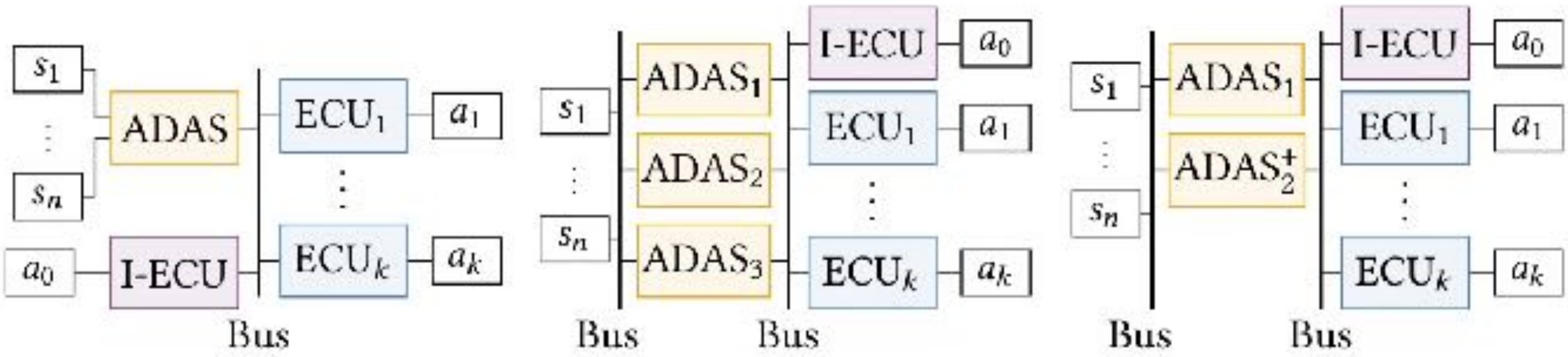
(d) Main path and fallback path

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

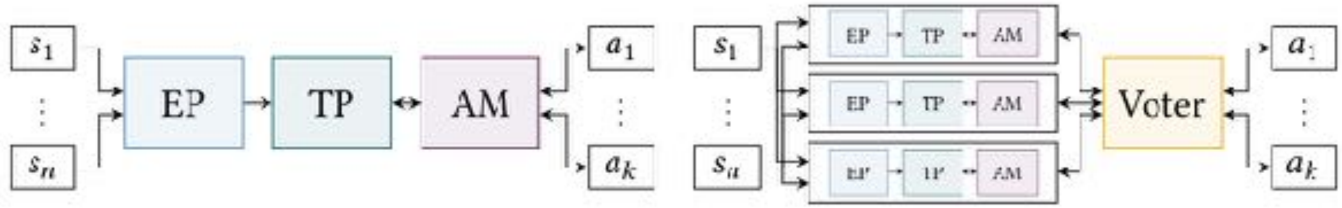
(a) nominal, (b) "TMR", and (c) ADAS+ architecture.

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

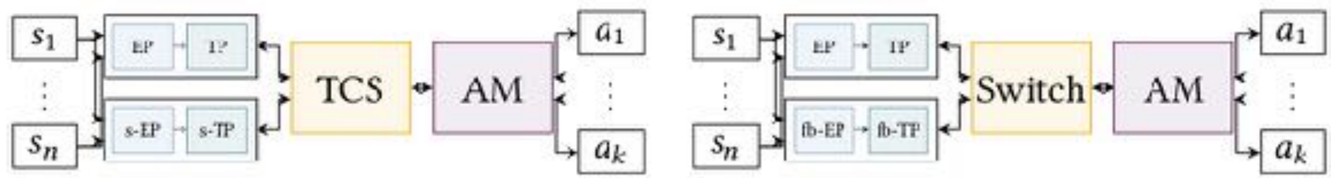
ADAS = Advanced Driver Assistance System, I-ECU = Integration ECU

Autonomous Vehicle Guidance

Software

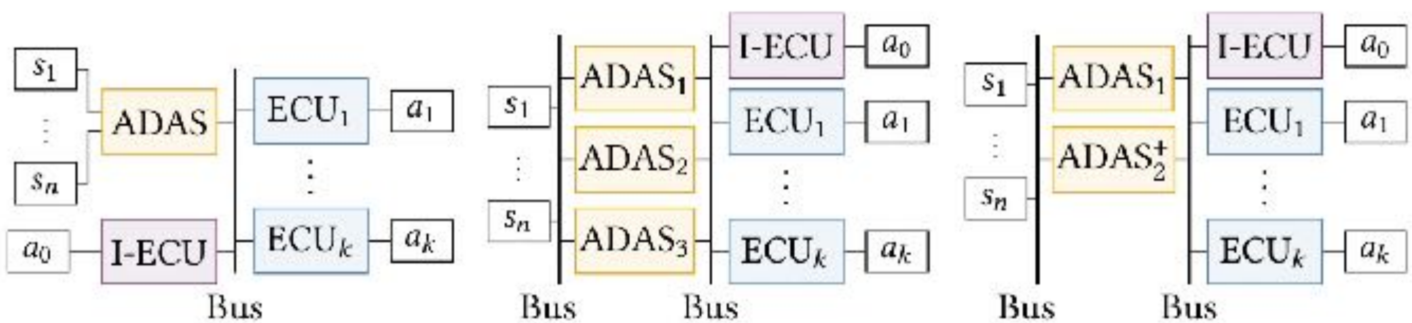


(a) Nominal function (b) Triple modular redundancy (TMR)



(c) Nominal path and safety path (d) Main path and fallback path

Hardware



Reliability Metrics Beyond Reliability and MTTF

System integrity ≈ 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?

Model Checking Queries

	Measure	Model Checking Queries
System	integrity	$1 - P(\Diamond^{\leq t} \text{ failed})$
	FIT	$\frac{1}{\text{lifetime}} \cdot (1 - P(\Diamond^{\leq \text{lifetime}} \text{ failed}))$
	MTTF	$\mathsf{ET}(\Diamond \text{ failed})$
Degradation	FFA	$1 - P(\Diamond^{\leq t} (\text{failed} \vee \text{degraded}))$
	FWD	$P((\neg\text{degraded}) \,\mathsf{U}^{\leq t}\, (\neg\text{degraded} \wedge \text{failed}))$
	MTDF	$\Sigma_{s \in \text{degraded}} (P(\neg\text{degraded} \,\mathsf{U}\, s) \cdot \mathsf{ET}^{s}(\Diamond \text{ failed}))$
	MDR	$\text{argmin}_{s \in \text{degraded}} (1 - P^{s}(\Diamond^{\leq t} \text{ failed}))$
	SILFO	$1 - \left(FWD + \Sigma_{s \in \text{degraded}} (P(\neg\text{degraded} \,\mathsf{U}^{\leq t}\, s) \cdot P^{s}(\Diamond^{\leq \text{drivecycle}} \text{ failed}))\right)$

