Reliability Formalisms

There are several popular graphical formalisms to express system reliability. The core of the solvers is the methods we have just examined. In particular, we will examine

- Reliability Block Diagrams
- Fault Trees
- Reliability Graphs

There is nothing particularly special about these formalisms except their popularity. It is easy to implement these formalisms, or design your own, in a spreadsheet, for example.

Reliability Block Diagrams

- Blocks represent components.
- A system failure occurs if there is no path from source to sink.

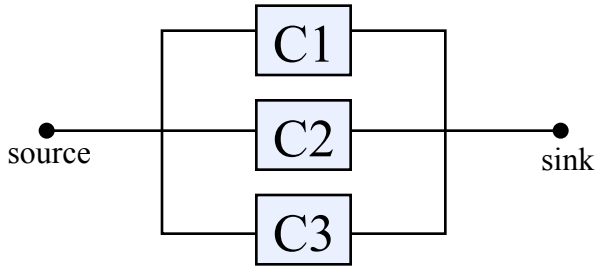
Series:

System fails if any component fails.



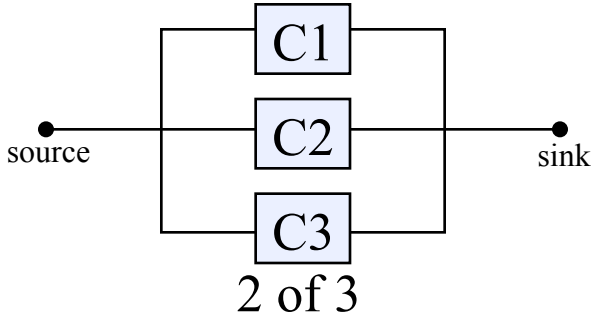
Parallel:

System fails if all components fail.



k of *N*:

System fails if at least *k* of *N* components fail.



 ©2006 William H. Sanders. All rights reserved. Do not duplicate without permission of the author.

Example

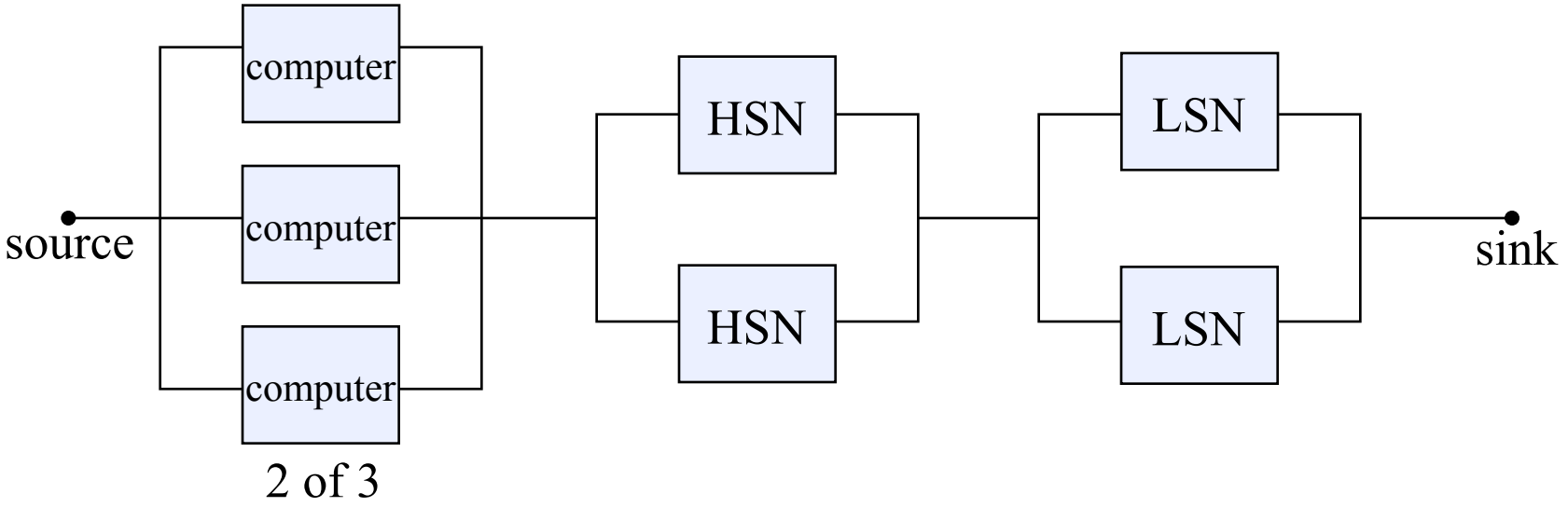
A NASA satellite architecture under study is designed for high reliability. The major computer system components include the CPU system, the high-speed network for data collection and transmission, and the low-speed network for engineering and control. The satellite fails if any of the major systems fail.

There are 3 computers, and the computer system fails if 2 or more of the computers fail. Failure distribution of a computer is given by F_C.

There is a redundant (2) high-speed network, and the high-speed network system fails if both networks fail. The distribution of a high-speed network failure is given by F_H.

The low-speed network is arranged similarly, with a failure distribution of F_L.

