Structural analysis

FTA and FMEA

Riccardo M.G. Ferrari

April 29th, 2025

TU Delft

TODAY'S CLASS

WHAT ARE WE GOING TO DO TODAY

1. Lecture (approx. 1h)
2. Exercise (approx. 30min)

Lecture

Overview

Refresher

Refresher

Component and Services Model

- **Components** provide **services**
- A **service** s is described by a **6-tuple**:
  $s = \{\text{cons}, \text{prod}, \text{proc}, \text{rqst}, \text{enable}, \text{res}\}$

Riccardo M. G. Ferrari and Alexander J. Gallo. *Structural Analysis – Components and Services Model*. Fault Diagnosis and Fault Tolerant Control – 2023

Refresher

Component and Services Model

- **Components** provide **services**
- A **service** s is described by a **6-tuple**:
  $s = \{\text{cons}, \text{prod}, \text{proc}, \text{rqst}, \text{enable}, \text{res}\}$

$$
\begin{aligned}
\text{cons} &= \{q_i,\ q_o\} \\
\text{prod} &= \{h\} \\
\text{proc} &= \left\{\dot{h} = q_i - q_o,\ h = \int \dot{h}\text{d}t\right\} \\
\text{rqst} &= \{1\} \\
\text{enable} &= \{1\} \\
\text{res} &= \{\text{vessel}, \text{pipes}\}
\end{aligned}
$$

Riccardo M. G. Ferrari and Alexander J. Gallo. *Structural Analysis – Components and Services Model*. Fault Diagnosis and Fault Tolerant Control – 2023

Refresher

Component and Services Model – Why use it?

- For **fault diagnosis**
  - Assuming components can be either **healthy** or **faulty** $\Rightarrow$ **root cause analysis** and **propagation analysis**
  - See next lecture on FTA and FMEA
- For **fault accommodation** via **switching** of hardware redundant components

Riccardo M. G. Ferrari and Alexander J. Gallo. *Structural Analysis – Components and Services Model*. Fault Diagnosis and Fault Tolerant Control – 2023

Refresher

Component and Services Model – Why use it?

- For **fault diagnosis**
  - Assuming components can be either **healthy** or **faulty** $\Rightarrow$ **root cause analysis** and **propagation analysis**
- For **fault accommodation** via **switching** of hardware redundant components

Riccardo M. G. Ferrari and Alexander J. Gallo. *Structural Analysis – Components and Services Model*. Fault Diagnosis and Fault Tolerant Control – 2023

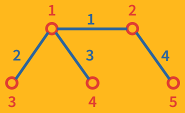
Refresher

Graphs

Definition (Graph)

$$\mathcal{G} \triangleq (\mathcal{N}, \mathcal{E})$$

example:



$\mathcal{N} \triangleq \{n_1, \ldots, n_N\}$ **nodes**

$\mathcal{E} \triangleq \{e_1, \ldots, e_M\}$ **edges**

i.e. $e_3 = (1,\ 4) \in \mathcal{E}$

Edges can be oriented!

i.e. $(1,\ 2) \in \mathcal{E}$ but $(2,\ 1) \notin \mathcal{E}$

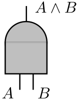
Reinhard Diestel. *Graph Theory*. Springer, 2017

Refresher

Fundamentals of Boolean Algebra

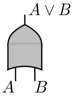
- Boolean variables hold **binary values**: $A \in \{\underbrace{0}_{\text{False}}, \overbrace{1}^{\text{True}}\}$
- Main **operations** we'll use

AND



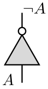
$A \wedge B$

OR



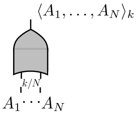
$A \vee B$

NOT



$\neg A$

Majority voting



$\langle A_1, \ldots, A_N \rangle_k$

Refresher

Fundamentals of Boolean Algebra

- **De Morgan's laws**:

$$\neg(A \wedge B) \iff \neg A \vee \neg B$$
$$\neg(A \vee B) \iff \neg A \wedge \neg B$$

- **Disjunctive normal form**: representation of a **semantic** (logic) **function** as the *disjunction* (OR) of several *conjunctions* (ANDs), e.g.:

$$f(A, B, C, D) = (A \wedge B) \vee (C \wedge D) \vee (A \wedge D)$$

Fault Tree Analysis

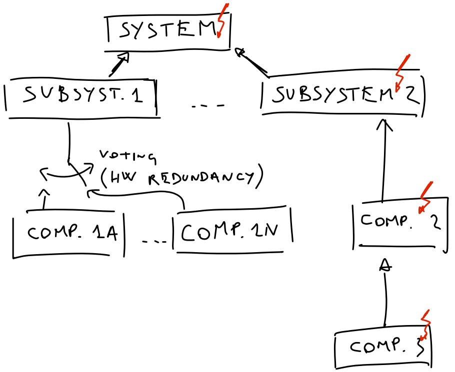
Fault Tree Analysis

Motivation and Overview

- Diagnosis method to analyse **effect** of **component failure** on system dependability
- Answers the question:

FTA – Objective

Can the failure of **a (subset of) component(s)** lead to the failure of the whole system?