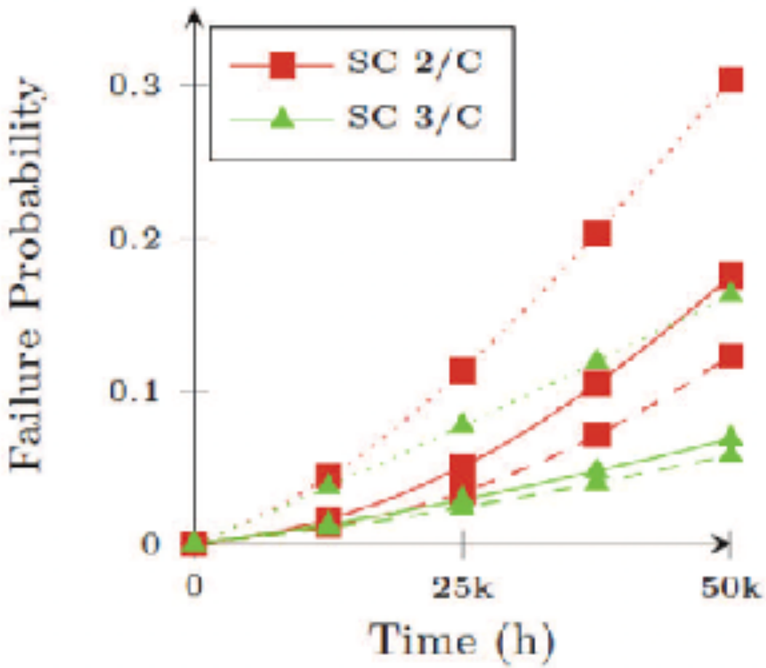
DFT Modeling Statistics

	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%

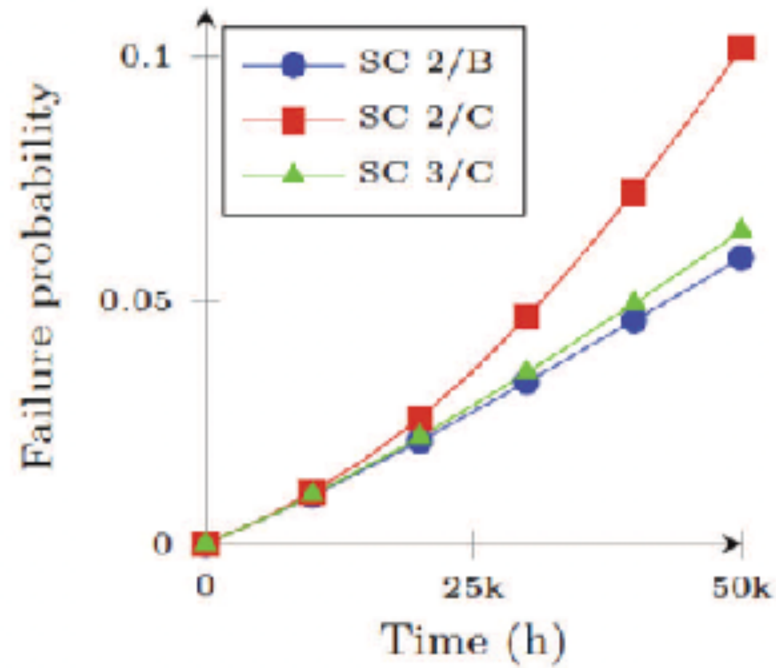
Storm

Analysis Results

[Ghadhab et al., RESS 2019]