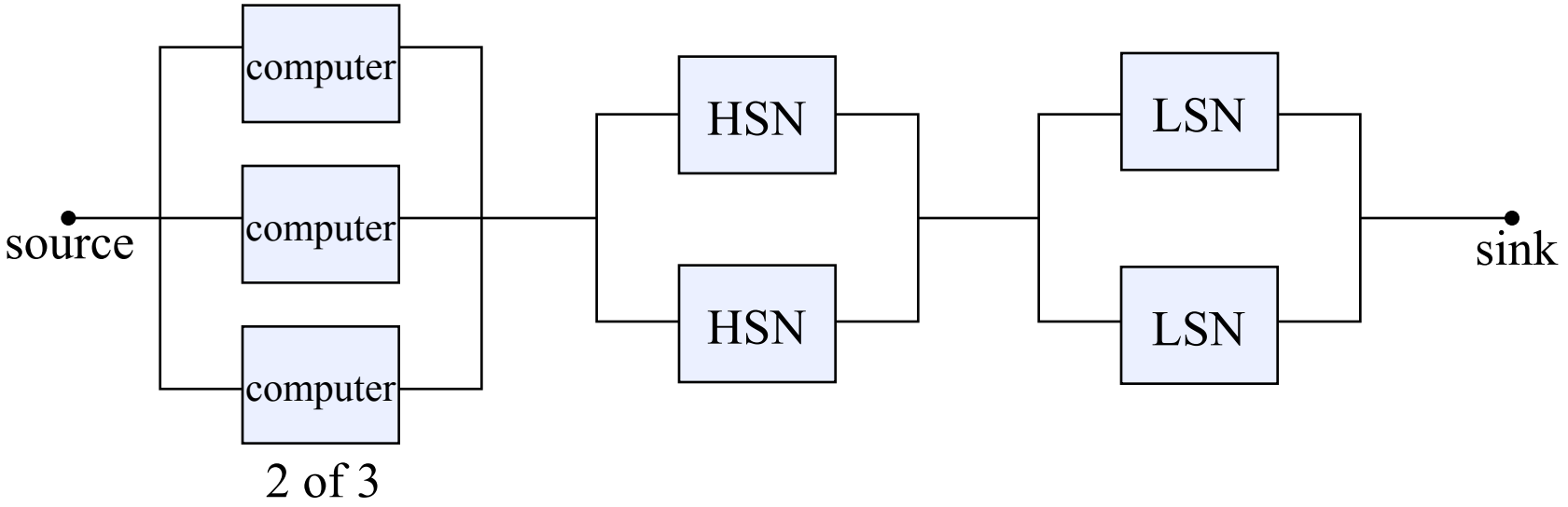
RBD Example



$$F_S(t) = 1 - \left(1 - \left(\sum_{i=2}^{3} \binom{3}{i} F_C^i(t)\left(1 - F_C(t)\right)^{3-i}\right)\right)\left(1 - \left(F_H(t)\right)^2\right)\left(1 - \left(F_L(t)\right)^2\right)$$

Probability all three systems survive to t

RBD Example

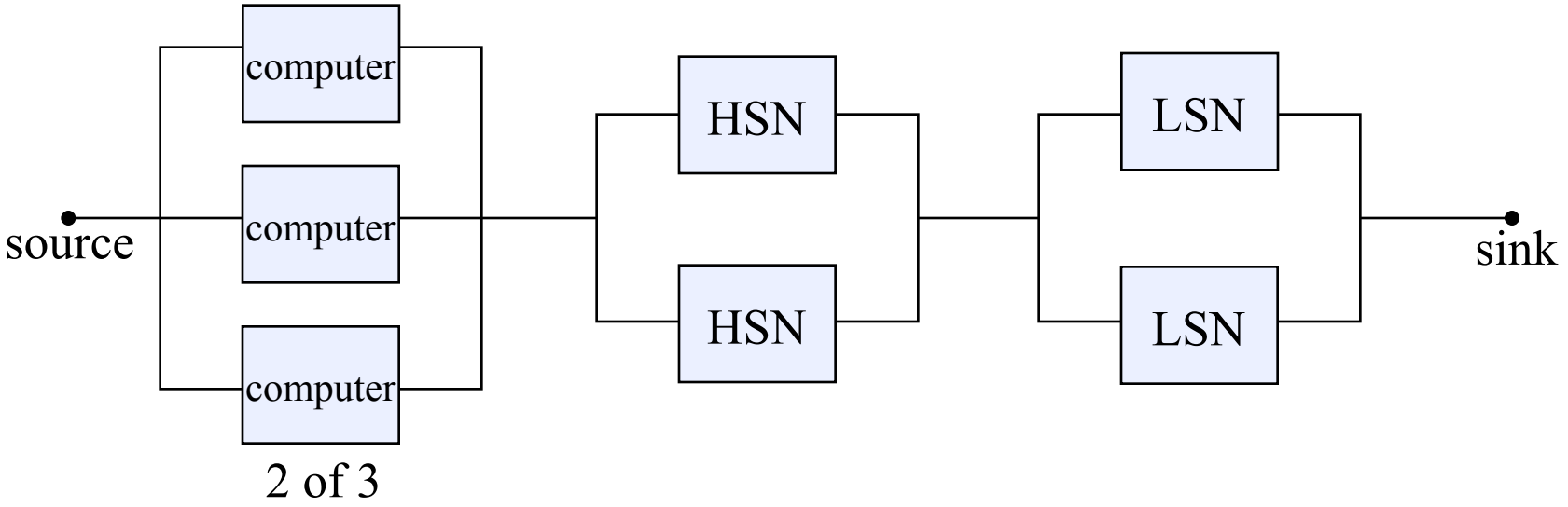


$$F_S(t) = 1 - \left(1 - \left(\sum_{i=2}^{3} \binom{3}{i} F_C^i(t)\left(1 - F_C(t)\right)^{3-i}\right)\right)\left(1 - \left(F_H(t)\right)^2\right)\left(1 - \left(F_L(t)\right)^2\right)$$

Probability low speed network survives to t

ECE/CS 541: Computer System Analysis, Instructor William H. Sanders. ©2006 William H. Sanders. All rights reserved. Do not duplicate without permission of the author.

