Heuristic algorithm for the Generalized Elementary Shortest Path Problem

Guerrero, W.J.¹,², Velasco, N.,³, Prodhon, C.,² Amaya, C.A.¹

¹ Pylo, School of Engineering, Universidad de los Andes
    Cra 1 Este No 19A - 40, Bogotá, Colombia.
    {wi-guerr, nvelasco, ca.amaya}@uniandes.edu.co

² ICD-LOSI (UMR-STMR CNRS 6279), Université de Technologie de Troyes
    12 rue Marie Curie, CS 42060 10004 Troyes Cedex, France
    {guerrerw, caroline.prodhon}@utt.fr

³ Invited Professor, ICD-LOSI (UMR-STMR CNRS 6279), Université de Technologie de Troyes
    12 rue Marie Curie, CS 42060 10004 Troyes Cedex, France

Mots-clés: Elementary shortest path problem, labeling algorithm, dynamic programming

1 Problem Description and Heuristic Algorithm

We introduce the generalized elementary shortest path problem (GESPP). It is defined over a complete weighted graph $G$ composed by a set $J$ of $n$ nodes, a source node $\{0\}$ and a sink node $\{n+1\}$. Each arc in $A = \{(i, j), \forall i, j \in J \cup \{0, n+1\}\}$ in the graph is associated to a cost $c_{ij} \in \mathbb{R}$. In consequence, the graph may contain negative cycles. The GESPP is a variant of the shortest path problem where nodes are aggregated in predefined non-disjoint clusters. Each cluster $k \in \Psi$ is associated with a profit $p_k \geq 0$ to the cost function if at least one node in $k$ is visited. Depending on the application, clusters could be interpreted as groups of nodes with linking features, easily reachable from each other, or some other kind of coverage guarantee.

Figure 1 presents the graph for an example of the GESPP considering $n = 9$. Nodes 0 and 10 are the source and sink respectively. Consider four clusters: $C_1 = \{1, 2, 4, 5\}$, $C_2 = \{2, 3\}$, $C_3 = \{7, 8\}$, and $C_4 = \{5, 6, 8, 9\}$. The path $\{0 - 2 - 4 - 7 - 10\}$ would have a cost equal to $c_{0,2} + c_{2,4} + c_{4,7} + c_{7,10} - p_1 - p_2 - p_3$.

Similar problems in the literature have been studied. The median shortest path problem (MSPP) [1], the median cycle problem (MCP) [3], and the TSP with profits [2] are reviewed in comparison to the GESPP. Applications could include CVRP pricing problems, transportation network design for new bus line, where two points in the city are to be connected while deserving a set of bus stops, or a telecommunication network design.

Let $x_{ij}$ be a binary decision variable indicating if the arc $(i, j)$ belongs to the path, and $y_k$ be a binary variable equal to 1 iff cluster $k \in \Psi$ is visited at least once. Let the GESPP be:

GESPP:  
$$\min \sum_{i \in J \cup \{0\}} \sum_{j \in J \cup \{n+1\}} c_{ij} x_{ij} - \sum_{k \in \Psi} y_k p_k$$  (1)
Subject to:

\[ \sum_{i \in J} x_{0,i} = 1 \quad (2) \]

\[ \sum_{i \in J} x_{i,n+1} = 1 \quad (3) \]

\[ \sum_{i \in J} x_{i,j} - \sum_{i \in J \cup \{0\}} x_{ji} = 0, \forall j \in J \quad (4) \]

\[ \sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \forall S \subseteq J \quad (5) \]

\[ \sum_{i \in S} \sum_{j \in J(S_k)} x_{ij} \geq y_k, \forall k \in \Psi \quad (6) \]

\[ x_{ij} \in \{0, 1\} \forall i \in J \cup \{0\}, \forall j \in J \cup \{n + 1\}, y_k \in \{0, 1\} \forall k \in \Psi \quad (7) \]

The objective is defined by equation (1) aiming to minimize the result of the path length cost after subtracting the cluster profits. Constraints (2), (3), and (4) are traditional to a shortest path problem. Equations (5) eliminate subtours. A cluster profit is obtained iff the path visits any node belonging to the corresponding cluster as stated by constrains (6). Variables are binary as defined by equations (7).

To find the shortest path, a label correcting heuristic algorithm is proposed. Each retailer \( j \) is associated with a set of labels representing paths from node 0 to \( j \). Each label \( L_j' \) keeps track of all visited clusters in the set \( S_{L_j'} \subseteq \Psi \), the visited retailers, and the path cost \( C(L_j') \). One label \( L_j' \) dominates another label \( L_j'' \), denoted as \( L_j' \prec L_j'' \), only if: 1) \( L_j' \) and \( L_j'' \) represent paths from node 0 to the same node \( j \); 2) if \( C(L_j') \leq C(L_j'') \); and 3) \( S_{L_j''} \subseteq S_{L_j'} \) implying that \( L_j' \) visits at least all the clusters visited by \( L_j'' \). Dominated labels are erased.

The algorithm enumerates the paths from 0 up to every node in the graph keeping only non-dominated labels. To speed up the search, a limit of \( K \) labels per node is imposed. The algorithm stops when all the existing labels have being extended to unvisited nodes. A local search procedure is performed as post-optimization.

## 2 Computational experiments

Tests for a set of 100 random instances with up to 100 nodes are run on an Intel Xeon with 2.80Ghz processor and 12 GB of RAM. As expected, for low values of \( K \), the algorithm requires extensive intensification in the post-optimization phase while larger values makes the algorithm to perform slowly in the first phase. The best found solution (UB) is obtained by running the algorithm with different \( K \) values up to 35. A lower bound (LB) is computed using a MIP solver if subtour elimination constraints are relaxed. Results show an average computation time (CPU) of 3.98s, a gap to UB of 0.2%, and an average gap to LB of 3.4% for a \( K = 4 \) providing the best trade-off between solution quality and CPU. Future research involves exact methodologies for the problem and improved lower bounds.

## References