Fault Tree Analysis

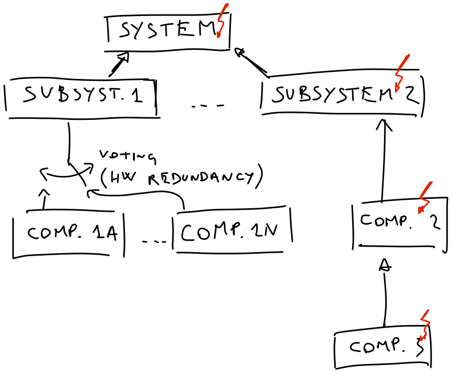
Motivation and Overview

↱Requires knowledge

- **Diagnosis method** to analyse **effect** of **component failure** on system dependability
- Answers the question:

FTA – Objective

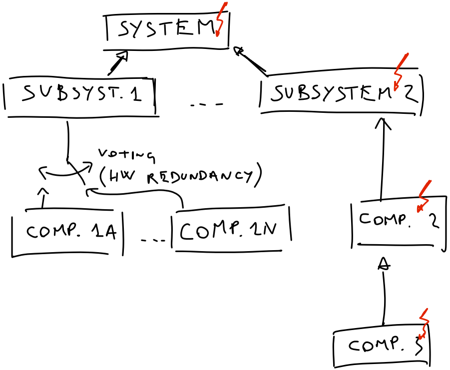
Can the failure of **a (subset of) component(s)** lead to the failure of the whole system?



Fault Tree Analysis

Knowledge base

- **Components** which make up the system
- What **services** are offered by each component
- How components are **interconnected**



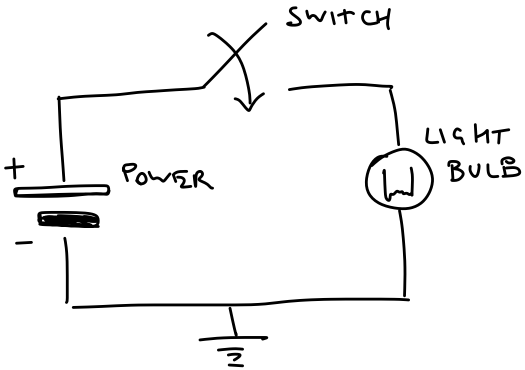
Note

Qualitative knowledge is enough!

Fault Tree Analysis

An Example

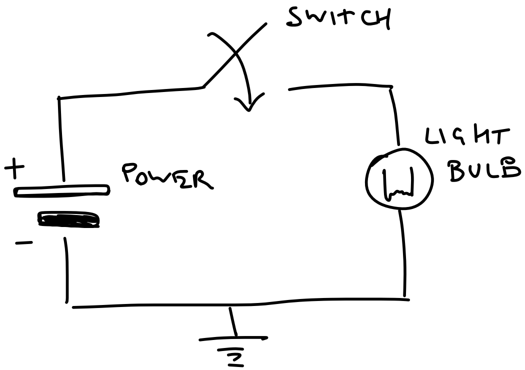
- Overall system representation: a light circuit



Fault Tree Analysis

An Example

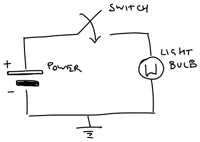
- Overall system representation: a light circuit
- Identification of the components



Fault Tree Analysis

An Example

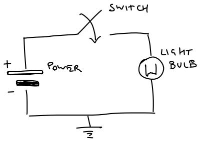
- Overall system representation: a light circuit
- Identification of the components
- A representation of how components compose **subsystems** and the **overall system**



Fault Tree Analysis

An Example

- Overall system representation: a light circuit
- Identification of the components
- A representation of how components compose **subsystems** and the **overall system**

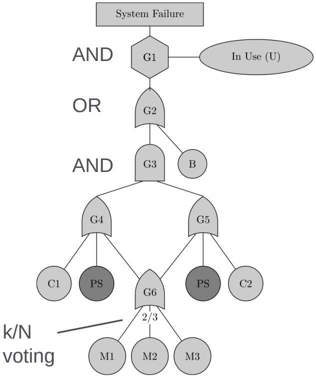


Note:
The propagation can be represented in **positive** or **negative** logic

Fault Tree Analysis

What is a fault tree

- A **directed, acyclic graph**
  - Not always a *tree*
- **Leaves** represent **component failures** (**Basic Events**)
- Component **interconnection** through **Logic Gates**
- **Root node** is the **Top Event**: complete **System Failure**