RBD Example



$$F_S(t) = 1 - \left(1 - \left(\sum_{i=2}^{3} \binom{3}{i} F_C^i(t)\left(1 - F_C(t)\right)^{3-i}\right)\right)\left(1 - \left(F_H(t)\right)^2\right)\left(1 - \left(F_L(t)\right)^2\right)$$

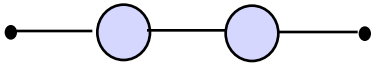
Probability both components of low speed network fail by t

Background : Series-Parallel Graphs

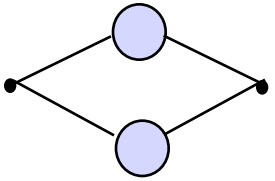
- Analysis of RBD is based on the notion of **series-parallel graphs**
- Defined recursively as follows:
- A series-parallel graph is comprised of nodes, edges, and subgraph connectors (SGC)
  - An edge connects a node and an SGC, or two SGCs
  - Every series-parallel graph has an SCG as source, and an SCG as sink
- All series-parallel graphs have one of the following forms:



Single node



Series of two SPGs
(understanding that the sink of one SPG is identically the source of the other

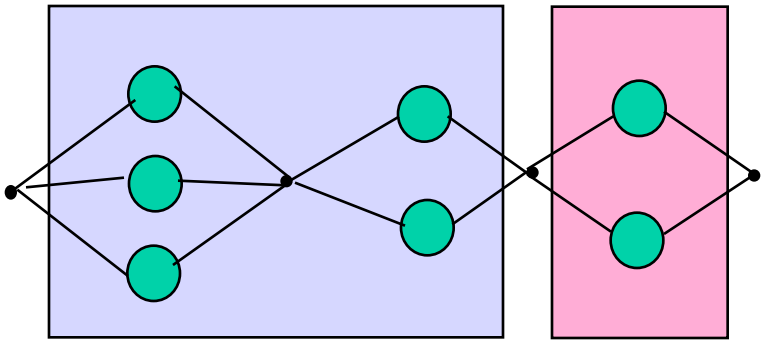


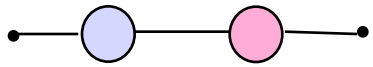
Parallel of two SPGs
(understanding that the source and sink endpoints connect (respectively) to the source and sink endpoints of the SPGs

Every node has a failure distribution
Failure time of series subgraph is minimum failure time of SPGs in series
Failure time of parallel subgraph is maximum failure time of SPGs in parallel

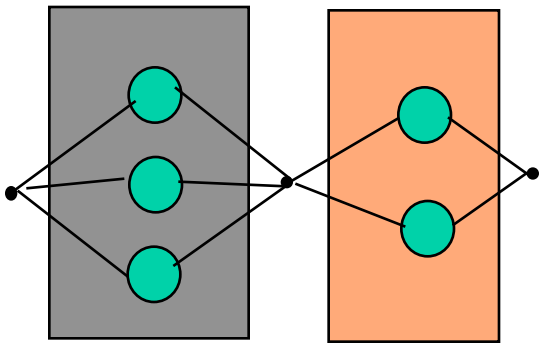
Given a graph, identify the series-parallel decomposition

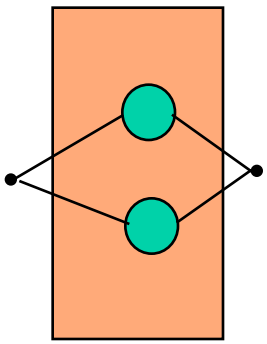
Series-Parallel Decomposition of NASA example

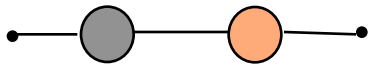


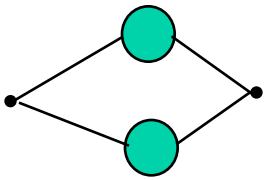


Use formula for series,
applied to failure distribution
of component subgraphs, which are ??







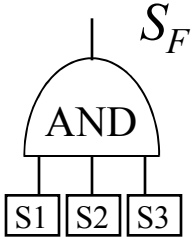


We can express this one, because node
distributions are known

Fault Trees

- Explicit represenation of system decomposition and dependency of system operation on subsystems
- Fault tree expresses *logical* conditions necessary for system failure
- Logical conditions expressed using AND, OR, and k-of-N 'gates'
  - These map into Boolean functions
- If the entire system fails if and only if every one of three subsystems S1,S2,S3 fails, we express this with and AND gate
  - Think of S1, S2, S3 being Boolean, reflecting whether subsystem has failed

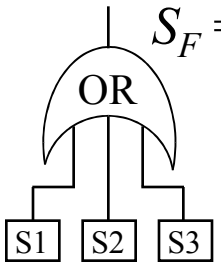
$S_F = S1 \bullet S2 \bullet S3$



S_F = Boolean expression of system failure

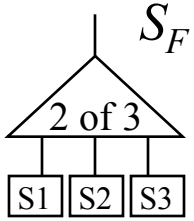
  - Conversely if entire system fails if and only if any one of S1, S2, S3 fails, we denote this with an OR gate