Sensitivity



System integrity
after degradation

Nuclear Power Plant

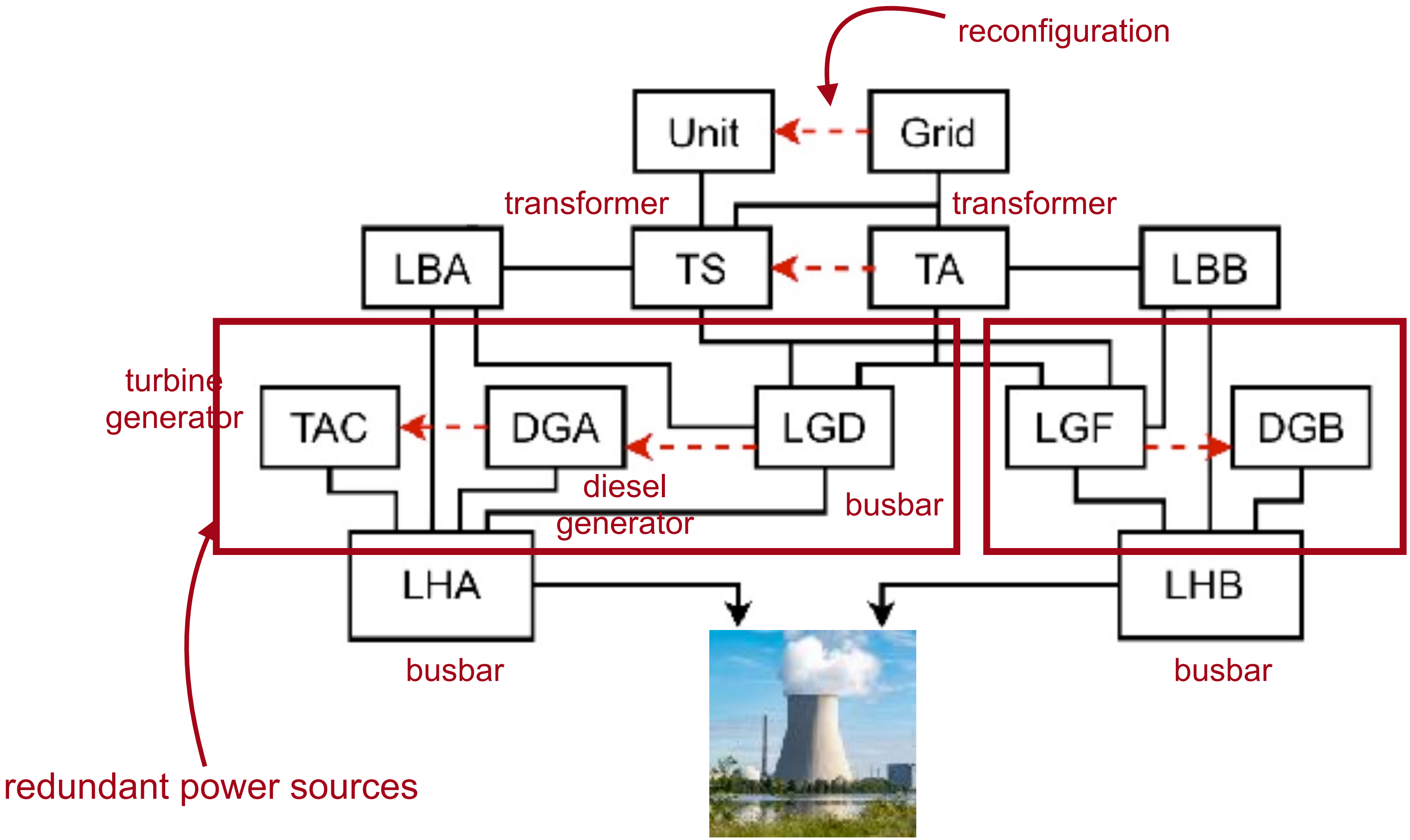
NPPS Benchmark

[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

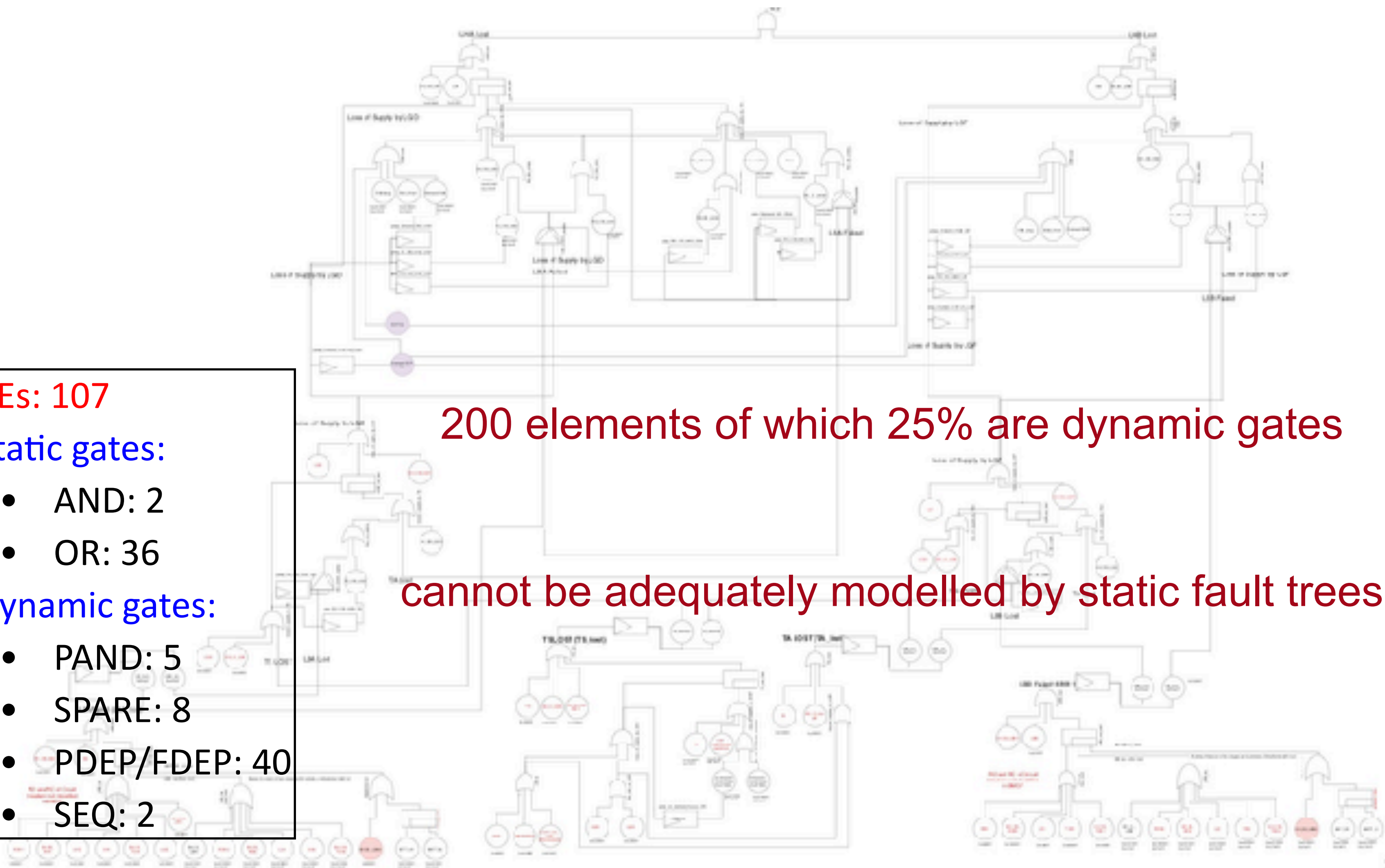
Nuclear Power Plant Power Supply

[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

Storm Evaluation Results

[Khan et al., DSN 2021]

Storm

Variant	Mission Time	STORM-FIGARO					
		State Space		Reported Bounds			CPU
		#state	#trans.	lb	ub	ub-lb	Time
Non repair-able	100 h	0.8 M	1.7 M	$3.4422E{-}06$	$3.4912E{-}06$	$4.9E{-}08$	14 m
		3.2 M	6.9 M	$3.4492E{-}06$	$3.4537E{-}06$	$4E{-}09$	59 m
	1000 h	0.8 M	1.7 M	$7.988E{-}03$	$7.991E{-}03$	$8.1E{-}05$	15 m
	10000 h	0.2 M	0.5 M	$3.593E{-}05$	$0.3608E{-}05$	$1.5E{-}06$	4 m
repair-able	10000 h	60 K	0.1 M	$3.538E{-}05$	$5.249E{-}05$	$1.7E{-}05$	1 m 30 s
	10000 h	0.1 M	0.4 M	$3.673E{-}05$	$3.834E{-}05$	$1E{-}07$	4 m 13 s
	10000 h*	0.3 M	0.8 M	$3.871E{-}06$	$4.235E{-}06$	$3E{-}07$	6 m 21 s

*Variant for sensitivity analysis

precision

Using analysis by partial-state space generation

What About Simulation?

Model checking

Pros

- No bias to certain scenarios
- (Mostly) complete coverage
- Precision almost for free
- Expressive properties

Cons

- State space explosion
- Computability
- Abstract models

Simulation

Pros

- Insensitive to state space
- Expressive models
- Detailed models

Cons

- Bias to certain scenarios
- Fatal unexplored scenarios
- No non-determinism
- High precision, high cost

