One Net Fits All

A Unifying Semantics of Dynamic Fault Trees Using Generalised Stochastic Petri Nets

Matthias Volk

Joint work with:
• Sebastian Junges (RWTH Aachen University)
• Joost-Pieter Katoen (RWTH Aachen University, University of Twente)
• Mariëlle Stoelinga (University of Twente, Radboud University Nijmegen)

UnRAVeL bi-weekly meeting
September 5, 2018
Outline

- DFT example
- GSPNs
- Templates
- Semantic issues
- Properties
DFT Example
Fault Tree Analysis

- Used in safety and reliability engineering
- Top-down failure analysis
- Models how a complex system fails due to subcomponent failures
Fault Tree Analysis

- Used in safety and reliability engineering
- Top-down failure analysis
- Models how a complex system fails due to subcomponent failures
Applications of Fault Tree Analysis
Applications of Fault Tree Analysis

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

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

- CPU failure
- Memory failure
- Power failure

Server failure
Fault Tree Example: Server
Fault Tree Example: Server

- Server failure
  - CPU failure
    - 2/4
  - Memory failure
  - Power failure
Fault Tree Example: Server

Static Fault Tree

- combinations of failures
- logical gates
Fault Tree Example: Server

Dynamic Fault Tree
- **order** of failures
- **additional** gates:
  - dependencies, **spare** components, **order-dependent** failures
Fault Tree Example: Server

- Server failure
  - CPU failure
    - 2/4
  - Memory failure
  - Power failure
Fault Tree Example: Server

- Server failure
- Memory failure
- Power failure

- CPU failure
- 2/4
- Cooling failure
Fault Tree Example: Server

In the diagram, the server failure is shown as the top event. The critical components that lead to this failure include:

- CPU failure
- Memory failure
- Cooling failure
- Power failure

Below the diagram, there is a list of incident/fault reports (in German only):

<table>
<thead>
<tr>
<th>Dienst</th>
<th>Kurz-Info</th>
<th>ab</th>
<th>bis</th>
<th>RSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW Proxy Server</td>
<td>Schwache Performance</td>
<td>28.08.2017</td>
<td>unbekannt</td>
<td></td>
</tr>
<tr>
<td>Serverhosting</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web-Server</td>
<td>07.08.2018</td>
<td>unbekannt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IT Center Infrastruktur</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Antrago</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Veranstaltungsmanagement - ZHV</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAP - ZHV</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sophos Antivirensoftware</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Windows Updates</td>
<td>04.09.2018</td>
<td>05.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rechner-Cluster</td>
<td>31.08.2018</td>
<td>unbekannt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WirelessLAN (MoPS, eduroam)</td>
<td>11.07.2018</td>
<td>unbekannt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mailingliste Mailman</td>
<td>07.08.2018</td>
<td>05.10.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sciebo</td>
<td>26.06.2018</td>
<td>unbekannt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTHmoodle</td>
<td>angekündigt: Wartung</td>
<td>05.09.2018</td>
<td>05.09.2018</td>
<td></td>
</tr>
<tr>
<td>Backup- und Archivsystem</td>
<td>04.09.2018</td>
<td>04.09.2018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTHonline</td>
<td>angekündigt: Wartung</td>
<td>05.09.2018</td>
<td>05.09.2018</td>
<td></td>
</tr>
</tbody>
</table>
Fault Tree Example: Server

Server failure

CPU failure

2/4

Memory failure

Cooling failure

Power failure
Fault Tree Example: Server

- Server failure
- CPU failure
- Memory failure
- Cooling failure
- Power failure
- Power UPS
Fault Tree Example: Server
DFT Timeline
DFT Timeline