$S_F = S1+S2+S3$



Fault Trees

- If the entire system fails if k-of-N subsystems fail, we use a k-of-N node



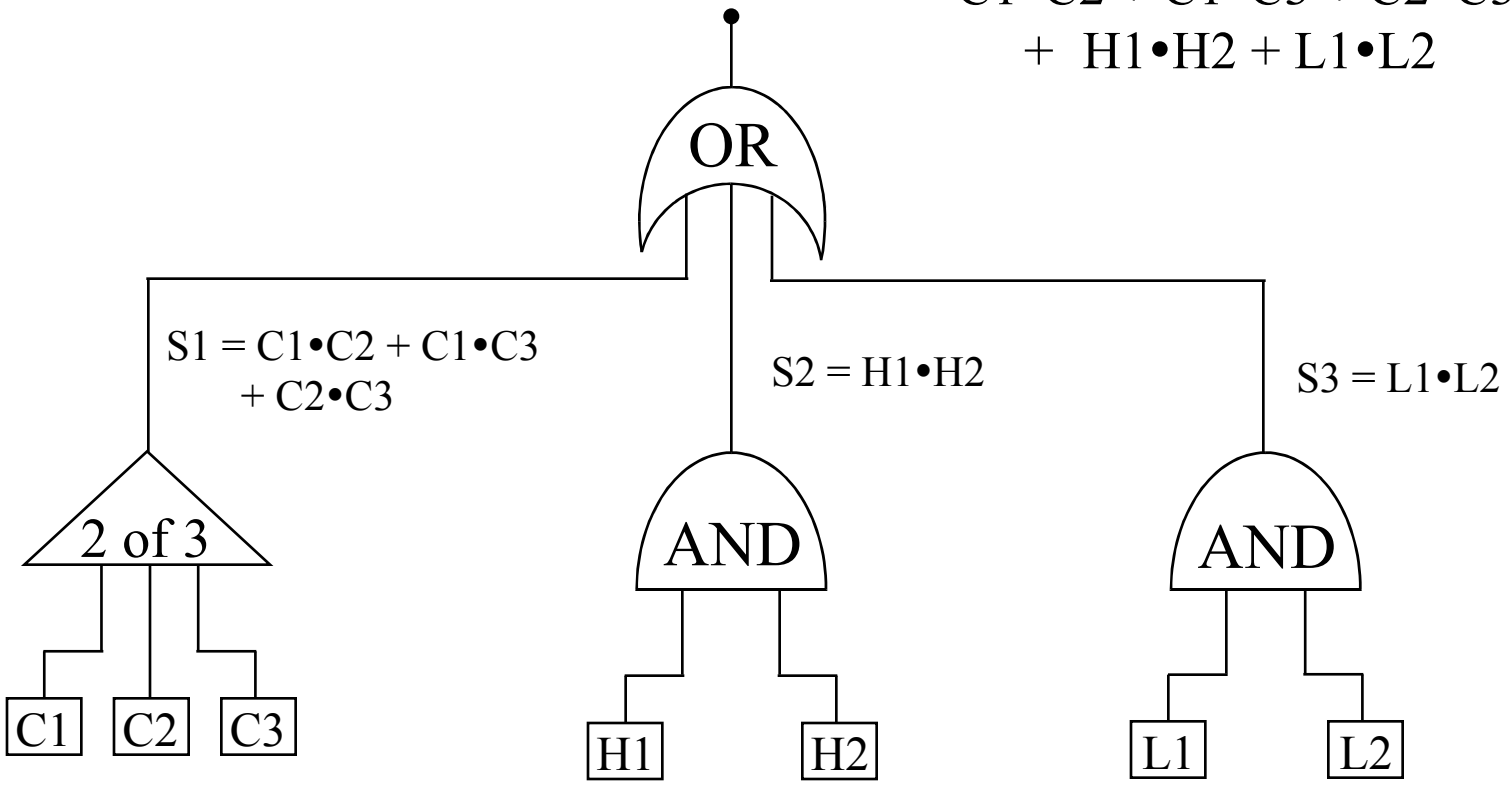
S_F = S1•S2 + S1•S3 + S2•S3

Boolean Expression of System Failure enumerates all ways of failing k components

- Then apply definition recursively to describe Boolean functions of the state of the subsystems
- Stop recursive refinement when the subsystem is a component

Fault Tree Expression of NASA example

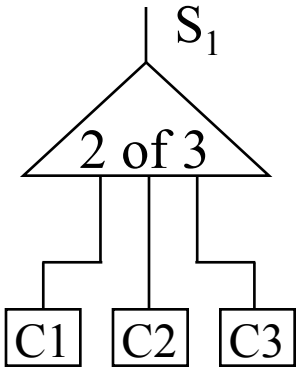
$$S_F = S1 + S2 + S3$$
$$= C1 \bullet C2 + C1 \bullet C3 + C2 \bullet C3 + H1 \bullet H2 + L1 \bullet L2$$



Using Fault Tree to Compute System Reliability

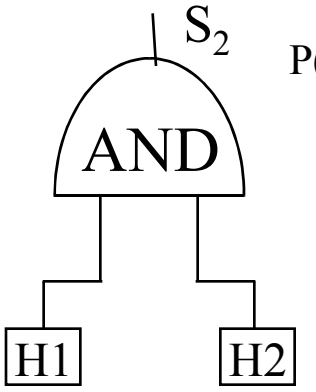
Using tree structure

- Start at the leaves, compute reliability of subsystem
- Example, NASA system
  - Use FT to denote 'failure time of'

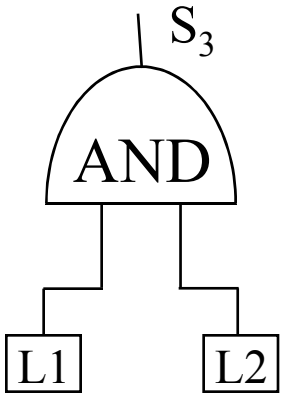


$P(FT(S_1) < t) = P(FT(\text{ 2-of-3}(C1,C2,C3)) < t)$

and we have formula for this (most directly when C1,C2,C3) are identical

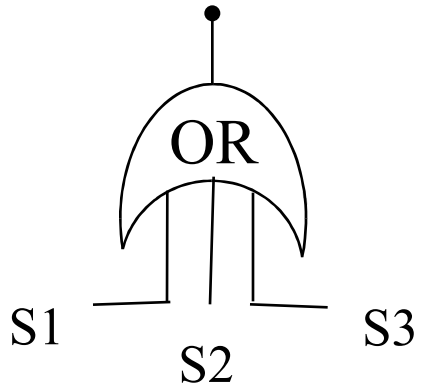


$P(FT(S_2) < t) =$
$P(\text{ FT(H1}< t) \text{ AND FT(H2}< t)$



$P(L(S_3) < t) =$
$P(\text{ FT(L1}< t) \text{ AND FT(L2}< t)$

Using Fault Tree to Compute System Reliability



$$P(\ FT(S_F) < t\ ) = P(\ FT(S1 < t) \text{ OR } FT(S2 < t) \text{ OR } FT(S3) < t\ )$$