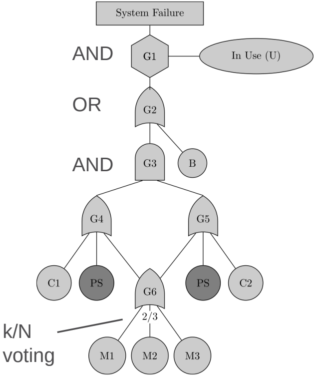
E. Ruijters and M. Stoelinga. "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". In: *Computer science review* (2015)

Fault Tree Analysis

What is a fault tree

Negative logic

Fault trees represent a **propagation of failure**: i.e., when component failure leads to system failure



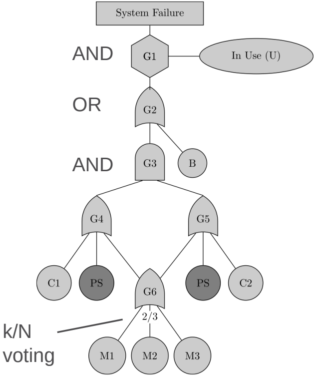
E. Ruijters and M. Stoelinga. “Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools”. In: *Computer science review* (2015)

Fault Tree Analysis

Formal description

Definition (Fault tree)

$$F = \langle \mathcal{E}_B, \mathcal{G}, T, I \rangle$$



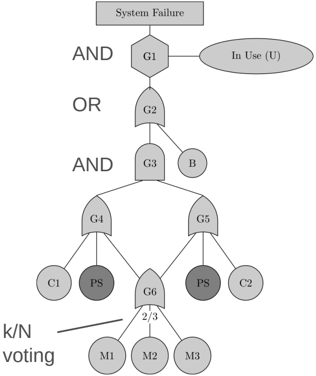
Fault Tree Analysis

Formal description

Definition (Fault tree)

$$F = \langle \mathcal{E}_B, \mathcal{G}, T, I \rangle$$

- $\mathcal{E}_B$ is the set of **basic events** $e \in \mathcal{E}_B$, $e \in \{0, 1\}$ (0 → Healthy, 1 → Faulty)



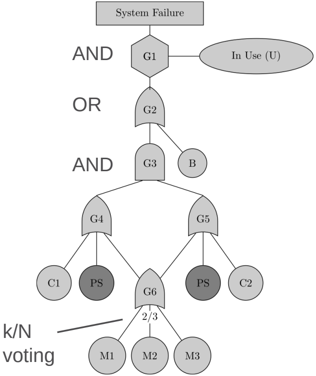
Fault Tree Analysis

Formal description

Definition (Fault tree)

$$F = \langle \mathcal{E}_B, \mathcal{G}, T, I \rangle$$

- $\mathcal{E}_B$ is the set of **basic events** $e \in \mathcal{E}_B$, $e \in \{0, 1\}$ (0: Healthy, 1: Faulty)
- $\mathcal{G}$ is the set of **gates**



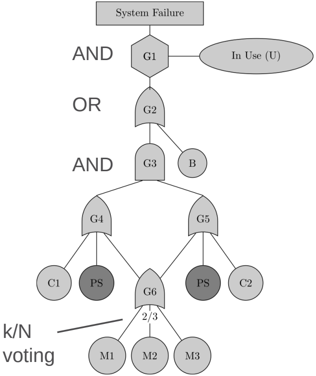
Fault Tree Analysis

Formal description

Definition (Fault tree)

$$F = \langle \mathcal{E}_B, \mathcal{G}, T, I \rangle$$

- $\mathcal{E}_B$ is the set of **basic events** $e \in \mathcal{E}_B$, $e \in \{0, 1\}$ (Healthy: 0, Faulty: 1)
- $\mathcal{G}$ is the set of **gates**
- $T : \mathcal{G} \to \{\wedge, \vee, \neg, \langle e_{i_1}, \dots, e_{i_N} \rangle_k\}$ maps gates to gate **types**



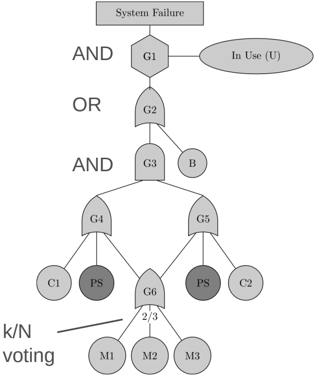
Fault Tree Analysis

Formal description

Definition (Fault tree)

$$F = \langle \mathcal{E}_B, \mathcal{G}, T, I \rangle$$

- $\mathcal{E}_B$ is the set of **basic events** $e \in \mathcal{E}_B$, $e \in \{0, 1\}$ (1: Faulty; 0: Healthy)
- $\mathcal{G}$ is the set of **gates**
- $T : \mathcal{G} \to \{\wedge, \vee, \neg, \langle e_{i_1}, \ldots, e_{i_N} \rangle_k\}$ maps gates to gate **types**
- $I : \mathcal{G} \to \mathscr{P}(\mathcal{E})$ maps gates to their **inputs**
  ↳ $\mathscr{P}(\mathcal{X})$ is the **power set** of $\mathcal{X}$
  - $\mathcal{E} = \mathcal{E}_B \cup \mathcal{G}$



