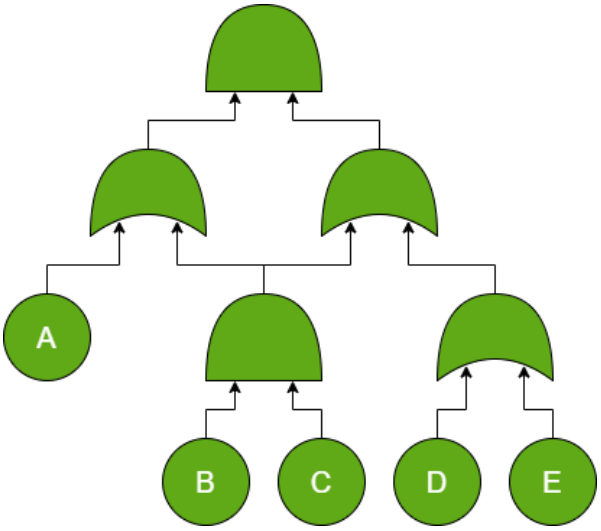
Exercises week 2

*Exercises marked with * are extra challenging.*

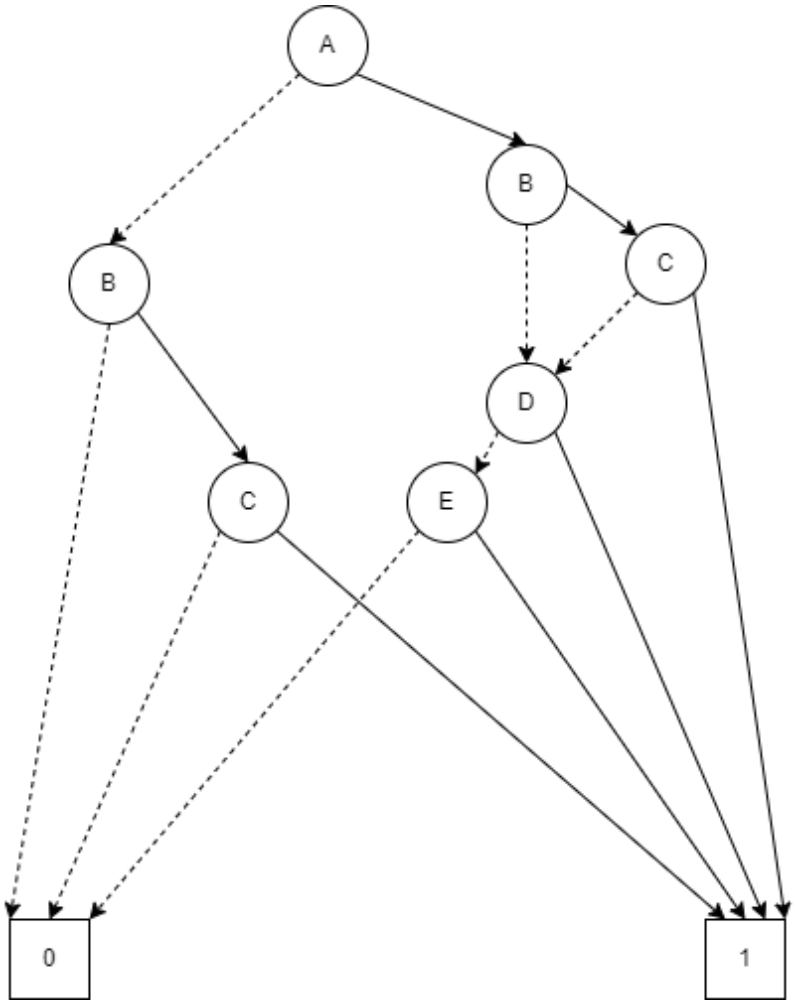


Problem 1 (Homework): For this fault tree:

1. What are the minimal cut sets?
2. Create a BDD for this fault tree, with variable order A,B,C,D,E.
3. Suppose the failure probabilities are $p_A = 1/2$, $p_B = 1/3$, $p_C = 1/4$, $p_D = 1/5$, $p_E = 1/6$. Calculate the fault tree's failure probability using the BDD.
4. Also approximate the failure probability via cut-set approximation.