Which we solve using previous subsolutions

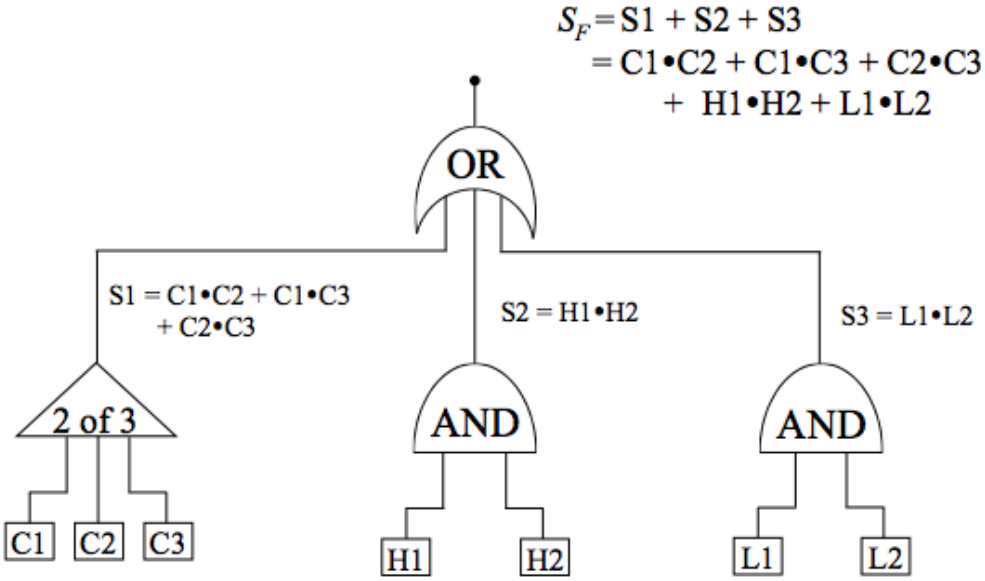
$$P(FT(S_1) < t) = P(FT(\text{ 2-of-3}(C1,C2,C3)) < t)$$

$$P(FT(S_2) < t) = P(\ FT(H1 < t) \text{ AND } FT(H2 < t)$$

$$P(L(S_3) < t) = P(\ FT(L1 < t) \text{ AND } FT(L2 < t)$$

Using Fault Tree to Discover Failure Causes

Fault tree leads to Boolean expression for system failure, recall



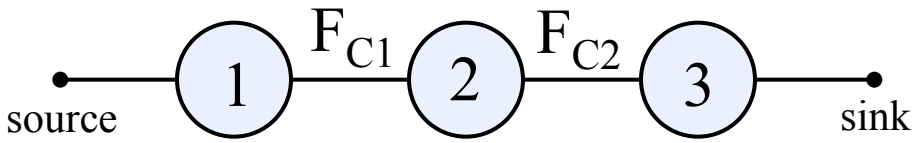
Each product in sum-of-products form identifies set of components which, when all failed, causes system to fail

- Can use this to look for minimum number of component failures that lead to system failure

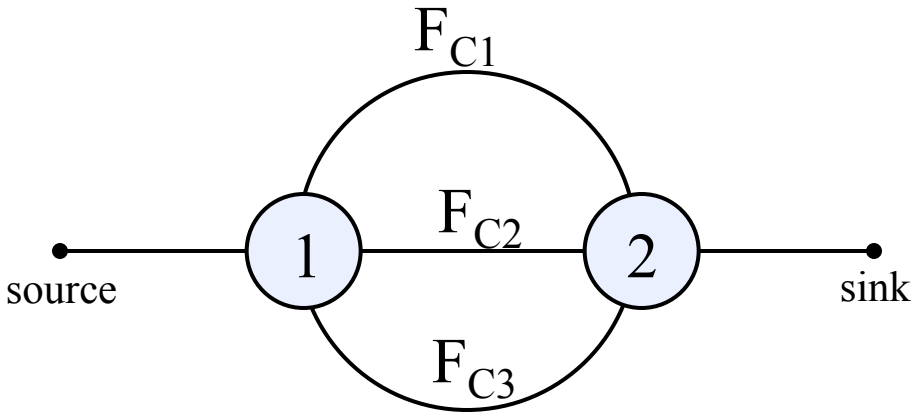
Reliability Graphs

- The *arcs* represent components and have failure distributions.
- A failure occurs if there is no path from source to sink.

Can implement series:



Can implement parallel:

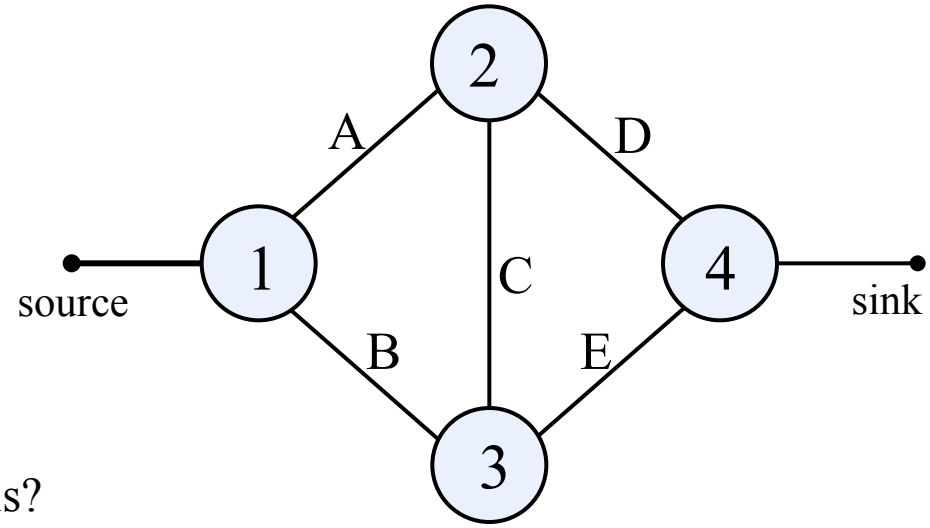


Reliability Graph Example

Reliability graphs can implement more complex interactions.

- Graphs from RBD and FT are essentially "series-parallel", a special class of graph
- A reliability graph is more general

For example, a telephone network "fails" if there is no path from source to sink.