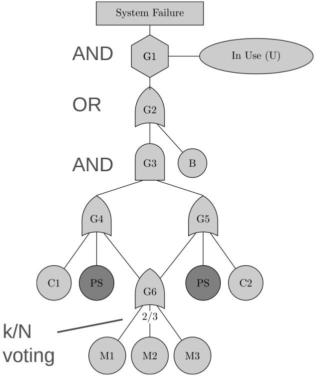
FAULT TREE ANALYSIS

FORMAL DESCRIPTION

Each fault tree has a **semantic** (logic) **function** associated to it

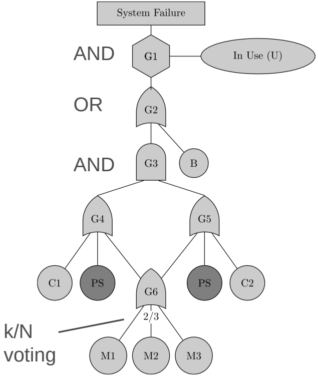
$$\pi_F : \mathscr{P}(\mathcal{E}_B) \times \mathcal{E} \to \{0, 1\}$$

- $\pi_F(\mathcal{S}, e_i) = 1$, if $e_i \in \mathcal{E}$ is a **failure** ($e_i = 1$) when all elements of $\mathcal{S} \subset \mathscr{P}(\mathcal{E}_B)$ are **failures**
- **Shorthand** $\pi_F(S)$ if e_i is the **top event**
  - $\pi_F(S)$ represents effect of component failure on system health



Fault Tree Analysis

Cut sets and minimal cut sets



Definition

A set $\mathcal{C} \subseteq \mathcal{E}_B$ is a **cut set** of F if

$$\pi_F(\mathcal{C}) = 1$$

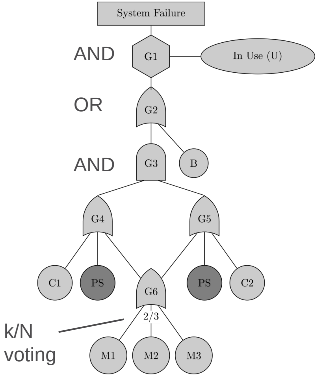
Definition

A set $\mathcal{C} \subseteq \mathcal{E}_B$ is a **minimal** cut set of F if it is a cut set of which no subset is a cut set, i.e.,

$$\pi_F(\mathcal{C}) = 1 \wedge \pi_F(\mathcal{C}') = 0, \forall \mathcal{C}' \,|\, \mathcal{C}' \subset \mathcal{C}$$

Fault Tree Analysis

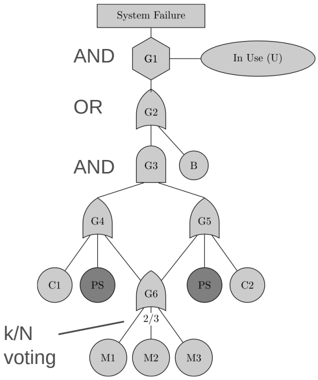
Cut sets and minimal cut sets



Some examples?

Fault Tree Analysis

Structure function



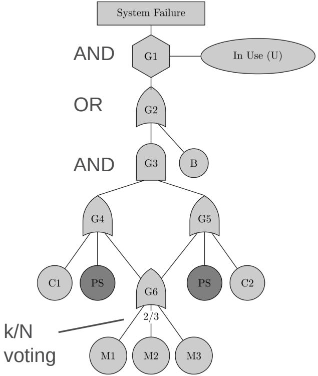
A method for **quantitative** analysis: the fault tree **structure function** ($N = |\mathcal{E}_B|$):

$$f : \{0,1\}^N \rightarrow \{0,1\}$$

- $f(e_1, \dots, e_N) = 1$ when the values of $e_i \in \{0,1\}, i \in \{1, \dots, N\}$ results in a **failure** at top event (i.e., system-wide failure)
- $f(\cdot)$ can be expressed in **disjoint normal form**
  - Every conjunction in f is a minimal cut set!

