1 Dijkstra’s Algorithm

1. A network between A and B is depicted in Figure 1. The numbers on the links correspond to the probability (as a percentage) that the link may fail. Link failures are independent from each other. How could you find the most reliable path from A to B? **Hint:** Use Dijkstra’s algorithm.

![Figure 1: Network Topology](image)

**Solution:**
Recall that Dijkstra’s Algorithm computes the *shortest* paths in a network. We thus need to map our problem to an optimization problem (*shortest*) that respects the monotonicity of the Dijkstra operation (addition). Here is why: the combination of two paths of lengths $\ell_1$ and $\ell_2$ yields a new path of length $\ell = \ell_1 + \ell_2$, where $\ell > \ell_i$. For probabilities of link failures, this property does not hold. Two links with failure probabilities $f_1$ and $f_2$ do not simply yield a path with failure probability $f = f_1 + f_2$. Why? The probability that the composed path fails is computed as (1) link 1 fails and link 2 is up, or (2) link 1 is up and link 2 fails, or (3) both links fail.

It is thus easier to compute the probability of *availability* instead. Here, it holds that two links with availability probabilities $a_1$ and $a_2$ simply yield a path with
availability probability $a = a_1 \cdot a_2$. Since we have that $a_i \leq 1$ for all $i$, the monotonocity property is preserved. We can thus apply Dijkstra with multiplication.

We first compute the availability probability for each link by $a_i = 100 - f_i$, as shown in Figure 2. Next, we run Dijkstra to obtain that the most reliable path from A to B is via C and D. The availability probability is 92.21, hence we obtain a failure probability of 7.79 for this path.

Figure 2: Network

2 Distance-Vector Routing

1. In Figure 3, after the network stabilizes, the link connecting nodes C and D becomes broken. Show how nodes A, B and C might experience the count-to-infinity phenomenon for their distances to node D. In demonstrating the count-to-infinity problem, show at least two updates containing distance-to-D, for each of the nodes A, B and C. For each of these updates, specify the distance-to-D advertised to the neighbors. Will the problem be resolved after some iterations?

Figure 3: Network Topology

Solution: Time 0
From A to D: distance 3
From B to D: distance 3
From C to D: distance 2

*The link fails*

Time 1
From A to D: distance 3
From $B$ to $D$: distance 3
From $C$ to $D$: distance 4

**Time 2**
From $A$ to $D$: distance 5
From $B$ to $D$: distance 5
From $C$ to $D$: distance 4

...  
From $A$ to $D$: distance 102
From $B$ to $D$: distance 100
From $C$ to $D$: distance 101

Yes, the problem will eventually be resolved. Thanks to the link between $B$ and $D$, all nodes will find an alternative path to $D$ and reach a stable state.