



Algorithms and Data Structures – Conditional Course Winter Term 2026/27

Exercise Sheet 10

Due: Wednesday, 13 January 2027, 12:00

Exercise 1: Dijkstra's Algorithm

(14 Points)

Implement Dijkstra's algorithm from the lecture. Consider a grayscale map such as the following: The



map is a grid of different shades of gray. Interpret each square as a vertex with a directed edge to each of its (up to four) neighbors. An edge's weight depends on the destination square's shade: lighter squares have lower incoming edge weights; black squares are unreachable, with weight ∞ .

The file `dijkstra_greyscale_map_large.txt` describes this graph. Use `graph.py` to convert it into a graph/adjacency list.¹ The first line specifies its size. Each subsequent line gives a vertex's x - and y -coordinates, its gray value (needed only for plotting), and the weights used to construct its edges.

Implement Dijkstra's algorithm in `shortest_path.py`. What is the shortest path from the top-left vertex $(0,0)$ to the bottom-right vertex $(59,32)$? Include your answer in your PDF. Also observe the following:

- For optional checks, use `dijkstra_greyscale_map_test.txt`. A visualization of the test map and a shortest path is shown below.
- Use Python's `PriorityQueue`. It does not directly support the lecture's `decrease-key` operation. Instead, insert the vertex again with its new distance.
- To plot your result, which is useful for testing, use the function provided in `gridToImg.py`.

¹The `main` function in `gridToImg.py` demonstrates how to read the graph.

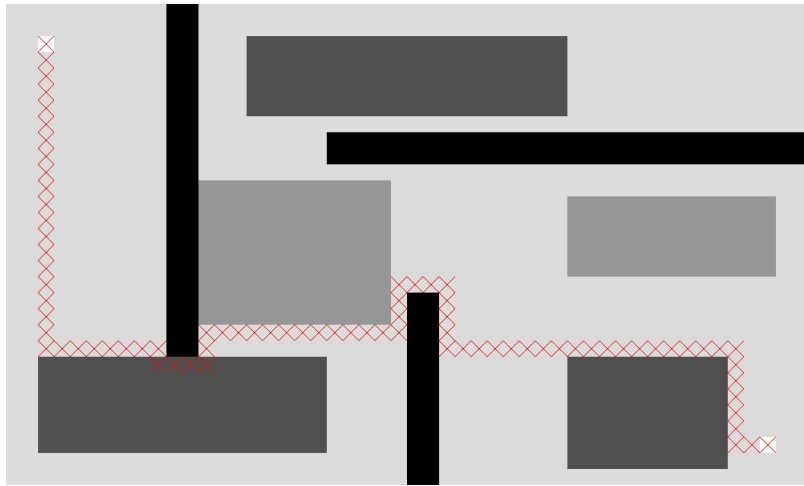
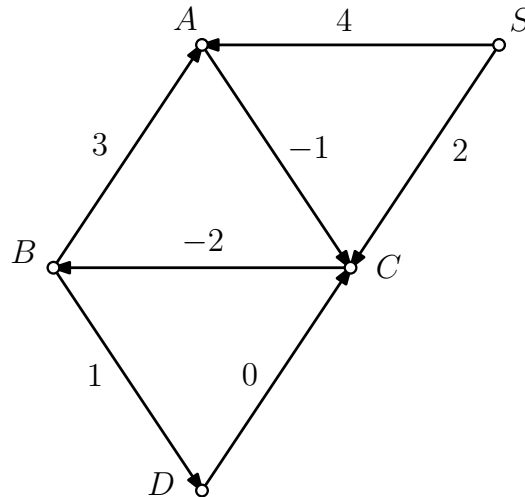


Figure 1: A shortest path in the test graph.

Exercise 2: Bellman-Ford

(6 Points)

Consider this directed graph with $n = 5$ vertices:



- (a) Does the graph contain a negative cycle? If so, mark the edges of one such cycle, for example by circling or coloring them. (1 Point)
- (b) Run Bellman-Ford from S . After *every* iteration of the outer loop, record the current distances from S in a table as below. (5 Points)

Important: In the inner loop, process edges in increasing order of weight, always in the order $-2, -1, 0, 1, 2, 3, 4$.

	$\delta(S, S)$	$\delta(S, A)$	$\delta(S, B)$	$\delta(S, C)$	$\delta(S, D)$
Initial	0	∞	∞	∞	∞
$i = 1$					
$i = 2$					
$i = 3$					
$i = 4$					