Fault Tree Analysis

Structure function



Build structure function and find all MCS

Fault Tree Analysis

Topics which are not covered

- Probabilistic fault trees
  - Propagating probability of failure
  - Allows for analysis of, e.g., **mean-time to failure** or **reliability**
- Dynamic fault trees
- Repairable fault trees
- Common cause of failure
- ...

E. Ruijters and M. Stoelinga. "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". In: *Computer science review* (2015)

Fault Mode and Effect Analysis

Fault Mode and Effect Analysis

Motivation and Overview

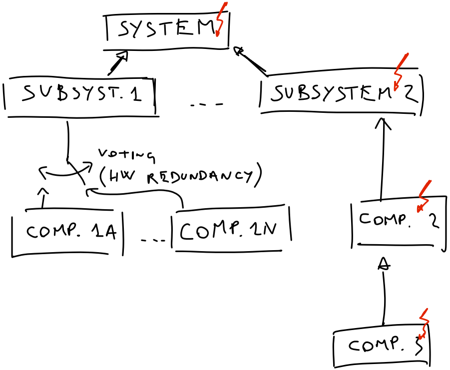
- Analyse **effect** of **component failure** on system dependability

FMEA – Objective

How does the failure of **a (subset of) component(s)** lead to the failure of the whole system?

FMEA vs FTA

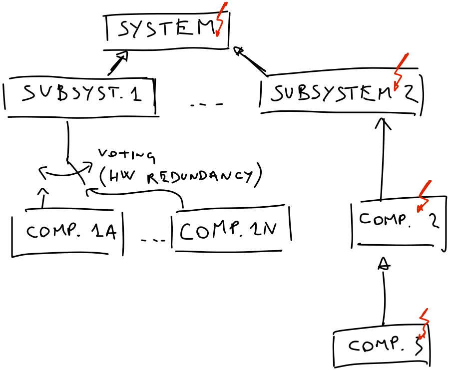
The main difference with FTA lies in the question of **how**



Fault Mode and Effect Analysis

Knowledge base

- **Components** which make up the system, and their **services**
- **Fault/failure** modes of each component
- **Effect** of each mode at component level
- How the mode of one component **affects** connected components



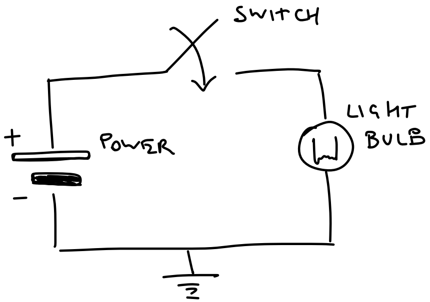
Note

Qualitative knowledge is (again) enough!

Fault Mode and Effect Analysis

Switch Example

- Overall system representation: a light circuit with its components

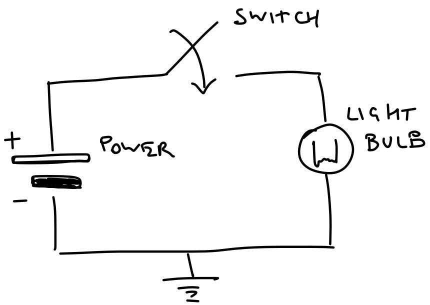


Fault Mode and Effect Analysis

Switch Example

- Overall system representation: a light circuit with its components

1. **Drained battery** fault mode and its effects

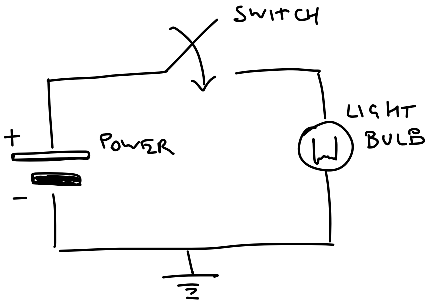


Fault Mode and Effect Analysis

Switch Example

- Overall system representation: a light circuit with its components

1. **Drained battery** fault mode and its effects
2. **Stuck switch** fault mode and its effects

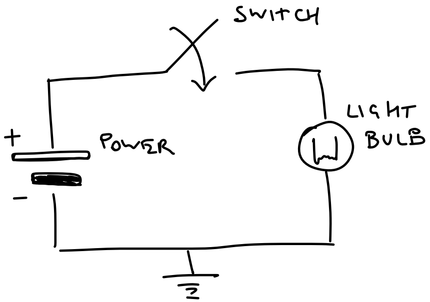


Fault Mode and Effect Analysis

Switch Example

- Overall system representation: a light circuit with its components

1. **Drained battery** fault mode and its effects
2. **Stuck switch** fault mode and its effects
3. **Broken bulb** fault mode and its effects