model checking provides **better precision** than simulation

Reliability: Simulation vs. Storm

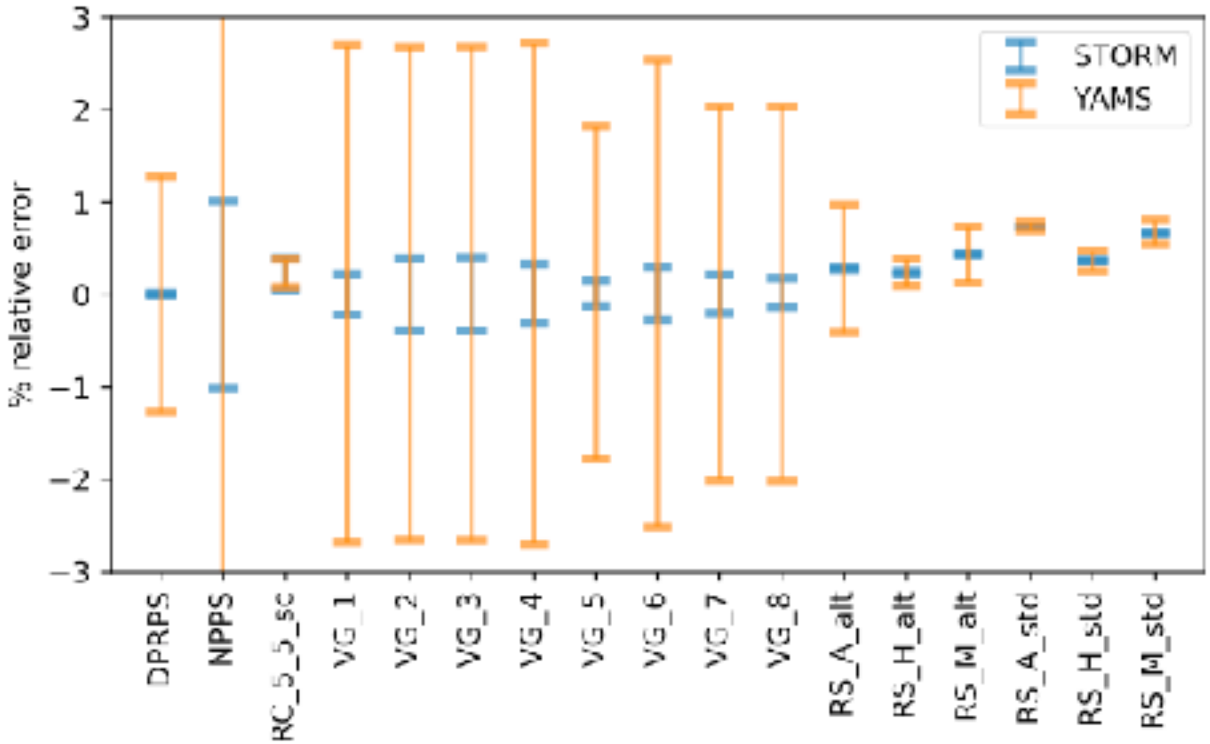
simulation

Benchmark	Non-Repairable State Space #States	Non-Repairable State Space #Trans.	Non-Repairable CPU Time STORM-FIGARO	Non-Repairable CPU Time YAMS	Repairable State Space #States	Repairable State Space #Trans.	Repairable CPU Time STORM-FIGARO	Repairable CPU Time YAMS
DPRRS	2 K	5 K	0.7 s	16.5 m	2 K	6 K	1.5 s	1.3 h
NPPS	10.3 M	21 M	2.7 h	1.5 h	0.48 M	1.1 M	10.4 m	1.4 h
RC_5_5_sc	1 K	3 K	0.1 s	2 m	1 K	6 K	0.3 s	4 m
VG_1	0.1 M	0.3 M	85 s	28 m	8 K	19 K	7 s	37 m
VG_2	0.2 M	0.6 M	2 m	22 m	7 K	15 K	7 s	51 m
VG_3	25 K	57 K	14 s	20 m	3 K	7.5 K	3 s	41 m
VG_4	12 K	28 K	6 s	8 m	1.6 K	3.7 K	1 s	11.8 m
VG_5	2 K	4.7 K	1 s	8 m	614	1.4 K	0.6 s	14 m
VG_6	0.05 M	0.1 M	38 s	21 m	3 K	8 K	3 s	48.5 m
VG_7	3.2 M	7.7 M	43 m	32 m	1.7 K	4.2 K	23 s	59 m
VG_8	18.9 M	45.8 M	8.8 h	13 m	0.87 M	1.8 M	18.5 m	3.45 h

Precision

[Khan et al., NASA FM 2021]

repairable models



YAMS:
10^7 simulations

Storm:
precision **10^{-3}**

probabilistic model checking:
provides **better precision** than simulation
supports **metrics beyond standard** reliability, availability, MTTF

Take-Home Messages

What?

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

How?

- Slim state-space generation +
- Efficient Markov chain model checking

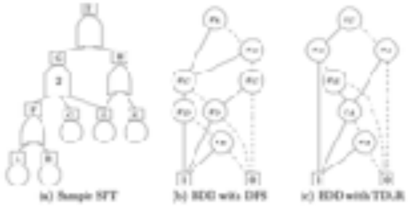
Try it out

https://www.stormchecker.org

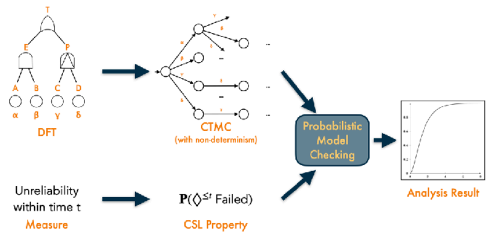
Storm

No myths.

Talk Overview

1. 
Classical Static Fault Tree Analysis

2. 
Dynamic Fault Trees

3. 

Scaling Up DFT Analysis

4.   
Industrial Case Studies

5. Storm
Storm Tool Demonstration

DGB Technologies

www.dgbtek.com

Implementing Next Generation Ideas

USA: 393 Crescent Ave.
Wyckoff NJ 7481

Germany: Keetman Str. 01,
47058 Duisburg

Pakistan: 21 CC, Parkview,
DHA-8, Lahore

POC: Falak Sher
chfalak@dgbtek.com
www.dgbtek.com

Leadership

Dr. Falak Sher (Formal Methods)
RWTH Aachen University - Germany
Exec. Director DGB Technologies

Ahmad Zafar (Formal Methods)
MS ITU Lahore, BS Fast Lahore
Publications 2+, LADC

Dr. Omer Beg (AI/ML/NLP)
PhD, Waterloo, Canada
Publications 50+

Scientific Advisory Board

Prof. Dr. Ir. Dr. h. c. Joost-Pieter Katoen
Head Software modeling and verification group MOVES
RWTH Aachen University - Germany

Dr. Emran Khan (PhD. Molecular Biology)
Ph.D RWTH Aachen University - Germany
Postdoc. Vienna Inst. for Biotechnology - Austria, Researcher UPenn - USA, Publications 20+

Dr. Maham Abbas Mela (PhD. Economics)
PhD Columbia University, USA
MS Stanford University, USA

R&D Team

Dr. Agha Ali Raza (Machine Learning & NLP)
PhD Carnegie Mellon University, USA
Publications 30+, CHI, InterSPEECH

Dr. Arif Mehmood (Computer Vision)
PhD, LUMS | Postdoc Qatar University
Publications 50+, TIP, CVPR

Dr. Mohsen Ali (Machine Learning)
PhD University of Florida, USA)
Publications 18, ICCV, CVPR, ECML

Dr. Maryam Mustafa (HCI, AR/VR)
PhD, TU Braunschweig, Germany
M.Eng, Cornell University, New York
Publications 20+, CHI, TAP, SIGCHI

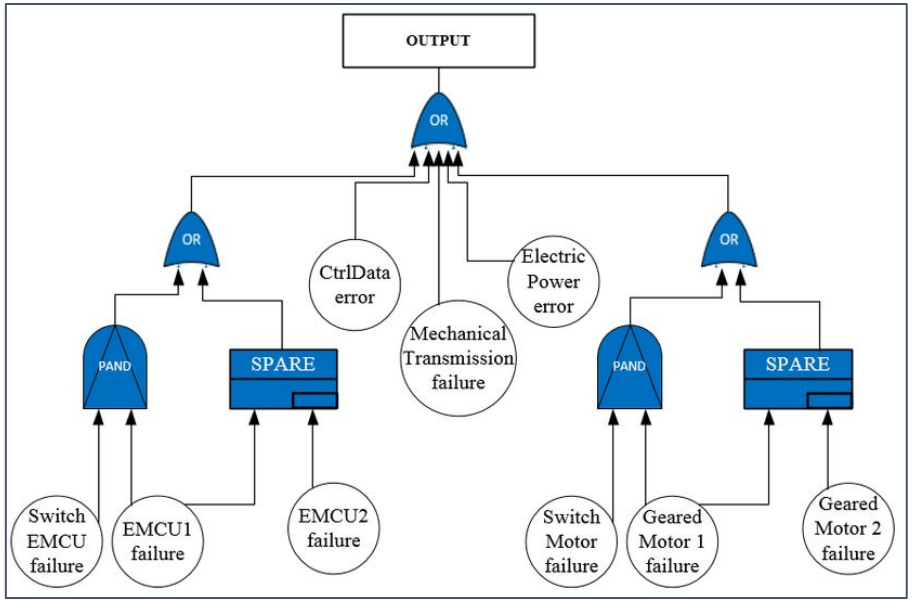
Dr. Ali Ahmed (Artificial Intelligence)
PhD, Postdoc MIT, USA
Publications 30+, IEEE TIT, NIPS

