The Traveling Salesperson Problem without Repetitions (TSP-NR) is an NP-hard combinatorial optimization problem defined as follows: Given the distances between n cities, return the cheapest tour that visits each of them once.

The Traveling Salesperson Problem with Repetitions (TSP-R) is a generalization of the former. It stipulates that for each city the range between minimum and maximum allowed number of visits can be specified (otherwise the default value is exactly one visit).

Dynamic Programming (DP) is a mathematical and computer engineering method used mainly for solving discrete optimization problems by first solving subproblems and storing their solutions and then combining the latter into a complete solution.

Answer Set Programming (ASP) is a form of fact-driven declarative programming that is based on the semantics of stable models. ASP is highly maintainable and flexible, however often with the price of a higher running time and memory consumption.

The motivation for our work derives from the need to easily prototype dynamic programming algorithms for relevant problems, such as the TSP, without losing on efficiency.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

It decomposes the input graph into a tree, such that each edge and each vertex of the graph must belong to at least one node of the decomposition. Further, if a vertex belongs to two different nodes, it must belong to all nodes on the path between the former.

At every node, D-FLAT executes an ASP program, which is tailored to the problem to be solved, stores the results for each node in a table and finally recomposes the optimal solutions by looking up all tables.

The goals were to:

- Develop a dynamic programming concept for the TSP.
- Prove the flexibility and efficiency of D-FLAT by encoding the TSP-NR and TSP-R based on the developed concept.
- Show that they are more performant than state-of-the-art ASP encodings on instances with small treewidth, the latter denoting a limited cyclicity of a graph.
- The memory consumption is also lower with D-FLAT for treewidths 2 & 3.
- The more vertices must be visited, the faster our encoding becomes.

For every newly introduced vertex (such as vertex 3 in node e of the tree decomposition in Figure 2) we guess whether to select the edges that lead to an adjacent current vertex (0 and 2) to be part of the tour.

If in a join node (here node b) the edge selection on the left branch does not correspond to the one on the right, we discard the solution candidate.

We keep a counter for the selected adjacent edges of each current vertex.

When a vertex was removed (e.g. vertex 1 in node e), we check the counter. If it is different from 2 for the TSP-NR, or not in the specified range times 2 for the TSP-R, we eliminate the solution candidate.

We memorize for each pair of current vertices (e.g. 0, 2, 3 in node e) if it is connected by a path.

If at removal, a vertex is not connected to any other current vertex we cannot obtain a connected tour and the candidate is discarded.

In order to comply to the principle of dynamic programming we always work only on vertices that are in the current node.

When traversing the tree, we do the following for every tree node using ASP:

- For every newly introduced vertex (such as vertex 3 in node e of the tree decomposition in Figure 2) we guess whether to select the edges that lead to an adjacent current vertex (0 and 2) to be part of the tour.
- If in a join node (here node b) the edge selection on the left branch does not correspond to the one on the right, we discard the solution candidate.
- We keep a counter for the selected adjacent edges of each current vertex.
- When a vertex was removed (e.g. vertex 1 in node e), we check the counter. If it is different from 2 for the TSP-NR, or not in the specified range times 2 for the TSP-R, we eliminate the solution candidate.
- We memorize for each pair of current vertices (e.g. 0, 2, 3 in node e) if it is connected by a path.
- If at removal, a vertex is not connected to any other current vertex we cannot obtain a connected tour and the candidate is discarded.

The motivation for our work derives from the need to easily prototype dynamic programming algorithms for relevant problems, such as the TSP, without losing on efficiency.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

The D-FLAT System, developed by the dbai Group, both promises flexibility and handles large amounts of data efficiently.

In order to comply to the principle of dynamic programming we always work only on vertices that are in the current node.

When traversing the tree, we do the following for every tree node using ASP:

- For every newly introduced vertex (such as vertex 3 in node e of the tree decomposition in Figure 2) we guess whether to select the edges that lead to an adjacent current vertex (0 and 2) to be part of the tour.
- If in a join node (here node b) the edge selection on the left branch does not correspond to the one on the right, we discard the solution candidate.
- We keep a counter for the selected adjacent edges of each current vertex.
- When a vertex was removed (e.g. vertex 1 in node e), we check the counter. If it is different from 2 for the TSP-NR, or not in the specified range times 2 for the TSP-R, we eliminate the solution candidate.
- We memorize for each pair of current vertices (e.g. 0, 2, 3 in node e) if it is connected by a path.
- If at removal, a vertex is not connected to any other current vertex we cannot obtain a connected tour and the candidate is discarded.