Fault Mode and Effect Analysis

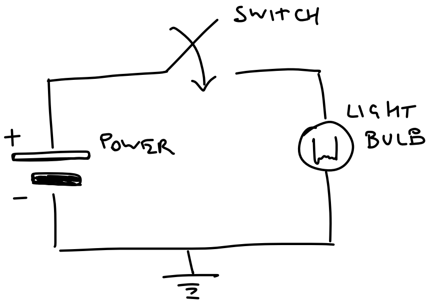
Switch Example

- Overall system representation: a light circuit with its components

1. **Drained battery** fault mode and its effects
2. **Stuck switch** fault mode and its effects
3. **Broken bulb** fault mode and its effects

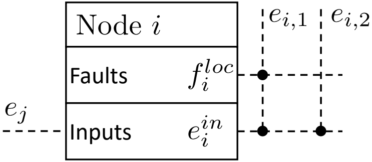
Failure propagation

Propagation occurs through effects



Fault Mode and Effect Analysis

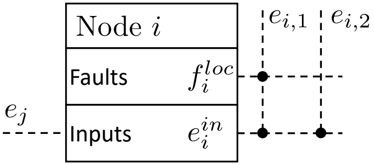
Graph representation



- **Node** associated with each component
  - **Local faults** f_i^{loc}
  - **Local effects** e_i
  - **Inherited effects** e_i^{in}

Fault Mode and Effect Analysis

Graph representation



- **Node** associated with each component
  - **Local faults** f_i^{loc}
  - **Local effects** e_i
  - **Inherited effects** e_i^{in}

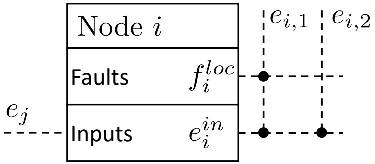
Inherited effects

Inherited effects are local effects of **other nodes**:

$$e_i^{in} = e_j$$

Fault Mode and Effect Analysis

Graph representation



- **Node** associated with each component
  - **Local faults** f_i^{loc}
  - **Local effects** e_i
  - **Inherited effects** e_i^{in}

How to relate faults and effects?

Boolean operators can help with formalization:

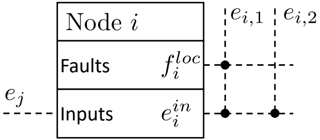
$$e_i = M_i \otimes f_i$$

Fault Mode and Effect Analysis

Boolean representation

Fault propagation can be represented as a **boolean mapping** between f and e:
↳Fault ↳Effect

$$e = M \otimes f$$



Definition (Boolean operation $\otimes$)

Given a **boolean matrix** $M \in \{0,1\}^{m \times n}$, $e = M \otimes f$ represents the following disjoint normal form:

$$e_{(i)} = (M_{(i,1)} \wedge f_{(1)}) \vee \cdots \vee (M_{(i,n)} \wedge f_{(n)})$$

Fault Mode and Effect Analysis

Boolean representation

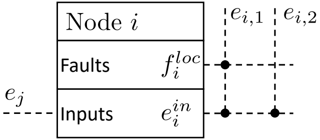
Example

For scalar f_i^{loc} and e_i^{in}:

$$f_i = \begin{bmatrix} f_i^{loc} \\ e_i^{in} \end{bmatrix} \quad e_i = \begin{bmatrix} e_{i,1} \\ e_{i,2} \end{bmatrix}$$

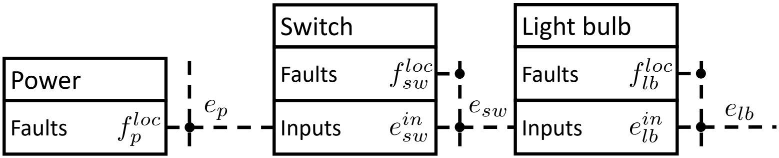
we obtain **boolean matrix**:

$$M_i = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$



Fault Mode and Effect Analysis

Light switch example: revisited

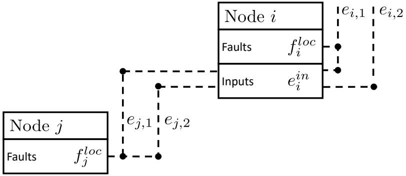


Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

How do we **combine** multiple boolean matrices

$$e_i = M_i \otimes \begin{bmatrix} f_i^{loc} \\ e_i^{in} \end{bmatrix}$$

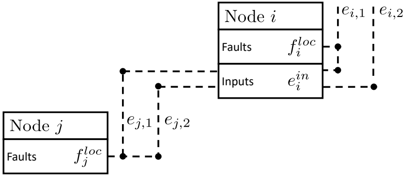


Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

How do we **combine** multiple boolean matrices

$$e_i = M_i \otimes \begin{bmatrix} f_i^{loc} \\ e_j \end{bmatrix}$$

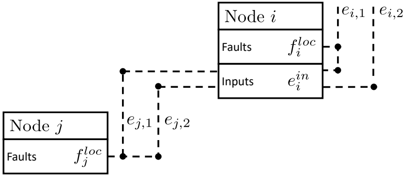


Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

How do we **combine** multiple boolean matrices

$$
\begin{aligned}
e_i &= M_i \otimes \begin{bmatrix} f_i^{loc} \\ e_j \end{bmatrix} \\
&= M_i \otimes \begin{bmatrix} f_i^{loc} \\ M_j \otimes f_j \end{bmatrix}
\end{aligned}
$$

