An Intuitive Method for Reliability Analysis of Dynamic Systems

Seung Ki Shin, Gyoung Tae Goh and Poong Hyun Seong
*Korea Advanced Institute of Science and Technology (KAIST)*

2008 PSAM9
May 23rd, 2008
Contents

- Introduction
- Reliability Graph with General Gates
- Adding Dynamic Nodes to RGGG
- Probability Tables for Dynamic Nodes
  - Discrete-time method
- Development of a Tool for Generating PT
- Summary & Further Study
- References
Introduction

- Among the methods to analyze system reliability, fault tree analysis is the most widely used method, but it’s not an intuitive method.

- Reliability graph with general gates (RGGG) method was proposed in order to overcome this shortcoming of fault tree.

- Conventional fault tree and RGGG method cannot capture the dynamic behavior of system associated with time dependent events.

- Dynamic fault tree method was proposed, but it’s also not intuitive. And when calculating the reliability from dynamic fault tree, usually Markov chain method is used, but it has the state space explosion problem.

- If it is possible to add a sequential concept to RGGG, it will become an intuitive dynamic method.
Reliability Graph with General Graph

- Reliability graph is an intuitive method, so it can model systems by one-to-one match graph.
- The reason why it’s not used widely is low expression power. (Only OR gate)
- RGGG which utilizes general gates was proposed.
- By determining the probability table for each node, RGGG can be transformed to an equivalent Bayesian network and the system reliability can be calculated.
Reliability Graph with General Graph

- **RGGG nodes**

- (a) OR  (b) AND  (c) k-out-of-n  (d) general purpose gate
Reliability Graph with General Graph

- Transformation to reliability graph with perfect node

\[ \Pr\{n_j \text{ as successful}\} = Pr_j \]
\[ \Pr\{a_j \text{ as successful}\} = Pa_j \]

Ex.) Probability table for a node with OR gate when \( n = 2 \)

<table>
<thead>
<tr>
<th>( y_1 )</th>
<th>( y_2 )</th>
<th>( y_A )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1 - ( r_{1A} )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1 - ( r_{2A} )</td>
</tr>
</tbody>
</table>

\[ y_1 = 1 \] (success) \[ y_2 = 1 \] (success) \[ y_A = 1 \] (success)
\[ y_1 = 0 \] (failure) \[ y_2 = 0 \] (failure) \[ y_A = 0 \] (failure)

\[ r_{1A} + r_{2A} - r_{1A}r_{2A} \]
\[ r_{1A} \]
\[ r_{2A} \]
\[ 0 \]
Reliability Graph with General Graph

- Comparison of fault tree and RGGG
Reliability Graph with General Graph

Ex. 1) Digital Plant Protection System

Fault Tree (complex)

Equivalent !!!

Structure of DPPS
(For low pressurizer pressure trip)

RGGG
(more intuitive)

Only 1 page !!!
Reliability Graph with General Graph

Ex. 2) Recirculation in KSNPs

Fault Tree (complex)

Calculation result: $2.786 \times 10^{-3}$

High Pressure Recirculation (in KSNP-type NPPs)

RGGG (more intuitive)

Calculation result: $2.7746 \times 10^{-3}$

Parts related to the actual system structure
Adding Dynamic Nodes to RGGG

- The existing reliability graph with general gate (RGGG) method cannot capture dynamic behaviors of a system associated with sequence dependent failure events.

- We add dynamic nodes to existing RGGG and conduct quantitative analysis by using discrete-time method.  
  → Dynamic RGGG
Adding Dynamic Nodes to RGGG

- **Priority-AND (PAND) node**
  - Node fails only if both signals from A and B are disconnected and a signal from A is disconnected before B.
Adding Dynamic Nodes to RGGG

- **Spare node**
  - Node fails only if a primary signal and all spare signals are disconnected.
Adding Dynamic Nodes to RGGG

- SEQ node
  - The input signals are constrained to be disconnected in the particular order and SEQ node fails if and only if all input signals are disconnected.
Adding Dynamic Nodes to RGGG

- FDEP node
  - This node is not necessary in RGGG.
  - Existing RGGG can illustrate the property of FDEP gate.
In order to transform a dynamic reliability graph to an equivalent Bayesian network, the probability table corresponding to each dynamic node should be determined.

- The discrete-time method is used.

\[ P_{ij}^k : \text{The probability that an arc } a_{ij} \text{ from node } n_i \text{ to } n_j \text{ is failed in the } k^{th} \text{ time interval} \]
- When the cumulative failure distribution function of } a_{ij} \text{ is } F_{ij}(t) ,

\[ P_{ij}^k = \int_{(k-1)t}^{kt} \frac{dF_{ij}(t)}{dt} \, dt \]

Divide the process time line into \( n \) same intervals.