Introduction of Dynamic Fault Trees
Dugan, Bavuso, Boyd

1992
DFT Timeline

Introduction of Dynamic Fault Trees
Dugan, Bavuso, Boyd

1992

Semantics on Monolithic CTMCs
Coppit, Sullivan, Dugan

1999
DFT Timeline

Introduction of Dynamic Fault Trees
Dugan, Bavuso, Boyd
1992

Semantics on GSPNs
Codetta-Raiteri
1999

Semantics on Monolithic CTMCs
Coppit, Sullivan, Dugan
2005
DFT Timeline

Introduction of Dynamic Fault Trees
Dugan, Bavuso, Boyd

Semantics on GSPNs
Codetta-Raiteri

1992
1999
2005
2007

Semantics on Monolithic CTMCs
Coppit, Sullivan, Dugan

Semantics on I/O-IMCs
Boudali, Crouzen, Stoelinga
DFT Timeline

- **Introduction of Dynamic Fault Trees**
  - Dugan, Bavuso, Boyd
  - 1992

- **Semantics on GSPNs**
  - Codetta-Raiteri
  - 1999

- **Semantics on Monolithic MAs**
  - Volk, Junges, Katoen
  - 2005

- **Semantics on Monolithic CTMCs**
  - Coppit, Sullivan, Dugan
  - 2007

- **Semantics on I/O-IMCs**
  - Boudali, Crouzen, Stoelinga
  - 2016
## DFT Semantics

<table>
<thead>
<tr>
<th></th>
<th>Semantics on Monolithic CTMCs</th>
<th>Semantics on I/O-IMCs</th>
<th>Semantics on GSPNs</th>
<th>Semantics on Monolithic MAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority gates</td>
<td>( \leq )</td>
<td>( &lt; )</td>
<td>( &lt; )</td>
<td>( \leq )</td>
</tr>
<tr>
<td>Nested spares</td>
<td>Not supported</td>
<td>Late claiming</td>
<td>Not supported</td>
<td>Early claiming</td>
</tr>
<tr>
<td>Failure propagation</td>
<td>Bottom-up</td>
<td>Arbitrary</td>
<td>Arbitrary</td>
<td>Bottom-up</td>
</tr>
<tr>
<td>FDEP forwarding</td>
<td>First</td>
<td>Interleaved</td>
<td>Interleaved</td>
<td>Last</td>
</tr>
<tr>
<td>Non-determinism</td>
<td>( \times ) Uniform distr.</td>
<td>( \checkmark ) Everywhere</td>
<td>( \times ) Uniform distr.</td>
<td>( \checkmark ) FDEPs</td>
</tr>
</tbody>
</table>
Motivation
Motivation

Problem: Different semantics

- What do DFTs precisely mean?
- DFT gates are easy to understand in isolation
- But hard to understand in combination
- Each tool has slightly different semantics:
  ➔ different analysis results
Motivation

Problem: Different semantics

- What do DFTs precisely mean?
- DFT gates are easy to understand in isolation
- But hard to understand in combination
- Each tool has slightly different semantics:
  → different analysis results

General framework needed

- use Generalized Stochastic Petri Nets (GSPNs)
- Priorities and partitioning define semantics
Outline

DFT example  GSPNs  Templates  Semantic issues  Properties
Outline

Generalised Stochastic Petri Nets
Definition 1 (GSPN). A generalised stochastic Petri net (GSPN) $G$ is a tuple $(P, T, I, O, H, m_0, W, \Pi Dom, \Pi, \mathcal{D})$ where
- $P$ is a finite set of places.
- $T = T_i \cup T_t$ is a finite set of transitions, partitioned into the set $T_i$ of immediate transitions and the set $T_t$ of timed transitions.
- $I, O, H : T \rightarrow (P \rightarrow \mathbb{N})$, the input-, output- and inhibition-multiplicities of each transition, respectively.
- $m_0 \in M$ is the initial marking with $M = P \rightarrow \mathbb{N}$ the set of markings.
- $W : T \rightarrow \mathbb{R}_{> 0}$ are the transition-weights.
- $\Pi Dom$ is the priority domain and $\Pi : T \rightarrow \Pi Dom$ the transition-priorities.
- $\mathcal{D} \in 2^{T_i}$, a partition of the immediate transitions.
Generalised Stochastic Petri Net (GSPN)

