UNIVERSITY OF MASSACHUSETTS

Department of Electrical and Computer Engineering

Fault Tolerant Computing | ECE655

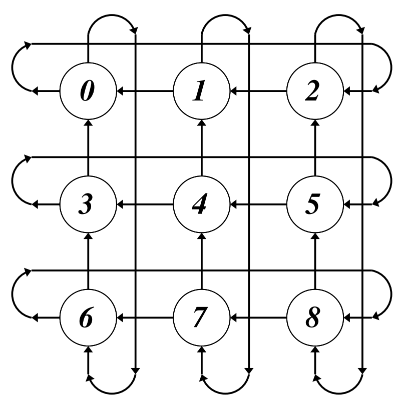
Homework 4

1. Show that the three cases enumerated in connection with the derivation of the hypercube network reliability lower bound, are mutually exclusive. Further, show that H_n is connected under each of these cases. Assume that $q_c = 0$, i.e., that the nodes do not fail.

2. Obtain by simulation the network reliability of H_n for $n = 5, 6, 7$. Assume that $q_c = 0$. Compare this result in each instance with the lower bound that we derived.

3. The links in an H_3 hypercube are directed from the node with the lower index to the node with the higher index. Calculate the path reliability for the source node 0 and the destination node 7. Denote by $p_{i,j}$ the probability that the link from node i to node j is operational and assume that all nodes are fault-free.

4. All the links in a given 3×3 torus network are directed as shown in the diagram below. Calculate the path reliability for the source node 1 and the destination node 0. Denote by $p_{i,j}$ the probability that the link from node i to node j is operational and assume that all nodes are fault-free.



5. In this problem, we will use Bayes's law to provide some indication of whether bugs still remain in the system after a certain amount of testing. Suppose you are given that the probability of uncovering a bug (given that at least one exists) after t seconds of testing is $1 - e^{-\mu t}$. Your belief at the beginning of testing is that the probability of having at least one bug is q. (Equivalently, you think that the probability that

the program was completely bug-free is $p = 1 - q$.) After t seconds of testing, you fail to find any bugs at all. Bayes's law gives us a concrete way in which to use this information to refine your estimate of the chance that the software is bug-free: find the probability that the software is actually bug-free, *given that you have observed no bugs at all, despite t seconds of testing.*

Let us use the following notation:

- A is the event that the software is actually bug-free.
- B is the event that no bugs were caught despite t seconds of testing.

(a) Show that $Prob\{A|B\} = \frac{p}{p+qe^{-\mu t}}$

(b) Fix $p = 0.1$, and plot curves of $Prob\{A|B\}$ against t for the following parameter values: $\mu = 0.001, 0.01, 0.1, 1.0, \;\; 0 \leq t \leq 10000$.

(c) Fix $\mu = 0.01$ and plot curves of $Prob\{A|B\}$ against t for the following parameter values: $p = 0.1, 0.2, 0.3, 0.4, 0.5$.

(d) What conclusions do you draw from your plots in (b) and (c) above?

6. Based on the expressions for sensitivity and specificity derive an expression for the probability of a false alarm (in a single stage of a recovery block structure).

7. In the context of the SIHFT technique, the term *data integrity* has been defined as the probability that the original and the transformed programs will not both generate identical incorrect results. Show that if the only faults possible are single stuck-at faults in a bus (see Figure 1) and k is either -1 or 2^ℓ with ℓ an integer, then the data integrity is equal to 1. Give an example when the data integrity will be smaller than 1. (Hint: Consider ripple-carry addition with $k = -1$.)

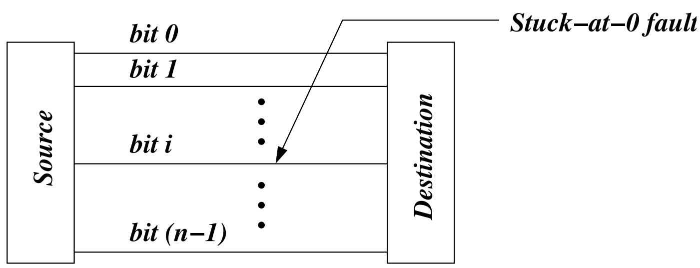


Figure 1: Example of the use of SIHFT.

8. Compare the use of the AN code to the RESO technique. Consider the types of faults that can be detected and the overheads involved.