Exercise Sheet 03 - Solution Task 2

Task 1

Task 2 *Bellman-Ford Algorithm*

You go shopping and realize in the store that you only have 2.5 € with you. Using the Bellman-Ford algorithm, find an optimal selection of the following items so that you can buy them.

Item i	Price a_i	Benefit b_i
1: Bread	1.5 €	4
2: Banana	0.5 €	4
3: Cookies	1 €	3
4: Rice	1.5 €	5
5: Spinach	1 €	4

Solution 2

0 p.

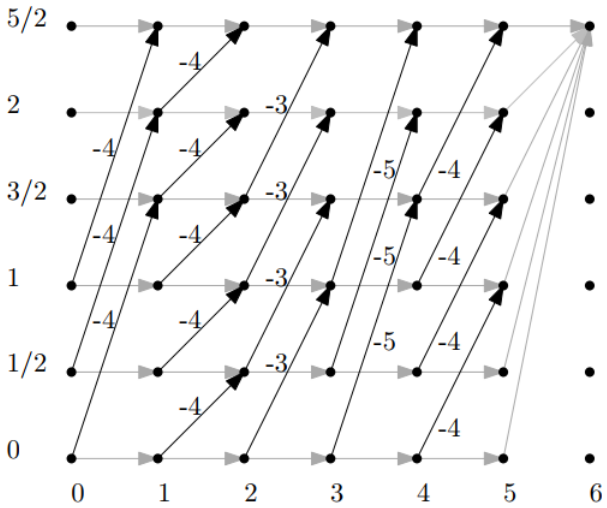
To apply the Bellman-Ford algorithm, we first model the problem as a shortest path problem on a graph. To do this, we use the numbering of the objects: 1 = bread, 2 = banana, ... ,5 = spinach and denote the volume of the i-th object by a_i and its benefit by b_i. As described in the lecture, the graph then has nodes of the form (i, x) with $i \in 0, 1, ..., 6$ and $x \in 0, \frac{1}{2}, 1, \frac{3}{2}, 2, \frac{5}{2}$. There are three varieties of edges:

1. Edges of the form $((i-1, x), (i, x))$ get length 0,
2. edges of the form $((i-1, x); (i, y))$ with $y - x = a_i$ have length $-b_i$,
3. edges of the form $((5, x), (6, \frac{5}{2}))$ have length 0.

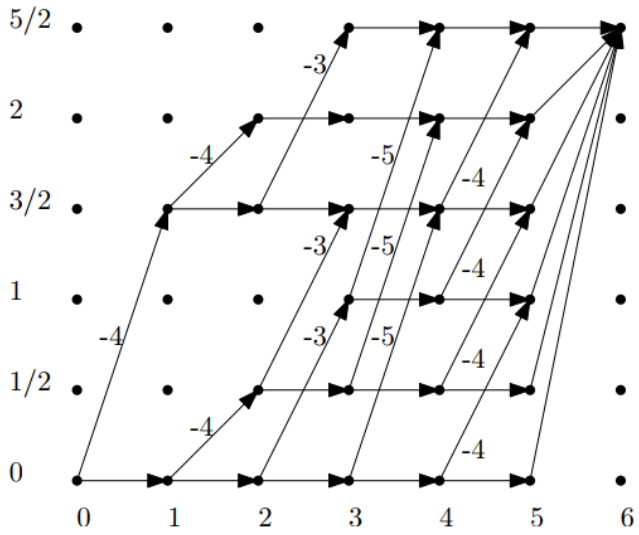
For better understanding, a short explanation of the resulting graph:
In the horizontal direction (x-axis) are the objects and in the vertical direction (y-axis) are the prices (or the "volume", if we are really talking about a knapsack). The edge weights correspond to the negative benefits. Simplified one can interpret the directed edges, after deletion of all redundant edges in such a way that one asks successively for all objects in each case whether they are taken along (arrow upward - 2.), or are not bought (arrow horizontally - 1.). Therefore all horizontal arrows have the weight 0. The 6th item or node in the upper right $(6, \frac{5}{2})$ is added to lead all possible paths to a destination (3.). The neagtive edge weights lead to the possibility to

search for a shortest (most useful) path, but has to switch to Bellman-Ford, because Dijkstra's algorithm does not work with negative weights. But since the resulting graph is circle-free, the Bellman-Ford in this particular example also expands only very similarly to the Dijkstra and not all repetitions, for all nodes are necessary, since there can be no further optimizations after reaching the target node.

We then obtain the following graph. (Edges of length 0 are drawn in gray, all other edges have their weight as label).