GSPN:

\[
\begin{align*}
\lambda & \quad \mu \\
\end{align*}
\]
Generalised Stochastic Petri Net (GSPN)

GSPN:
• places
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions

\[ \lambda \quad \mu \]
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs

\[ \lambda \quad \mu \]
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs
- inhibitor arcs

\[ \lambda \quad \mu \]
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs
- inhibitor arcs
- initial marking
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs
- inhibitor arcs
- initial marking
- transition-weights

\[ \lambda \quad w: 0.5 \quad w: 0.5 \quad \mu \]
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs
- inhibitor arcs
- initial marking
- transition-weights
- transition-priorities
Generalised Stochastic Petri Net (GSPN)

GSPN:
- places
- immediate transitions
- timed transitions
- input arcs
- output arcs
- inhibitor arcs
- initial marking
- transition-weights
- transition-priorities
- partitioning of immediate transitions
GSPN: Firing

\[ \lambda \quad \mu \]
GSPN: Firing
GSPN: Firing

\[ \lambda \quad \mu \]
GSPN: Firing
GSPN: Firing
GSPN: Firing
Determine which transitions can fire:

1. Enabled transitions
2. maximal priority
3. non-deterministically in different partitions
4. Probabilistically by transition weights (in same partition)
GSPN: Firing order

Determine which transitions can fire:

1. Enabled transitions

2. maximal priority

3. non-deterministically in different partitions

4. Probabilistically by transition weights (in same partition)
Determine which transitions can fire:

1. Enabled transitions

2. maximal priority

3. non-deterministically in different partitions

4. Probabilistically by transition weights (in same partition)
GSPN: Firing order

Determine which transitions can fire:

1. Enabled transitions
2. maximal priority
3. non-deterministically in different partitions
4. Probabilistically by transition weights (in same partition)
Markov Automata semantics for GSPNs

Non-deterministic choice in Markov Automaton
Markov Automata semantics for GSPNs

Non-deterministic choice in Markov Automaton

Randomization in Markov Automaton

[ Eisenbraut et al., Petri Nets 2013 ]
Outline

- DFT example
- GSPNs
- Templates
- Semantic issues
- Properties
Outline

GSPN templates for DFT elements
General translation
General translation
General translation
General translation

Interface places: connecting templates

Gate specific templates
General translation

\[ X \leq Z \]

A \quad B

\[ \text{Failed} \quad Z \quad \text{Failed} \]

\[ \text{Failed} \quad X \quad \text{Failed} \]

\[ \text{Failed} \quad A \quad \text{Failed} \quad B \]

\[ \text{FailSafe} \quad Z \quad \text{Failed} \quad Z \]

\[ \text{Failed} \quad X \quad \text{Failed} \]

\[ \text{Failed} \quad A \quad \text{Failed} \quad B \]
Template for BE

Failed

Unavailable

fail-active

fail-passive

Active

Disabled

\( \lambda \)

\( \mu \)
Template for AND

\[ \text{Failed} \quad \text{Unavailable} \]

\[ C_1 \quad \ldots \quad C_n \]

\[ \text{Failed} \quad \text{Failed} \]

\[ C_1 \quad C_n \]
Template for OR

\[ C_1 \quad \cdots \quad C_n \]

\[ \begin{array}{c}
\text{Failed} \\
\cdots \\
\text{Failed}
\end{array} \quad \begin{array}{c}
\text{Unavailable} \\
\cdots \\
\text{Failed}
\end{array} \]
Template for inclusive PAND

\[ \leq \]

\[ C_1 \quad \cdots \quad C_n \]
Template for FDEP

- $c_1$
- $c_2$
- $c_n$

- Failed $c_1$
- Failed $c_2$
- Unavailable $c_n$
- Failed $c_2$
- Unavailable $c_n$
Template for SPARE

Next $c_1$ 

Failed $c_1$ 

Unavail. $c_1$ 

Claim 

Child-fail 

Next $c_2$ 

Failed $c_2$ 

Unavail. $c_2$ 

Claim 

Child-fail 

Unavail. 