How do we solve this?

Solving by Conditioning

Recall that $P[E \mid F] = \dfrac{P[E \cap F]}{P[F]}$

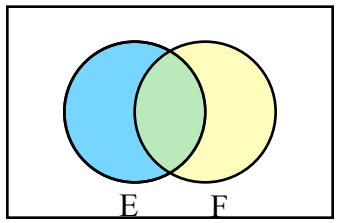
If F and $\overline{F}$ are complementary events, i.e.,

$$F \cup \overline{F} = \Omega \text{ and } F \cap \overline{F} = \varnothing$$

then there is a trick :

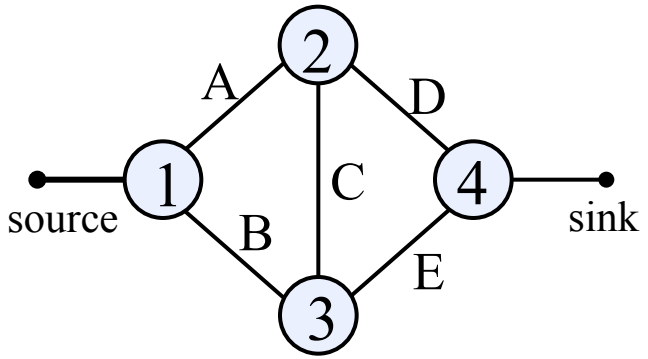
$$P[E] = P[E \cap F] + P[E \cap \overline{F}]$$

$$P[E] = P[E \mid F]P[F] + P[E \mid \overline{F}]P[\overline{F}]$$



Ω

If you can solve $P[E \mid F]$, $P[F]$, $P[E \mid \overline{F}]$, and $P[\overline{F}]$, then you can solve $P[E]$.

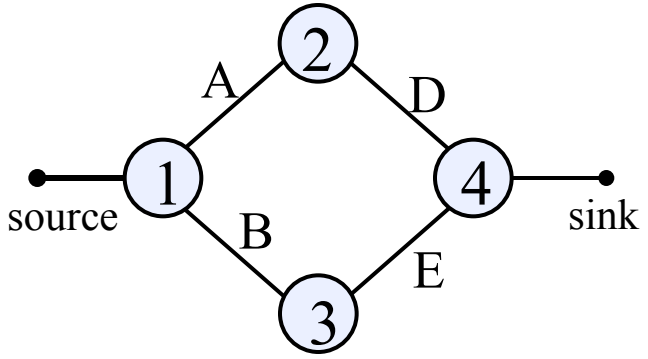


First, *condition* the system on link C being failed.

Then the system becomes the series AD in parallel with the series BE.

S is time of failure

Pr{A and D alive at t}



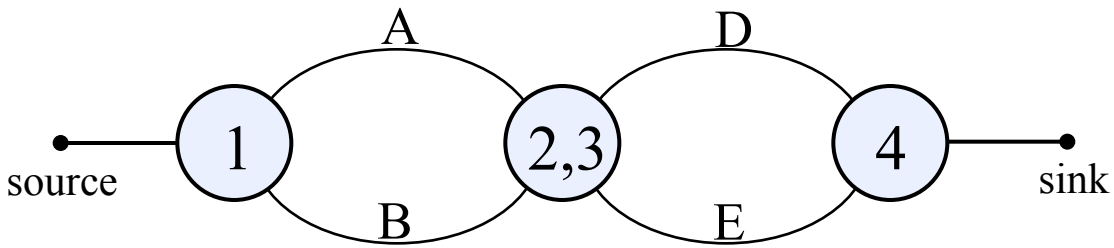
$$F_{S|C\ Fail}(t) = P[S \le t \mid C \le t] = \left(1-\left(1-F_A(t)\right)\left(1-F_D(t)\right)\right)\left(1-\left(1-F_B(t)\right)\left(1-F_E(t)\right)\right)$$

and $P[C \le t] = F_C(t)$

Pr{both AD and BE series dead at t}

Second, *condition* the system on link C being up.

- System fails if either both A&B fail, or both D and E fail --- a series



At least one of {A,B} up

$$F_{S|C\,up}(t) = P[S \le t \mid C > t] = 1 - (1 - F_A(t)F_B(t))(1 - F_D(t)F_E(t)),$$

and $P[C > t] = 1 - P[C \le t] = 1 - F_C(t)$

Thus, $F_S(t) = F_{S|C\,Fail}(t)F_C(t) + F_{S|C\,up}(t)(1 - F_C(t))$.

Conditioning Fault Trees

It is also possible to use conditioning to solve more complex fault trees. If the same component appears more than once in a fault tree, it violates the independent failure assumption. However, a conditioned fault tree can be solved.

Example: A component *C* appears multiple times in the fault tree

conditioned on C being up :

- AND gates with C as input become 0 (impossible for all components to be failed)
- OR gates with C as input remain OR gates without C as input
- k of N gates become k of (N-1) gates

conditioned on C being down

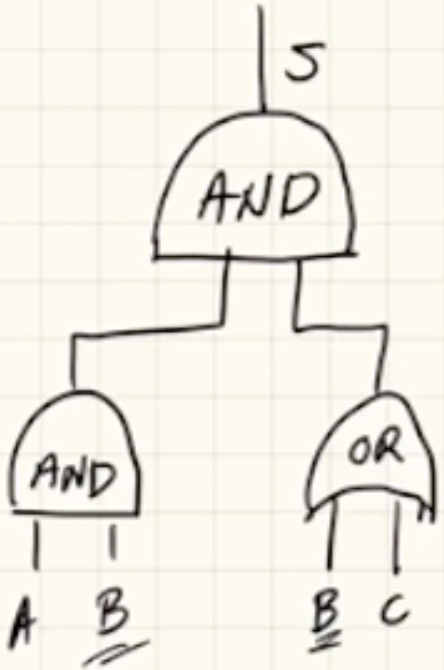
- AND gates with C as input remain AND gates without C as input
- OR gates with C as input become 1 (any failure causes the OR to fail)
- k of N gates become (k-1) of N-1 gates