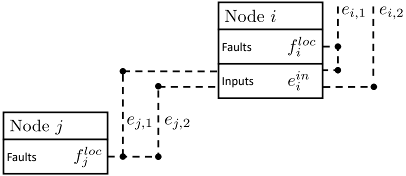


Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

How do we **combine** multiple boolean matrices

$$
\begin{aligned}
e_i &= M_i \otimes \begin{bmatrix} f_i^{loc} \\ e_j \end{bmatrix} \\
&= \underbrace{\left(M_i \otimes \begin{bmatrix} I & 0 \\ 0 & M_j \end{bmatrix} \right)}_{M_{ij}} \otimes \begin{bmatrix} f_i^{loc} \\ f_j \end{bmatrix}
\end{aligned}
$$



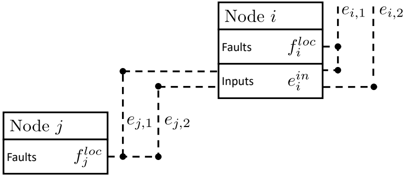
Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

How do we **combine** multiple boolean matrices

$$e_i = M_i \otimes \begin{bmatrix} f_i^{loc} \\ e_j \end{bmatrix}$$

$$= \underbrace{\left(M_i \otimes \begin{bmatrix} I & 0 \\ 0 & M_j \end{bmatrix} \right)}_{M_{ij}} \otimes \begin{bmatrix} f_i^{loc} \\ f_j \end{bmatrix}$$



Validity of operation

This composition only holds if there are **no loops**

Fault Mode and Effect Analysis

Hierarchical composition of boolean matrices

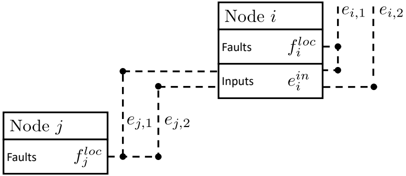
Example

Scalar f_i^{loc}, f_j^{loc}, and $e_i^{in} \in \{0,1\}^2$:

$$M_i = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad M_j = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

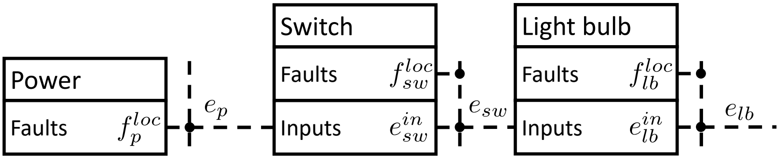
we obtain the combined **boolean matrix**:

$$M_{ij} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$



Fault Mode and Effect Analysis

Light switch example: hierarchical composition



Fault Mode and Effect Analysis

Inverse inference

Previous boolean mapping puts **faults** (causes) in relation to **effects**:

- Which faults may cause the observed effects?
- **Inverse inference problem**
  - Related to the problem of **isolation**

Note

There may be **more than one fault** that can explain the observed effects: mapping is surjective, not bijective

Fault Mode and Effect Analysis

Inverse inference

Inverse inference can be achieved by using the **same boolean matrix** M, with a different boolean operator

Definition (Boolean operator $\odot$)

Given a **boolean matrix** $M \in \{0,1\}^{m \times n}$, $f = M^\top \odot e$ represents the following logical function:

$$f_i = (M_{1,i} == e_1) \wedge \cdots \wedge (M_{m,i} == e_m)$$

Fault Mode and Effect Analysis

Inverse inference

Inverse inference can be achieved by using the **same boolean matrix** M, with a different boolean operator

Definition (Boolean operator $\odot$)

Given a **boolean matrix** $M \in \{0,1\}^{m \times n}$, $f = M^\top \odot e$ represents the following logical function:

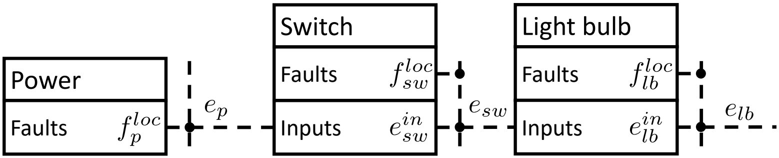
$$f_i = (M_{1,i} == e_1) \wedge \cdots \wedge (M_{m,i} == e_m)$$

Careful

Note the **transpose** applied to the boolean matrix

Fault Mode and Effect Analysis

Light switch example: inverse inference



Conclusion

Conclusion

In this lecture we covered

- **Fault Tree Analysis**
- **Fault Modes and Effects Analysis**

Next lecture: **Change detection algorithms**

Conclusion

Thank you for your attention!