Quantitative Analysis Tools Based on Probabilistic Model-checking

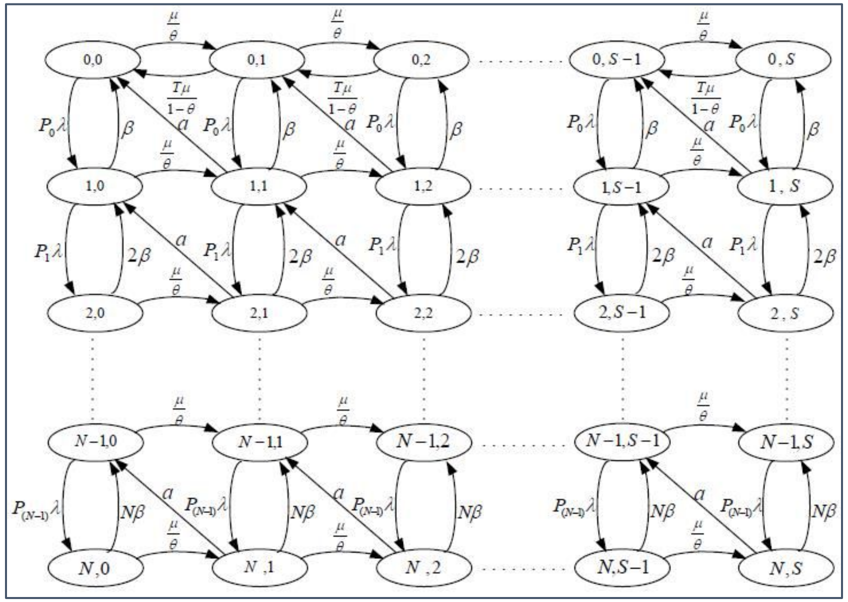
DGB builds tools for the analysis of stochastic systems modeled as:

- ➢ [Markov automata (MA)]
- ➢ [Dynamic fault trees (DFTs)]
- ➢ [Generalized Stochastic Petri nets (GSPN)]

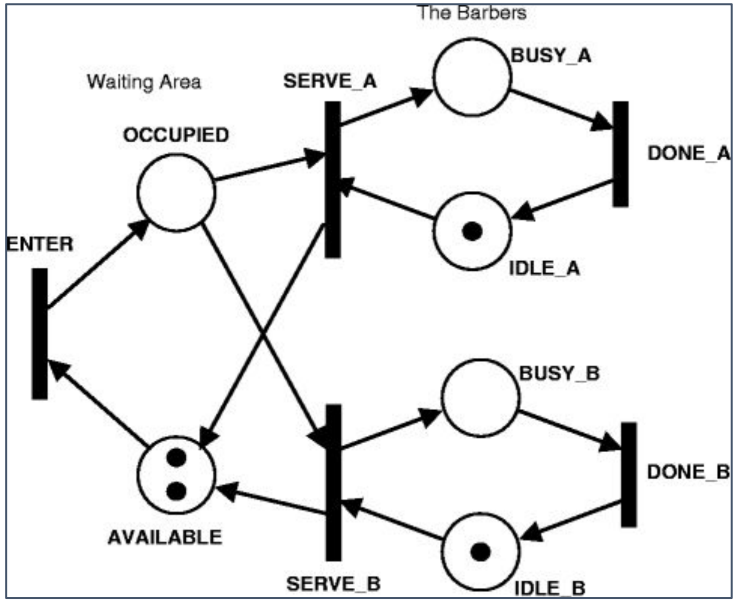
based on state-of-the-art probabilistic model-checker **STORM**.



Fault Trees



Markov automata



Petri Nets

Dynamic Fault Trees for Probabilistic Risk Assessment

Background

Quantitative risk assessment is a fundamental action to ensure safe operations of critical high-tech fail-operational systems. The rigorous and powerful risk assessment in the development of systems is more important than ever because:

The international standards have increased safety constraints e.g. ISO 26262 for autonomous driving.

There is an ever-growing penetration of AI/ML components in the systems.

Various techniques have been developed throughout the years to analyze the safety and reliability of systems.

One of the most relevant is Fault Tree Analysis (FTA) applied by millions of engineers to many safety-critical systems.

Their use is required for instance by the Federal Aviation Authority (FAA) , the Nuclear Regulatory Commission (NRC) USA, space agencies like NASA and ESA.

Dynamic Fault Trees (DFTs)

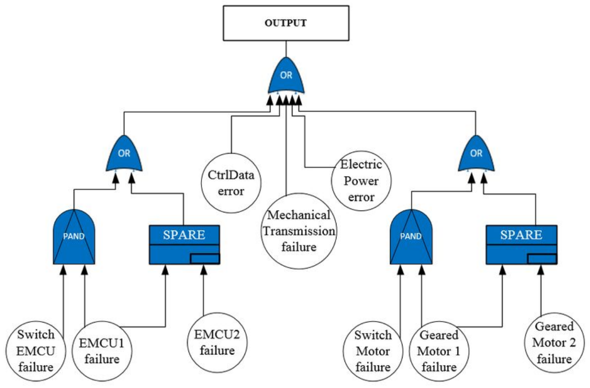
While fault trees are easy to understand and thus, widely used, their modeling capabilities are severely limited. This lack of flexibility hinders accurate and precise modeling of real-life systems e.g., self-driving cars, hyperloops and drones. DFTs, co-developed with NASA, overcome these deficiencies and faithfully model fail-operational systems having

Redundant components

Probabilistic dependencies e.g. CCF

Temporal dependencies

Non-deterministic behaviour



Quantitative Measures

While fault tree models represent how failures occur at system component level and how they propagate through sub-systems, eventually leading to system level failures, their analysis focuses on computing various dependability metrics, i.e. key performance indicators that measure how well a system performs. Standard metrics are the systems:

Reliability: The probability that no failure occurred until time T.

Conditional Reliability: The probability that no failure occurred until time T given a component has already failed.

Availability: The average percentage of time that a system is operational.

Mean time to failure: The mean time between system failures.

Criticality of components: To what extent does a component failure contribute to a system failure.

Various extensions of these measures include the cost and impact of failures.

Dynamic Fault Trees for Probabilistic Risk Assessment

Why do we need fault tree analysis for risk assessment?