$$F_S(t) = F_{S|C\ Fail}(t)F_C(t) + F_{S|C\ Up}(t)(1 - F_C(t))$$

Where $S|C\ Fail$ is the system given that C has failed

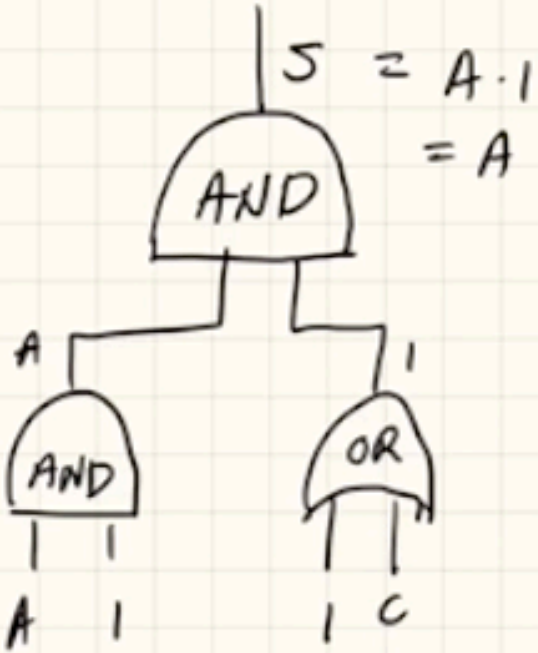
and $S|C\ Up$ is the system given that C has not failed.