For further information:

Course page on **Brightspace**

or

our **MS Team**

Exercises

Exercises

Fault Tree Analysis

Brief

Consider an automated logistics transport system in a port is composed of:

- 4 Automated Guided Vehicles (AGVs); each AGV is equipped with
  - two redundant localization sensors $LS_{i,1}$ and $LS_{i,2}$
  - a local control unit LCU_i
  - the vehicle body and drivetrain AGV_i
- one centralized control unit CCU, which coordinates the AGVs. If it is not operating correctly, the system may still perform, so long as all AGVs are operational. If not, at least 3 AGVs must be operational for the system to operate correctly
- Two motion capture cameras MCC_i, $i \in \{1, 2\}$, of which one may be malfunctioning.

Exercises

Fault Tree Analysis

Exercise 1

Draw the **fault tree** of the system described (use the labels given to define the basic events)

Exercise 2

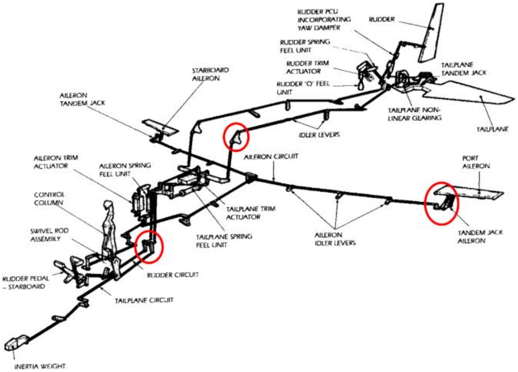
Write the **structure function** of the given fault tree

Exercise 3

List all the **minimal cut sets**

Exercises

Fault Modes and Effects Analysis



Exercises

Fault Modes and Effects Analysis

Component	Fault (F) / propagated effect (P)	Effect
Control column	1. (f_{cc_1}) Stuck column 2. (f_{cc_2}) Wrong calibration	1. (e_{cc_1}) Pilot command does not reach the controller 2. (e_{cc_2}) Incorrect pilot command to controller
Motion sensors	1. (f_{m_1}) Biased sensor measurement	1. (e_{m_1}) Wrong information regarding airplane movement
Air-flow sensors	1. (f_{af_1}) Biased sensor measurement 2. (f_{af_2}) No sensor measurement	1. (e_{af1}) Biased information transmitted to controller 2. (e_{af_2}) No information given to controller

Exercises

Fault Modes and Effects Analysis

Component	Fault (F) / propagated effect (P)	Effect
Controller	1. (f_{c_1}) Microcontroller short-circuited 2. (f_{c_2}) Error in firmware update 3. (p_{c_1}) Pilot command does not reach the controller 4. (p_{c_2}) Incorrect pilot command to controller 5. (p_{c_3}) Wrong information about airplane movement 6. (p_{c_4}) Biased information transmitted to controller 7. (p_{c_5}) No information given to controller	1. (e_{c_1}) No signal to aileron 2. (e_{c_2}) Wrong signal to aileron 3. (e_{c_1}) No signal to aileron 4. (e_{c_2}) Wrong signal to aileron 5. (e_{c_2}) Wrong signal to aileron 6. (e_{c_2}) Wrong signal to aileron 7. (e_{c_1}) No signal to aileron
Electrical cables	1. (f_{e_1}) Short circuit 2. (p_{e_1}) No signal to aileron 3. (p_{e_2}) Wrong signal to aileron	1. (e_{e_1}) No signal to aileron 2. (e_{e_1}) No signal to aileron 3. (e_{e_2}) Wrong signal to aileron
Aileron	1. (f_{a_1}) Hydraulic leak 2. (f_{a_2}) Stuck piston 3. (p_{a_1}) No signal to aileron 4. (p_{a_2}) Wrong signal to aileron	1. (e_{a_1}) Wrong movement 2. (e_{a_2}) No movement 3. (e_{a_2}) No movement 4. (e_{a_1}) Wrong movement

Exercises

Fault Modes and Effects Analysis

Exercise 1

Draw the **fault modes and effects** graph of the system described

Exercise 2

Write the **local boolean matrices** M_i (make sure to include the definition of f_i and e_i)

Exercise 3

Write the **full boolean matrix** M resulting from the hierarchical composition of the components (make sure to include definition of f)

Exercise 4

Write inverse inference relation linking **effects** to **faults**