- Depict the logical relationship between a system failure and its contributing causes graphically
- Quantify the probability of system failure based on its components and the logic of its architecture
- Allocate the safety requirements of the system to its components
- Assess the effects of single and combined failures
- Assess the effects of the exposure time of the hidden failures on the system safety
- Assess the source of common cause failures
- Assess the nature of fail-safe design (fault tolerance and error tolerance)
- Assess the effects of design change on its safety
- Figure out the optimal design wrt cost
- Most widely used technique for Reliability, Maintainability and Safety Analysis worldwide
- International standards require rigorous and powerful fault tree analysis techniques e.g. ISO 26262 for automotive
- Rapidly increased usage of AI components in modern systems necessitates a rigorous risk assessment

Sample risks which stakeholders analyse in different industries

- Aviation
  - Plane avionics fail in midair
  - Engine fails at takeoff
  - Emergency doors open in midair
- Automotive
  - Rear view cameras stop working
  - Lane warning systems behave abnormally
  - Gearing system stops working
- Defence
  - A weapon malfunctions at the time of use
  - A weapon activates prematurely
  - A weapon misses its intended target
- Medical
  - Ventilator stops working for a critical patient
  - Pace-make behaves abnormally
  - Radiation dose is not controlled properly
  - Blood pressure is not measured properly

Dynamic Fault Trees Analysis Tool

www.dgbtek.com

Features of our DFT analysis tool

➢ The unique tool for formal analysis of dynamic fault trees (DFT)
  ○ DFTs were co-developed with NASA for risk assessment

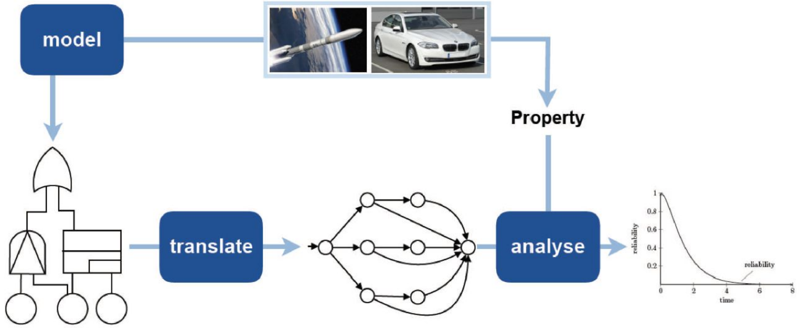
➢ It faithfully models fail-operational systems that have
  ○ Redundant components
  ○ Probabilistic dependencies among components e.g. CCF
  ○ Temporal dependencies of components, and
  ○ Non-deterministic behaviour

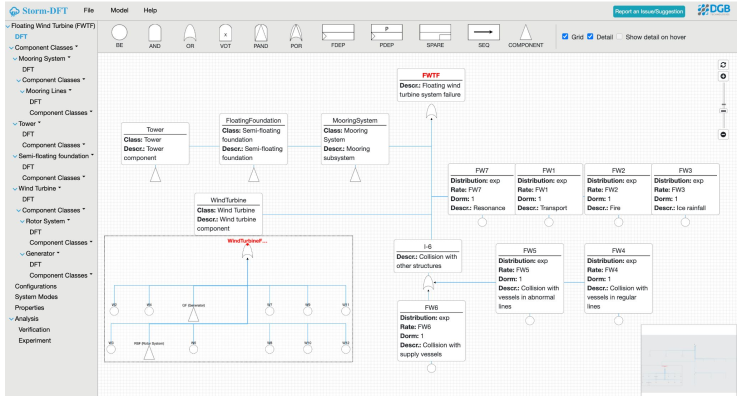
➢ The analysis is based on the theory of probabilistic model-checking
  ○ Formally proven algorithms published in top venues
  ○ The fastest algorithms – won QComp 2019-20 competitions
  ○ Provides hard probabilistic guarantees instead of statistical ones

➢ Web-based graphical interface: drag-&-drop, simulation, experimentation

➢ Algorithms used in projects with BMW, German Railway, EDF (Électricité de France)

➢ Co-developed with MOVES@RWTH and FMT@Twente Universities – top R&D centers in Germany and The Netherlands





Risk Assessment Measures Verifiable by Our Tool

www.dgbtek.com

Verifiable Quantitative Measures

- Probability that a system will fail within a given time period – reliability
- Probability that a system is fully functional (no redundant comp. failure) within a given time period – full-functional availability
- Probability that a system will fail within a given time period before any of its redundant component fails – failure without degradation
- The expected time a system takes to fail when it operates with a limited functionality (due to e.g. a redundant component failure) – mean-time from degradation to failure
- The criticality of a degraded state, in terms of the probability that the system fails within e.g. a typical drive cycle of one hour while being degraded already
- The effect on the overall system reliability when imposing limits on the time a system remains operational in a degraded state
- Identification of critical components (with high failure probability) within a given time, and
- Many more CS Logic-specified measures

Formal Methods Experts

Dr. Falak Sher
Ph.D. RWTH Aachen University - Germany
Formal Methods Expert
CEO DGB Technologies LLC
chfalak@dgbtek.com

Prof. Dr. Ir. Dr. h. c. Joost-Pieter Katoen
Head of modeling and verification group
RWTH Aachen University - Germany
Consultant DGB Technologies LLC
katoen@cs.rwth-aachen.de

DGB Offices

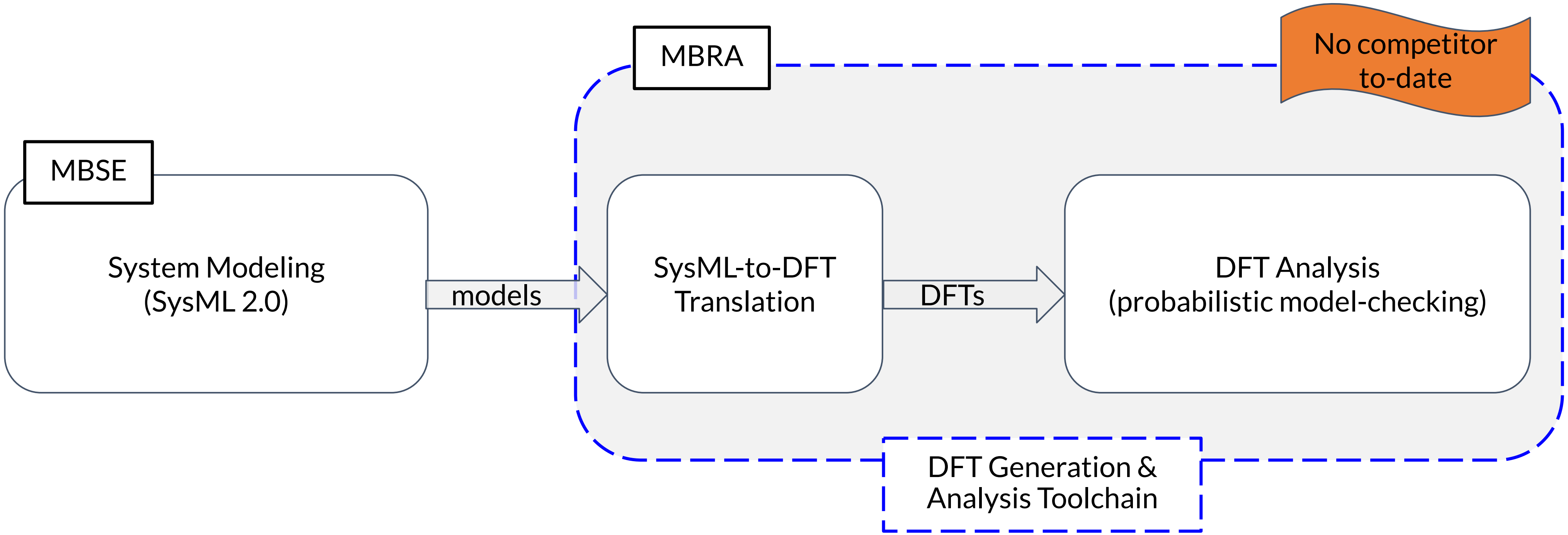
USA: 393 Crescent Ave.
Wyckoff NJ 7481
+1 (201) 466-7066

Germany: DGB Technologies
Keetman Str. 01, 47058 Duisburg
T: +49 176 346 74943

Pakistan: DGB Technologies
21 CC, DHA 8 Ex-Parkview, Lahore
T: +92 333 474 4438

Toolchain for Generation & Analysis of DFTs from SysML 2.0

We build a toolchain to automate model-based risk assessment (MBRA) in parallel with model-based systems engineering (MBSE).