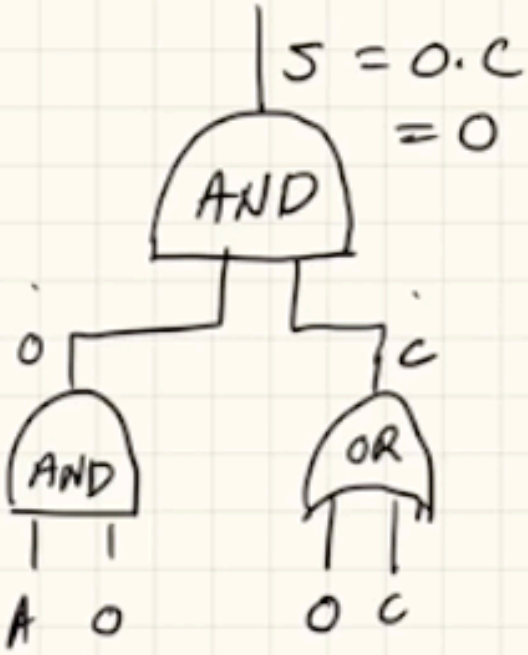
$$P(FT(S) \leq t \mid FT(B) \leq t)$$



$$P(FT(S) \le t \mid FT(B) \le t)$$

$$= P(FT(A) \le t)$$



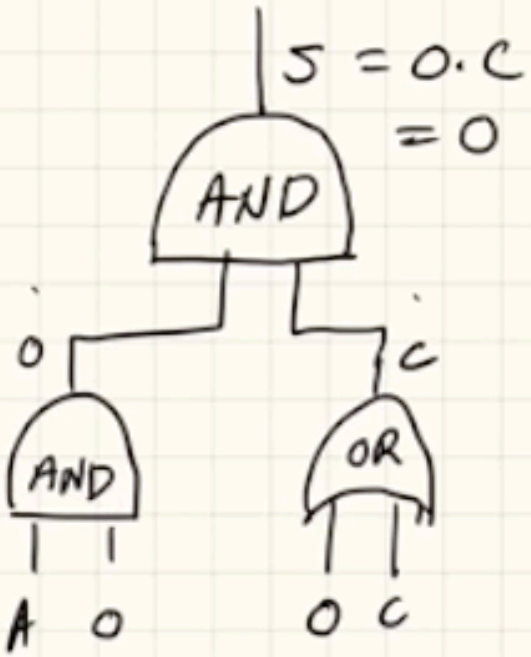


$$P(FT(S) \le t \mid FT(B) \le t)$$

$$= P(FT(A) \le t)$$

$$P(FT(S) \le t \mid FT(B) > t)$$

$$= 0$$



$$P(FT(S) \le t \mid FT(B) \le t)$$

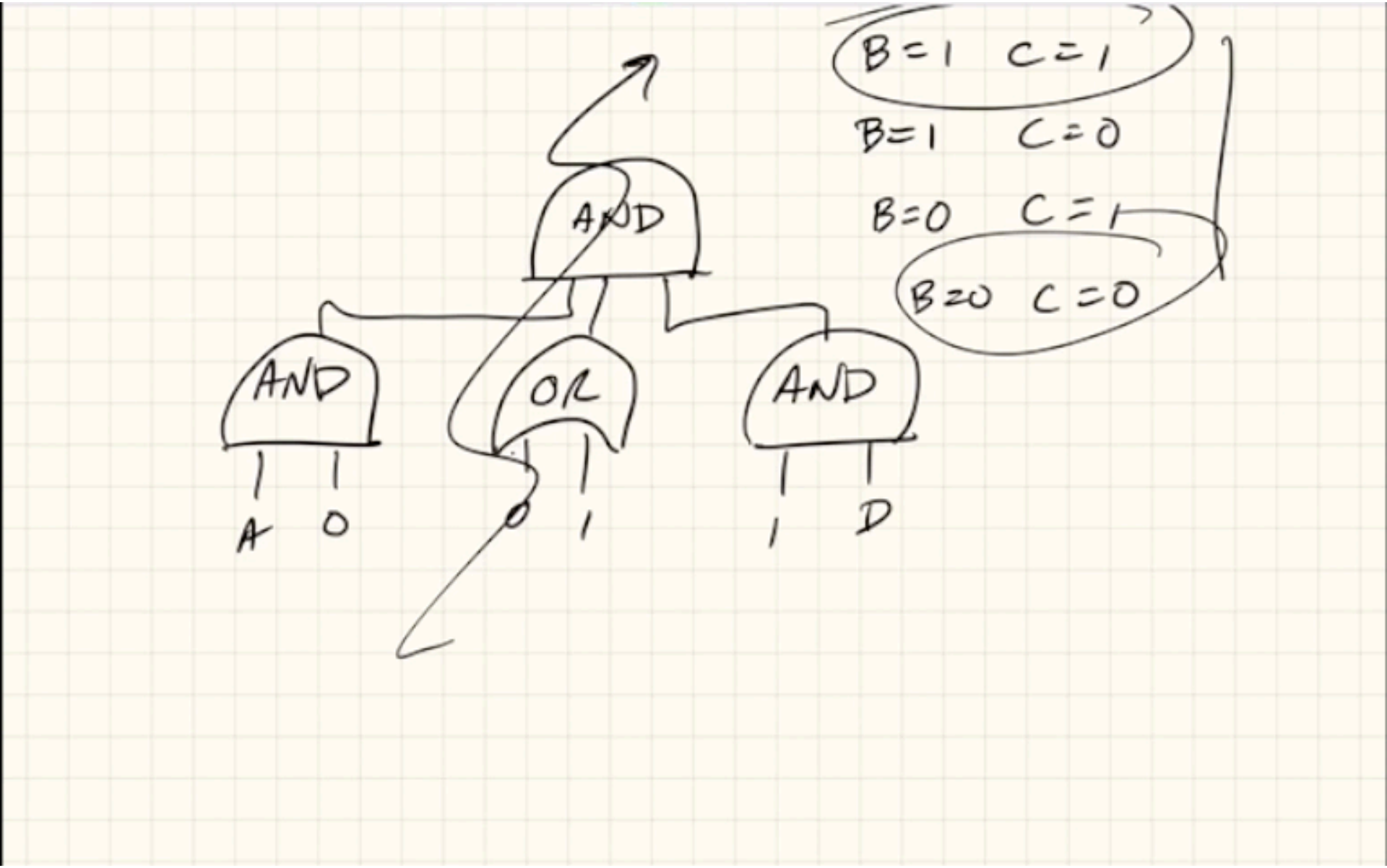
$$= P(FT(A) \le t)$$

$$P(FT(S) \le t \mid FT(B) > t)$$

$$= 0$$

$$P(FT(S) \le t) = P(FT(B) \le t) \cdot P(FT(A) \le t)$$

$$t > 0$$


B=1 C=1
B=1 C=0
B=0 C=1
B=0 C=0
AND
AND
OR
AND
1
0
0
1
1
D
A

Reliability/Availability Point Estimates

- Frequently, the desired measure of a reliability model is the reliability at some time t. Thus, the distribution of the system reliability is superfluous; $R(t)$ is the only thing of interest.

- This condition simplifies computation because all that is necessary for solution is the reliability of the components at time t. Solution then becomes a straightforward computation.

- If a system is described in terms of the *availability* of components at time t, then we may compute the system availability in the same way that reliability is computed. The restriction is that all component behaviors must be independent of one another.

Reliability/Availability Tables

A system comprises N components. Reliability of component i at time t is given by $R_{Xi}(t)$, and the availability of component i at time t is given by $A_{Xi}(t)$.

Condition	System Reliability	System Availability
system fails if all components fail	$R_S(t) = 1 - \prod_{i=1}^{n} (1 - R_{Xi}(t))$	$A_S(t) = 1 - \prod_{i=1}^{n} (1 - A_{Xi}(t))$
system fails if one component fails	$R_S(t) = \prod_{i=1}^{n} R_{Xi}(t)$	$A_S(t) = \prod_{i=1}^{n} A_{Xi}(t)$
system fails if at least k components fail, identical distribution	$R_S(t) = \sum_{i=k}^{N} \binom{N}{i} (1 - R_{Xi}(t))^i R_X(t)^{N-i}$	$A_S(t) = \sum_{i=k}^{N} \binom{N}{i} (1 - A_X(t))^i A_X(t)^{N-i}$
system fails if at least k components fail, general case	$R_S(t) = \sum_{g \in G_k} \left(\prod_{X \in g} (1 - R_X(t)) \right) \left(\prod_{X \notin g} R_X(t) \right)$	$A_S(t) = \sum_{g \in G_k} \left(\prod_{X \in G} (1 - A_X(t)) \right) \left(\prod_{X \notin g} A_X(t) \right)$

Estimating Component Reliability

- For hardware, MIL-HDBK-217 is widely used.
  - Not always current with modern components.
  - Lacks distributions; it only contains failure rates.
  - While not perfect, it seems to be the best source that exists. However, numbers from MIL-HDBK-217 should be used with caution.

- Due to the nature of software, no accepted mechanism exists to predict software reliability before the software is built.
  - Best guess is the reliability of previously built similar software.

- In all cases, numbers should be used with caution and adjusted based on observation and experience.

- No substitute for empirical observation and experience!

Modeling Process

- Reliability models are built only after proper service is specified.
- Reliability models are built to answer the question "What subsystem or components must be proper for the system to be proper?"
- Build models hierarchically out of subsystems.
- Estimation and guesses are acceptable, but state them explicitly.
- If unsure, do sensitivity analysis to see how much it matters.

Reliability Modeling Process

- Realistic systems result in large RBDs and must be managed hierarchically.

RBD Process(system)
Define the system
Define "proper service"
Create RBD out of components
for each component
    if component is simple
        obtain reliability data of component
    else
        Do RBD Process(component)
end if
Compute reliability of system
Do results meet specification?
Modify design and repeat as necessary

Review

- Reliability: review of definition
- Failure rate
- System reliability
  - Independent failure assumption
  - Minimum, maximum, *k* of *N*
  - Reliability block diagrams, fault trees, reliability graphs
- Reliability modeling process