Solution:

1. The minimal cut sets are {A,D}, {A,E}, and {B,C}.

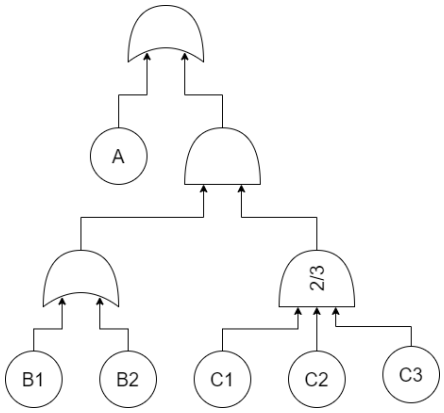
2. 

3. There are 6 paths A -> 1 in the BDD:
   a. A → B → C → 1 — $p_A p_B p_C$ — 0.04167
   b. A → B → C ⇢ D → 1 — $p_A p_B (1-p_C) p_D$ — 0.025
   c. A → B → C ⇢ D ⇢ E → 1 — $p_A p_B (1-p_C)(1-p_D) p_E$ — 0.01667
   d. A → B ⇢ D → 1 — $p_A (1-p_B) p_D$ — 0.06667
   e. A → B ⇢ D ⇢ E → 1 — $p_A (1-p_B)(1-p_D) p_E$ — 0.04444
   f. A ⇢ B → C → 1 — $(1-p_A) p_B p_C$ — 0.04167

   Hence the total failure probability is the sum, which is 0.23611.
4. We have three minimal cut sets:
   a. {A,D} — $p_A p_D$ — 0.1
   b. {A,E} — $p_A p_E$ — 0.08333
   c. {B,C} — $p_B p_C$ — 0.08333

   In total we get 0.26667, which indeed is an overapproximation of the answer to question 3.

Problem 2: In the FT to the right, the failure probability of each BE is p.



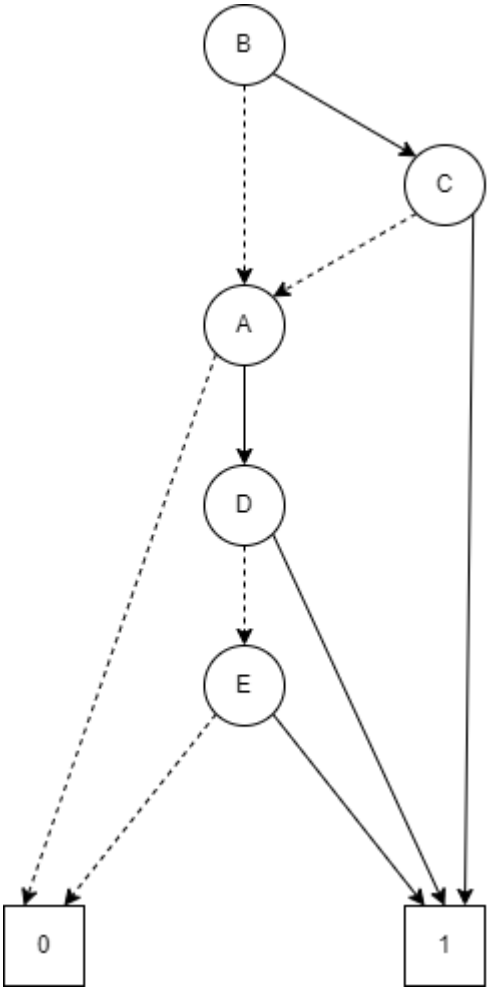
1. Suppose you want to calculate the failure probability of the FT exactly. Do you use the bottom-up method, the BDD method, or the cut set method? Why?
2. Use the method you selected in the previous question to calculate the failure probability.

Solution:

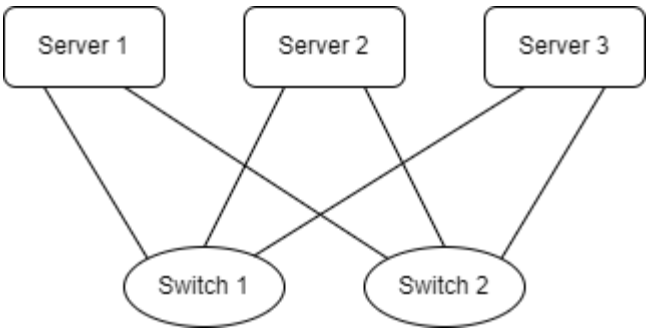
1. The cut set method is out since that only provides an (over-)approximation of the failure probability. Since this FT is tree-structured (no nodes with multiple parents), we can use the bottom-up method. This is preferable to the BDD method, since we don't have to create the BDD first.
2. We calculate the probabilities bottom-up:
   OR(B1,B2): $1-(1-p)^2 = 2p-p^2$
   Voting gate: $3p^2(1-p)+p^3 = 3p^2-2p^3$
   AND-gate: $(2p-p^2)(3p^2-2p^3) = 6p^3-7p^4+2p^5$
   Root: $1-(1-p)(1-6p^3+7p^4-2p^5) = p+6p^3-13p^4+9p^5-2p^6$

Problem 3: Find a variable order for the FT of problem 1 that results in a smaller BDD than that from 1.2.

Solution: There is more than one solution to this, but a key insight is that once we know the failures of B and C, the rest does not matter anymore. So we take the order B,C,A,D,E, and we get the BDD on the right.



Problem 4: Consider the following situation: You are the reliability engineer of the company CoolCloudSolutions. Your boss has promised your customers a reliability of 99%. The architecture is depicted in the figure below:

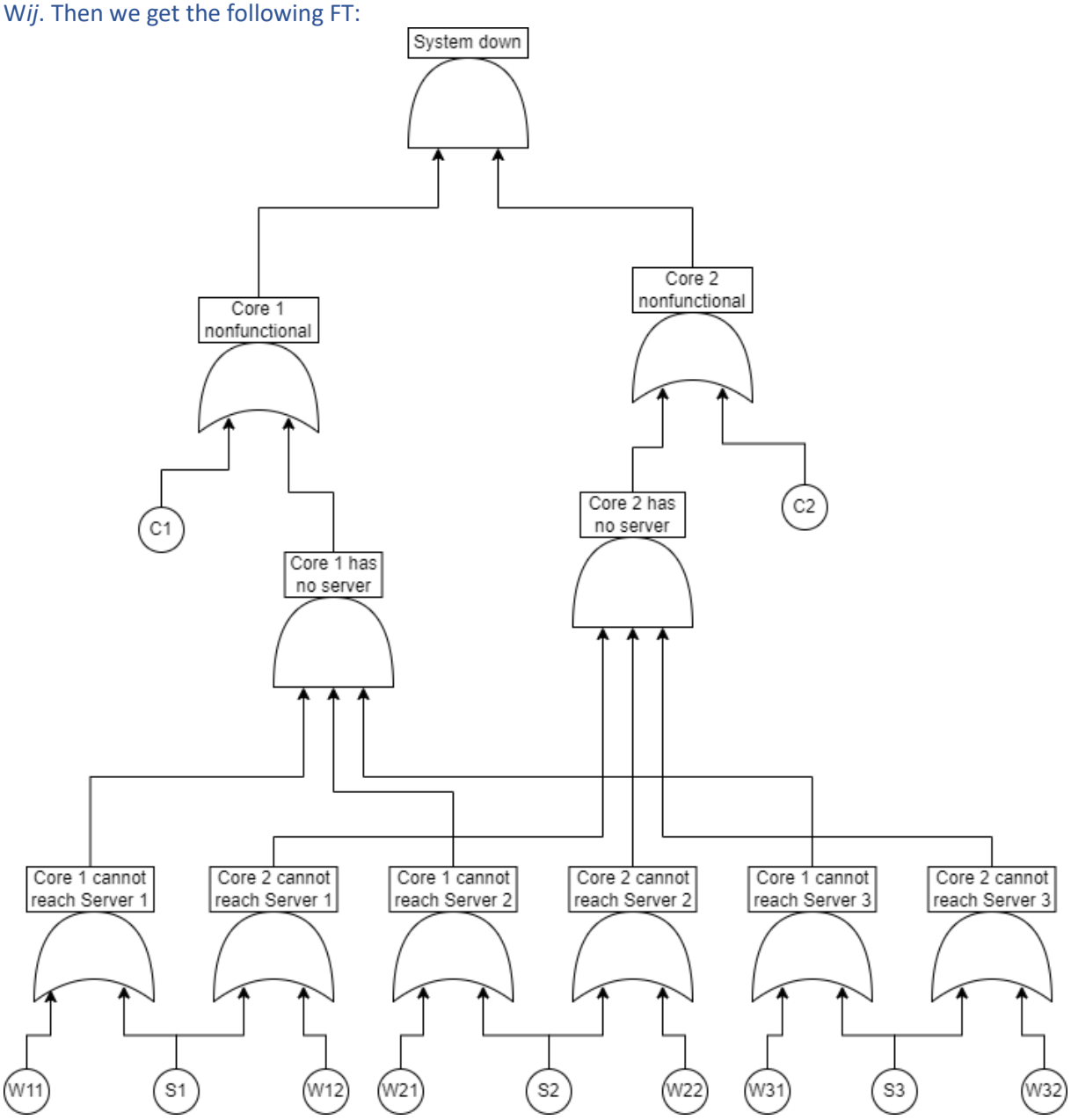


1. Model this system as a FT. Assume that all servers have the same functionality. For the system to be operational, at least one of the 3 servers must be operational, together with its network cable and one core switch.
2. How many minimal cut sets are there of order 1,2,3?
3. What are the most critical elements?

4. Suppose the servers have failure probability 0.1, the switches have failure probability 0.2, and the cables have failure probability 0.2. Does this architecture meet the reliability requirements?
5. *How many cut sets are there in total?

Solution:

1. We name the servers S1—S3, the cores C1,C2, and the cable from server *i* to core *j* we name W*ij*. Then we get the following FT:



2. Every attack must reach both nodes “Core 1 nonfunctional” and “Core 2 nonfunctional”. Since there is no single BE that activates both of these, there is no MCS of size 1. There is a single MCS of size 2, namely {C1,C2}. Now let us find all MCS of size 3. If an MCS contains C1, but not C2, we must activate all 3 children of “Core 2 has no server”. This requires at least 3 additional BEs, bringing the total to 4. So an MCS of size 3 cannot contain C1, and by symmetry it cannot contain C2 either. So we must activate both “Core 1 has no server” and “Core 2 has no server” with only 3 BEs, which must be S1,S2,S3. We conclude that {S1,S2,S3} is the only MCS of size 3.
3. The most critical elements are C1 and C2, since these are in the only cut set of size 2.

4. No. The probability of {C1,C2} taking plase is 0.04, so the total reliability will be at most 96%. In fact, it will be less than that, since there are additional MCS.
5. We divide the MCS into four groups:
    a. MCS that contain both C1 and C2. There is only one of these, namely {C1,C2}.
    b. MCS that contain C1 but not C2. These activate all three children of "Core 2 has no server". For each of these children, we have 2 options, which would give us 2^3=8 different MCS. However, we discount the option {C1,S1,S2,S3} since this is not minimal. This leaves us with 7 MCS in this category.
    c. MCS that contain C2 but not C1. By symmetry, there are again 7 of these.
    d. MCS that contain neither C1 nor C2. These have to activate all of the lower 6 OR-gates. Thus, for server 1 we have two options: either S1 fails, or both W11 and W12 fail. The same is true for the other servers, so this gives us 2^3 = 8 MCS in total.

   We conclude that there are 1+7+7+8 = 23 MCS.

Problem 5: Suppose that we extend the fault tree formalism with an XOR (eXclusive OR) gate. This has at least 2 children, and fails if exactly one of its children fails. Now consider a XOR-gate with independent children A and B.

1. Assume that A and B fail with probability p_A and p_B respectively. What is the probability for the XOR gate to fail?
2. Suppose the probability for A to fail within time t is given by $p_A(t) = 1 - e^{-\lambda t}$ and probability for B to fail within time t is given by $p_B(t) = 1 - e^{-\mu t}$. What is the probability for the XOR gate to fail within time t?
3. *Which of the following failure probability computation methods still work for XOR-FTs?
    1. The bottom-up method (for FTs that are actually trees, i.e., nodes do not share children)
    2. The BDD method
    3. The cut set method for overapproximating probabilities.

Solution

1. This probability is $p_A+p_B-2p_Ap_B$.

2. $e^{-\lambda t}+e^{-\mu t}-2e^{-(\lambda+\mu)t}$.

3. The answer is as follows:

a. The bottom-up method still works, because at any node, its childrens' failures are still independent.

b. The BDD method still works, as this works for any Boolean function.

c. The cut set method still works: the top node still only fails if there is some minimal cut set that fails.