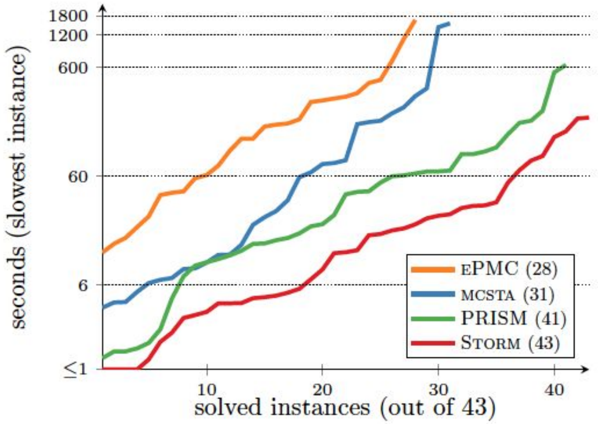


Storm: A Markov Analysis tool based on model-checking

www.dgbtek.com

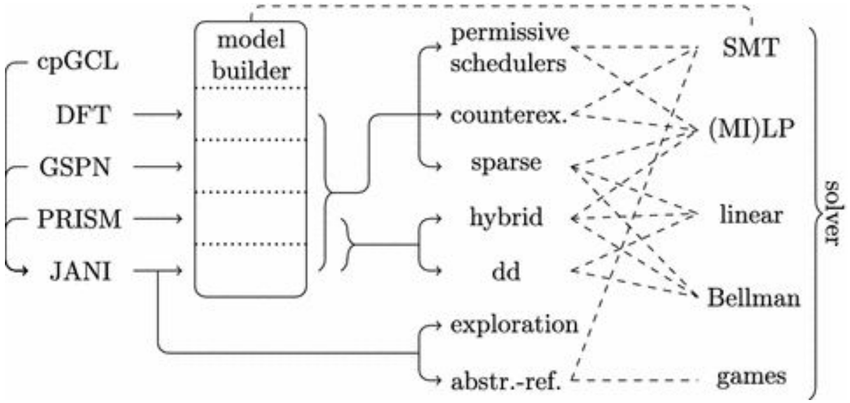
It is fast, often the fastest

"*overall, the Storm dominates the competition*" [QComp 2020]



It supports multiple input languages

- JANI:
  - Intermediate language for many probabilistic model checkers
- Generalized Stochastic Petri Nets (GSPNs):
  - Petri nets with "exponential" and "immediate" transitions
  - Storm supports *Confused* GSPNs
  - Prominent in performance and dependability analysis
- Dynamic Fault Trees (DFTs):
  - Dugan's DFTs with *p-FDEPs* and "nested" SPAREs, etc.
  - Tailored state-space generation and reduction techniques
  - Prominent in reliability engineering

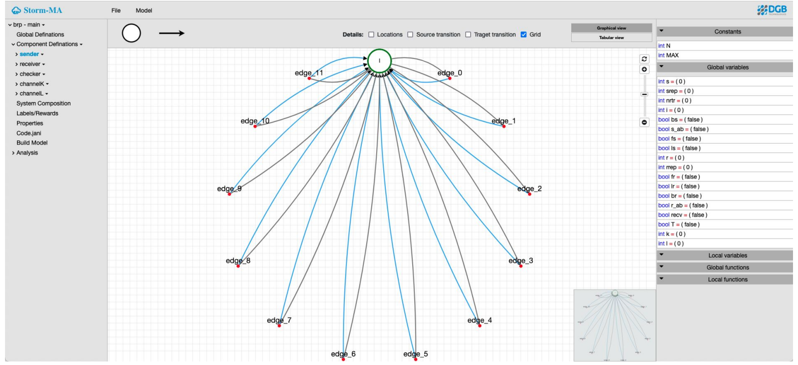


It is modular

- easy exchange of solvers and symbolic engines
- enables rapid prototyping, via Python APIs

It has web-based graphical interface (GUI)

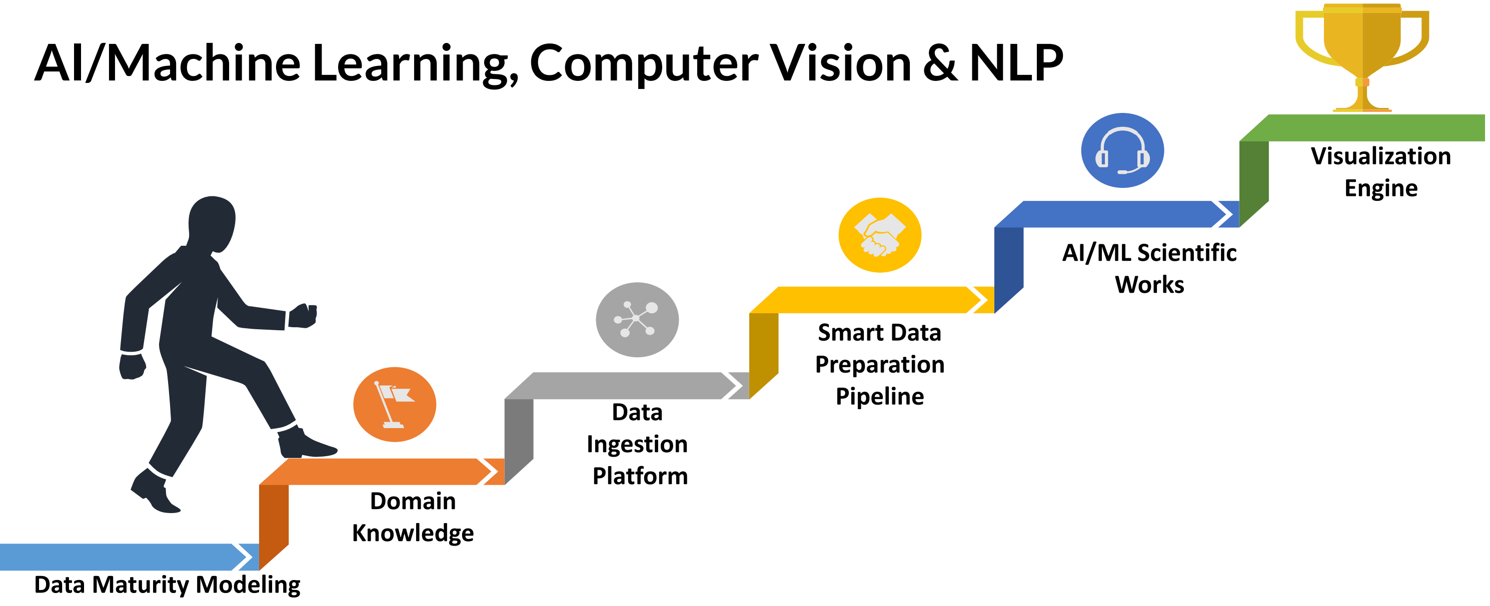
- drag & drop editors for Markov automata, DFT* and GSPN*
- automatic translation to intermediate JANI language, etc.