- Output of each node is one of \( \{I_1, I_2, \ldots, I_n, \infty\} \).
  - \( I_k \) means that the node is failed in \( k^{th} \) interval.
  - \( I_\infty \) means that the node is never failed.
Probability Tables for Dynamic Nodes

- In order to obtain the accurate reliability, \( n \) (the number of time discretization) should increases, but as \( n \) increases, the probability table of each node becomes more complex.
  - The number of blanks that should be filled is \((n+1)^3\) per each node. (2 inputs)
  
  Ex) \( n = 3 \),

<table>
<thead>
<tr>
<th></th>
<th>( I_1 )</th>
<th>( I_2 )</th>
<th>( I_3 )</th>
<th>( I_\infty )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( I_1 )</td>
<td>( I_2 )</td>
<td>( I_3 )</td>
<td>( I_\infty )</td>
</tr>
<tr>
<td></td>
<td>( I_1 )</td>
<td>( I_2 )</td>
<td>( I_3 )</td>
<td>( I_\infty )</td>
</tr>
<tr>
<td></td>
<td>( I_2 )</td>
<td>( I_\infty )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( I_3 )</td>
<td>( I_\infty )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( I_\infty )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

64 blanks should be filled.

- The rule of making probability tables is necessary to reduce making time and mistakes.
Let output of \( n_1, n_2, n_A \) as \( I_x, I_y, I_z \)

- \( I_x \) means that \( n_1 \) fails at \( x^{th} \) interval.

### Ex.) \( n=3 \) \( x, y, z \in \{1, 2, 3, \infty\} \)

<table>
<thead>
<tr>
<th>( n_1 )</th>
<th>( n_2 )</th>
<th>( I_1 )</th>
<th>( I_2 )</th>
<th>( I_3 )</th>
<th>( I_\infty )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_1 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>( 1-P_{2A} )</td>
<td>( 1-P_{2A} )</td>
<td>( 1-P_{2A} )</td>
<td>( 1-P_{2A} )</td>
<td>( 1-\Sigma )</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>( P_{2A} )</td>
<td>( P_{2A} )</td>
<td>( P_{2A} )</td>
<td>( P_{2A} )</td>
<td>( 1-\Sigma )</td>
</tr>
<tr>
<td>( I_\infty )</td>
<td>0</td>
<td>( P_{2A} )</td>
<td>( P_{2A} )</td>
<td>( P_{2A} )</td>
<td>( 1-\Sigma )</td>
</tr>
</tbody>
</table>

\( P_{ij}^k \): The probability that an arc \( a_{ij} \) from node \( n_i \) to \( n_j \) is failed in the \( k^{th} \) time interval.
Probability Tables for Dynamic Nodes

- **SPARE node**

  ![SPARE node diagram]

  - \( f > x \): \( y \)
  - \( f < x \): \( y \) \( \cdot \) \( P(X_f \text{ fails at the } f^\text{th} \text{ interval}) \cdot P(Y_f \text{ fails at or before the } y^\text{th} \text{ interval}) \)
  - \( x < f < y \): \( y \) \( \cdot \) \( P(X_f \text{ fails at the } f^\text{th} \text{ interval}) \)
  - \( y \leq f < x \): \( y \) \( \cdot \) \( P(X_f \text{ fails at the } f^\text{th} \text{ interval}) \)
  - \( f = x < y \): \( y \) \( \cdot \) \( P(X_f \text{ fails before the } f^\text{th} \text{ interval}) \cdot P(Y_f \text{ fails at or before the } y^\text{th} \text{ interval}) \)
  - Else: \( y \) \( \cdot \) \( 1 \cdot (\text{sum of the other probabilities in the same row}) \)

- **SEQ node**

  ![SEQ node diagram]

  - \( g < 3 \): \( y \)
  - \( 3 \leq g < \infty \): \( y \) \( \cdot \) \( \sum P(X_f \text{ fails at the } f^\text{th} \text{ interval}) \cdot P(Y_f \text{ fails at the } y^\text{th} \text{ interval}) \) \( \text{when} \ a = k = c = g \)
  - \( g = \infty \): \( y \) \( \cdot \) \( 1 \cdot (\text{sum of the other probabilities in the same row}) \)
Development of a Tool for Generating PT

DRGGG

Relex Studio

Unreliability: 0.166873
Reliability: 0.833127
Difference: 0.003369

0.836496

0.166873
0.003369
0.833127
Development of a Tool for Generating PT

Unreliability: 0.217048
Reliability: 0.782952
Difference: 0.002629

Unreliability: 0.217048
Reliability: 0.782952
Difference: 0.002629
Development of a Tool for Generating PT

Unreliability : 0.171759
Reliability : 0.828241

Difference : 0.000112
Summary & Further Study

- We added dynamic nodes to RGGG in order to model a dynamic system intuitively.
- When analyzing reliability of a dynamic system through dynamic RGGG, making the probability tables is the most difficult work.
  - We adopted discret-time method to make probability tables.
  - The rule of filling probability tables was proposed.
  - As n(discretization number) increases, the result will be more accurate.
Summary & Further Study

- Further Study
  - Development of software tool
    - Reducing the calculation time.
    - Development of a tool for connecting the generated PT and a Bayesian Network tool.
References