Shortest $(0,0)-(6,\frac{5}{2})$-paths in this graph correspond to optimal knapsack packings. Since we are only interested in $(0,0)-(6,\frac{5}{2})$-paths, we can ignore all edges that do not lie on such paths. That is, we can apply the Bellman-Ford algorithm (with $s=(0,0)$) to the following reduced graph:



For better understanding and distinction, we number the $dist_i(v)$ in the index after the passes of the loop and first set the $dist_0$ values as follows:
$dist_0((0,0)) = 0$, $dist_0((i,x)) = \infty$ for $i, x \neq 0$.
Then the for loop begins, determining the values $dist_{k+1}$ for each increasing parameter k, as well as the predecessors.

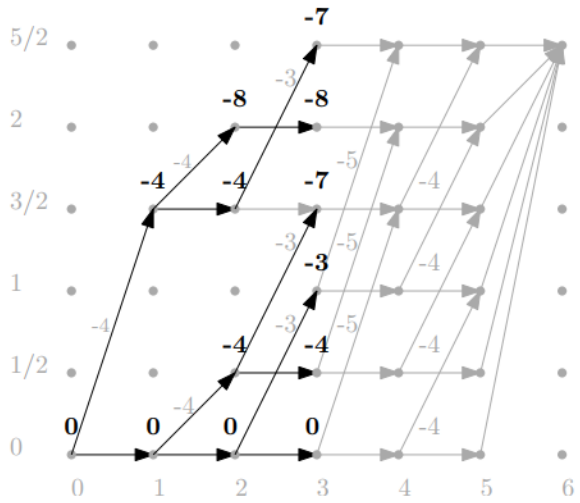
$k = 0$: After first setting $dist_1 = dist_0$, then for each edge $((i,x),(j,y))$ it is tested whether $dist_1((j,y)) > dist_0((i,x)) + \ell((i,x),(j,y))$. If this is the case, $dist_1((j,y)) = dist_0((i,x)) + \ell((i,x),(j,y))$ and $pred((j,y)) = (i,x)$ is set.
Since $\infty = dist_1((1,0)) > dist_0((0,0)) + \ell((0,0),(1,0)) = 0$, we get $dist_1((1,0)) = 0$ and $pred((1,0)) = (0,0)$. Similarly, we get $dist_1((1,\frac{3}{2})) = -4$ and $pred((1,\frac{3}{2}) = (0,0)$.
All other edges have as starting node a node (i,x) with $dist_0((i,x)) = \infty$, so the if-condition is not satisfied for any other edge. We obtain:
$dist_1((0,0)) = 0$, $dist_1((1,0)) = 0$, $dist_1((1,\frac{3}{2})) = -4$, $dist_1((i,x)) = \infty$ otherwise.
Moreover, we have $pred((1,0)) = (0,0)$ and $pred((1,\frac{3}{2})) = (0,0)$.

$k = 1$: As in the first step ($k = 0$), in this step all nodes under consideration are reached by exactly one edge, which determines the new value of the node is determined. It results in: $dist_2((0,0)) = 0$, $dist_2((1,0)) = 0$, $dist_2((1,\frac{3}{2})) = -4$, $dist_2((2,0)) = 0$, $dist_2((2,\frac{3}{2})) = -4$, $dist_2((2,\frac{1}{2})) = -4$, $dist_2((2,2)) = -8$, $dist_2((i,x)) = 1$ otherwise. Moreover, we get $g((2,0)) = g((2,\frac{1}{2})) = (1,0)$ and $g((2,\frac{3}{2})) = g((2,2)) = (1,\frac{3}{2})$.

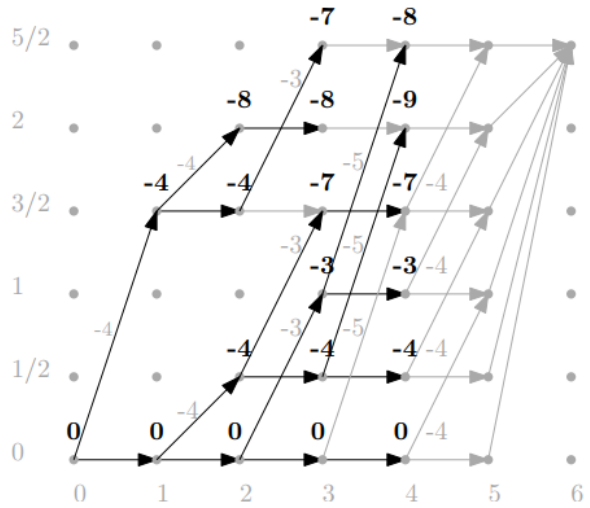
$k = 2$: Except for the node $(3,\frac{3}{2})$, all nodes are again reached by exactly one

edge. For node $(3, \frac{3}{2})$, both edges $((2, \frac{1}{2}), (3, \frac{3}{2}))$ and $((2, \frac{3}{2}), (3, \frac{3}{2}))$ are tested. Since $dist_2((2, \frac{1}{2})) + \ell((2, \frac{1}{2}), (3, \frac{3}{2})) = -4 + (-3) = -7$ gives the smaller value than $dist_2((2, \frac{3}{2})) + \ell((2, \frac{3}{2}), (3, \frac{3}{2})) = -4 + 0 = -4$, the first of the two edges determines the value of $(3, \frac{3}{2})$.