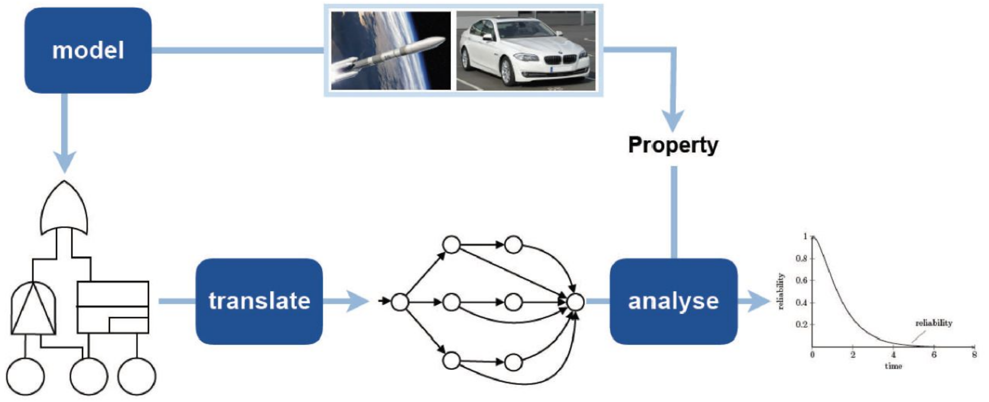
DGB Verticals

www.dgbtek.com

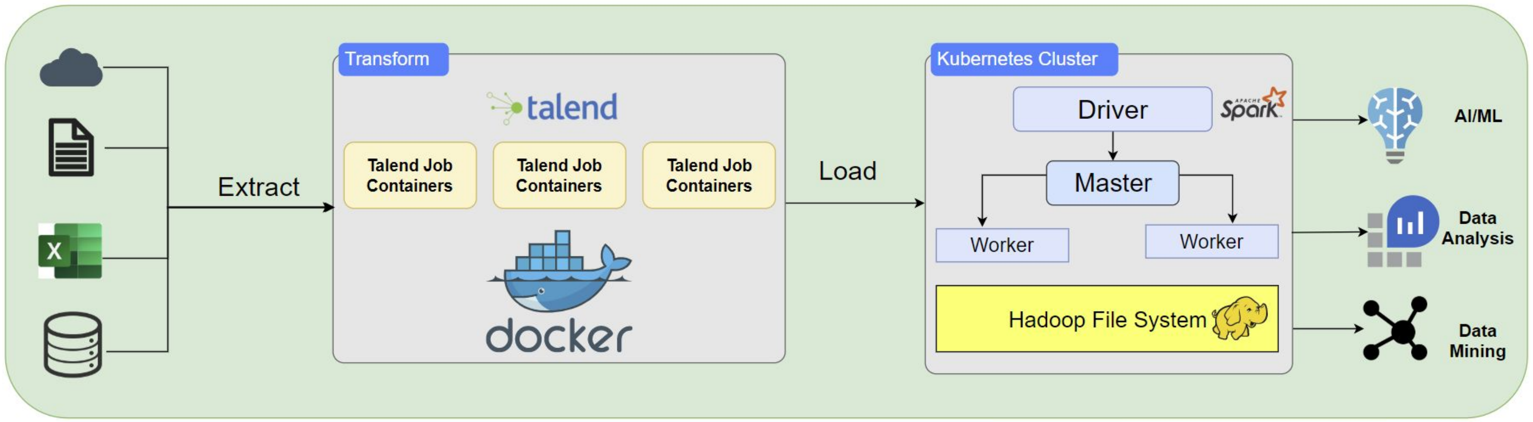
AI/Machine Learning, Computer Vision & NLP



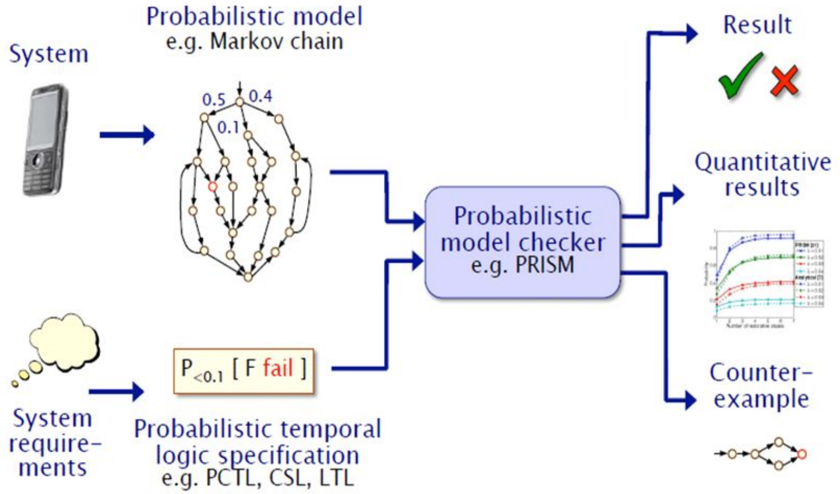
Reliability Analysis



Big Data Analytics and Cloud computing



Stochastic Verification



Sample Industrial Projects

★ IBNR Prediction *(AI/ML)*
(Wiseman Innovations- USA, https://wisemaninnovations.com)

Predict the total cost incurred in a month by an Accountable Care Organization (ACO) in the USA using partial available claims information using time forecasting algorithms like LSTM

★ Textual Data Analytics *(AI/ML/NLP)*
(Grunenthal- Germany, https://www.grunenthal.de)

Transformed data into MySQL and apply multiple AI/ML models like Regression, Clustering, Summarize Text and Word2Vec for quantitative and qualitative analysis on pharmaceutical data to better understand the needs of targets.

★ SysmL to DFT Translator*(Formal)*
(Robert Bosch - Germany, https://www.bosch.de)

Build a toolchain to automate model-based risk assessment (MBRA) in parallel with model-based systems engineering (MBSE) using System Modeling Language SysML.

★ Cloud-based Big Data Infrastructure *(AI/ML/Big Data)*
(Integ Consulting - USA, https://www.integconsulting.com/)

★ Created a scalable architecture using restful FASTAPI server and Spark to perform dynamic ETL on big data workloads.

★ Developed an end-to-end AI/ML pipeline which includes data preprocessing, model training, deployment and inference using AWS SageMaker. Used SageMaker builtin algorithms like XGboost for regression, binary and multi-class classification, RCF for anomaly detection and more.

★ Developing an AI/ML infrastructure using Spark and AWS services like Lambda, SageMaker, S3, ECR, EMR, Glue and more.

★ Skills: Apache Kafka, Amazon Web Services (AWS), Machine Learning, Docker Products, PySpark, AWS SageMaker

Clients and Collaborators

RWTH AACHEN UNIVERSITY

University of Twente
The Netherlands

IBM

NATIONAL RADIO TELECOM CORPORATION
N R T C
PAKISTAN

BOSCH

R Studio

integ

wiseman
innovations

IMS
Integrated Medical Sensors

GRÜNENTHAL

Loop Network