Failed
Template for SPARE
Template for SPARE

Unavail. c1

Next

c1

Failed

c1

Claim

Unavail. c1

Failed

c2

Claim

Unavail. c2

Failed

c2

Unavail. c2

Next

c2

Unavail. c1

Next

c1
Template for SPARE
Template for SPARE
Template for SPARE

\[ c_1 \quad c_2 \]

\[ \text{Next} \quad c_1 \quad \text{unavail.} \quad \text{Next} \quad c_2 \quad \text{unavail.} \quad \text{Failed} \]

\[ \text{claim} \quad c_1 \quad \text{Unavail.} \quad \text{claim} \quad c_2 \quad \text{Unavail.} \]

\[ \text{child-fail} \quad \text{Failed} \quad c_1 \quad \text{Failed} \quad c_2 \]

\[ \text{Unavail.} \quad \text{Unavail.} \]
Template for SPARE

![Diagram of a Petri net with transitions and places labeled as follows:
- Places: c1, c2, Unavail, Failed
- Transitions: Next c1, Next c2, claim, child-fail
- Arrows show the flow from c1 to Unavail, c2 to Unavail, c1 to Failed, c2 to Failed, and vice versa.]

Matthias Volk - One Net Fits All
UnRAVeL bi-weekly meeting
September 5, 2018
Template for SPARE

Next $c_1$

$\text{Unavail.}$

Next $c_2$

$\text{Unavail.}$

Failed $c_1$

Failed $c_2$

$\text{claim}$

$\text{claim}$

$\text{child-fail}$

$\text{child-fail}$

$\text{Unavail.}$

$\text{Unavail.}$
Outline

- DFT example
- GSPNs
- Templates
- Semantic issues
- Properties
Outline

Semantic issues
Semantic issue 1: Failure propagation

 FailSafe
 Z
 Failed
 Z
 Failed
 X
 Failed
 Z
 Failed
 A
 Failed
 B
 Failed
 A
 Failed
 B
Semantic issue 1: Failure propagation

![Diagram of failure propagation](image-url)
Semantic issue 1: Failure propagation

\begin{itemize}
  \item Failed A
  \item Failed B
  \item Failed X
  \item Failed Z
  \item FailSafe Z
\end{itemize}
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation

![Diagram showing failure propagation](image)
Semantic issue 1: Failure propagation

\[ A \leq Z \]

\[ X \to A \]

\[ Z \to B \]

\[ \text{FailSafe} \]

\[ \text{Failed} \]

\[ \text{Failed} \]

\[ \text{Failed} \]

\[ \text{Failed} \]

\[ \text{Failed} \]

\[ \text{Failed} \]
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation

A → Z ≤ B

FailSafe

Z

Failed

Z

X

Failed

X

Failed

A

Failed

B

Research Training Group – Uncertainty and Randomness in Algorithms, Verification, and Logic
Semantic issue 1: Failure propagation

A ≤ Z

Failed A

FailSafe Z

Failed B

Failed X

Failed Z

Failed
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation

[Diagram of a network with nodes labeled A, B, X, and Z, showing failure propagation through FailSafe and other connections.]

A -> B
X <- Z

Failed X
Failed A
Failed B
Failed Z

FailSafe Z
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation
Semantic issue 1: Failure propagation

Propagation order can decide between failure and fail-safe
Semantic issue 1: Failure propagation

Set priorities to define failure propagation

A

B

Z

X

Failed

Failed

Failed

FailSafe

Z

X

A

B

Z

Failed

Failed

Failed
Semantic issue 1: Failure propagation

Set priorities to define failure propagation
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding
Semantic issue 2: FDEP forwarding

Again priorities define FDEP forwarding

\[ \leq \]
Semantic issue 2: FDEP forwarding

Again priorities define FDEP forwarding
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism

Output of GreatSPN Editor
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism

Spare race: which gate claims the shared element?
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism
Semantic issue 3: Non-determinism

Spare race can decide between failure and fail-safe

D ——> S ——> Z ——> T

X

A

B

C

≤
Semantic issue 3: Non-determinism

Spare race is conflict in GSPN

Diagram:

- D
- S
- T
- X
- A
- B
- C
- Z

≤
Semantic issue 3: Non-determinism

Spare race is conflict in GSPN
Semantic issue 3: Non-determinism

Set transition partitioning to define how to solve conflict:
- non-determinism
- randomisation
1. Failure propagation
2. FDEP forwarding
3. Non-determinism

All semantics

Failure propagation?

Arbitrary

Bottom-up

FDEP forwarding?

Interleaved with gates

IOIMC, Orig. GSPN

FDEP forwarding?

Monolithic CTMC, Orig. GSPN

Before gates

IOIMC

Non-determinism?

Yes

IOIMC

No

Orig. GSPN

Monolithic MA

After gates

Non-determinism?

Yes

Monolithic MA

No

New GSPN

Monolithic CTMC
# DFT Semantics

<table>
<thead>
<tr>
<th></th>
<th>Semantics on Monolithic CTMCs</th>
<th>Semantics on I/O-IMCs</th>
<th>Semantics on GSPNs</th>
<th>Semantics on Monolithic MAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure propagation</td>
<td>Bottom-up</td>
<td>Arbitrary</td>
<td>Arbitrary</td>
<td>Bottom-up</td>
</tr>
<tr>
<td>FDEP forwarding</td>
<td>First</td>
<td>Interleaved</td>
<td>Interleaved</td>
<td>Last</td>
</tr>
<tr>
<td>Priorities</td>
<td>$\pi(v) &lt; \pi(v_i)$</td>
<td>$\pi(v) = \pi(v')$</td>
<td>$\pi(v) = \pi(v')$</td>
<td>$\pi(v) &lt; \pi(v_i)$</td>
</tr>
<tr>
<td></td>
<td>$\pi(f) &gt; \pi(v)$</td>
<td></td>
<td></td>
<td>$\pi(f) &lt; \pi(v)$</td>
</tr>
<tr>
<td>Non-determinism</td>
<td>$\times$</td>
<td>$\checkmark$</td>
<td>$\times$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>Partitioning</td>
<td>${T_i}$</td>
<td>${{t} \mid t \in T_i}$</td>
<td>${T_i}$</td>
<td>${{t} \mid t \in T_i}$</td>
</tr>
</tbody>
</table>
Outline

DFT example

GSPNs

Templates

Semantic issues

Properties
Properties of resulting GSPNs

• Compositional

• Size of GSPN is linear in size of DFT

• Each transition fires at most once  ➔ no time-traps

• GSPN is 2-bounded
  • all places except Unavailable are 1-bounded
Properties of resulting GSPNs

• Compositional

• Size of GSPN is linear in size of DFT

• Each transition fires at most once
  ➜ no time-traps

• GSPN is 2-bounded
  • all places except Unavailable are 1-bounded

• GSPN translation implemented in probabilistic model checker

  Storm

  http://www.stormchecker.org

• Export in common GSPN formats, e.g. PNML or for GreatSPN Editor
Conclusion

- GSPN semantics for Dynamic Fault Trees
- Compositional in gates
- Framework unifies existing DFT semantic variants
- Each DFT semantic variant defined by:
  - transition priorities
  - transition partitioning
Conclusion

• GSPN semantics for Dynamic Fault Trees
• Compositional in gates
• Framework unifies existing DFT semantic variants
• Each DFT semantic variant defined by:
  • transition priorities
  • transition partitioning

Future work
• Analyse resulting GSPNs using common Petri Net tools
• Investigate unfoldings of underlying nets
• Extend framework to incorporate repairs