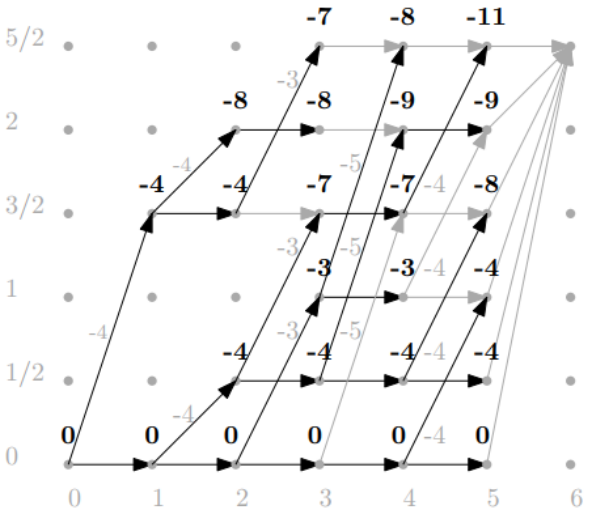
We get:
$dist_3((0,0)) = 0$,
$dist_3((1,0)) = 0, dist_3((1, \frac{3}{2})) = -4$,
$dist_3((2,0)) = 0, dist_3((2, \frac{3}{2})) = -4$,
$dist_3((2, \frac{1}{2})) = -4, dist_3((2,2)) = -8$,
$dist_3((3,0)) = 0, dist_3((3, \frac{3}{2})) = -7$,
$dist_3((3,12)) = -4, dist_3((3,2)) = -8$,
$dist_3((3,1)) = -3$, $dist_3((3, \frac{5}{2}) = -7$,
$dist_3((i,x)) = 1$ otherwise:
Moreover, we have $pred((3,0)) = pred((3,1)) = (2,0)$, $pred((3, \frac{1}{2})) = pred((3, \frac{3}{2})) = (2,12)$, $pred((3, \frac{5}{2})) = (2, \frac{3}{2})$, and $pred((3,2)) = (2,2)$. We record the result pictorially. The value of $dist_3$ of a node stands next to the node if it is not 1. The edges determined by the function g edges are black.



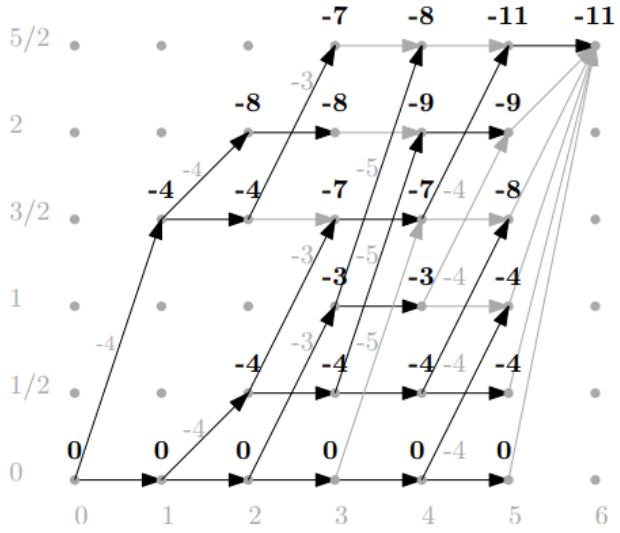
$k = 4$: The bottom three nodes $(4,0)$, $(4, \frac{1}{2})$ and $(4,1)$ are again reached by only one edge. The other three nodes are reached by two edges each. For these, both possibilities are tested and the better one is recorded. ¨ This time we describe the result only pictorially:



$k = 5$: The lower two nodes $(5, 0)$ and $(5, \frac{1}{2})$ are again reached by only one edge. The other four nodes are reached by two edges each. For these, both possibilities are tested and the better one is recorded. We describe the result again only pictorially:



$k = 5$: Here we consider only the node $(6, \frac{5}{2})$, which is reached by six edges, each of length 0. Thus, the shortest path is provided by the edge whose starting node has the smallest value. This is the node $(5, \frac{5}{2})$. The result figuratively:



The for loop still runs until $k = 34$. However, the values of $dist_k$ do not change anymore. So this is the final result. We obtain the optimal choice of items for our knapsack by tracing back the (black) path leading to node $(6, \frac{5}{2})$ and packing the item corresponding to the terminal node for each non-horizontal edge: Banana, Cookies, and Spinach. Together these objects have a utility of 11 and a price of 2.5.