Journal of Discrete Algorithms 46–47 (2017) 16–24

Contents lists available at ScienceDirect

Journal of Discrete Algorithms

www.elsevier.com/locate/jda

New relationships for multi-neighborhood search for the minimum linear arrangement problem

Fred Glover [a], César Rego [b,*]

[a] ECEE, College of Engineering and Applied Science, University of Colorado, Boulder, CO 80309, USA
[b] School of Business Administration, University of Mississippi, University, MS 38677, USA

ARTICLE INFO

Article history:
Received 20 September 2016
Received in revised form 4 October 2017
Accepted 5 October 2017
Available online 14 October 2017

Keywords:
Multi-neighborhood search
Metaheuristics
Minimum linear arrangement
Exponential neighborhoods
Combinatorial leverage

ABSTRACT

We develop a series of theorems about the graph structure of the classical Minimum Linear Arrangement (MinLA) problem which disclose properties that can be exploited by Multi-Neighborhood Search (MNS) algorithms. As a foundation, we differentiate between swaps of labels attached to adjacent and non-adjacent nodes to create two new neighborhood classes, and show how our theorems yield efficient algorithms for updating key arrays used by local search procedures. In addition, we introduce a class of neighborhoods called set-based neighborhoods supported by a theorem that identifies solutions (labelings) for the MinLA problem in polynomial time that dominate exponential numbers of alternative solutions. The component neighborhoods within this new neighborhood class can be applied in various sequences in conjunction with the first two new neighborhoods introduced. Our results also apply to problems with objectives different than those of MinLA. Finally, our results make it possible to exploit the new neighborhoods according to the user's choice of MNS protocols and alternative local search algorithms.

1. Introduction

The Minimum Linear Arrangement (MinLA) Problem has had a long history as a challenging problem in combinatorial optimization. First stated by Harper [8], the MinLA problem has been studied through the years using a wide range of solution algorithms. The problem was demonstrated to be NP-hard by Garey and Johnson [2] and subsequently shown to be related to two other well-known layout problems, the bandwidth and profile minimization problems. However, as pointed out by McAllister [11], an optimal solution for one of these problems is not necessarily optimal for either of the other related problems.

One of the attractions of the MinLA problem is the fact that it can be formulated as a problem of labeling nodes of a graph, thus giving it an appealing graph theoretic foundation. Owing also to its prominence as a layout problem, a significant number of solution algorithms have been proposed for MinLA. Some of the more notable methods introduced since 1990 include a Sequential Spectral Sequencing (SSQ) method by Juvan and Mohar [9] based on computing eigenvectors of the Laplacian matrix of the graph, a sequential ordering heuristic by McAllister [11] that constructs solutions based on choosing nodes to label according to their degree with respect to previously labeled vertices, a graph-based approach using the Gomory–Hu tree method by Adolphson and Hu [1], a combined SSQ/simulated annealing algorithm by Petit [12], a multilevel algorithm by Safro, Ron and Brandt [14], and an enhanced two stage simulated annealing algorithm by

* Corresponding author.
E-mail addresses: fred.glover@colorado.edu (F. Glover), crego@bus.olemiss.edu (C. Rego).

https://doi.org/10.1016/j.jda.2017.10.003
1570-8667/

Rodríguez-Tello et al. [13] that has been shown to outperform previous algorithms. Quite recently a new algorithm by Martí et al. [10] incorporating scatter search and path relinking [3,6] yields superior outcomes in terms of both execution time and the number of best solutions obtained.

The contribution of this paper departs from the orientation underlying previous leading approaches for the MinLA problem by developing theorems based on the MinLA graph structure that identify new search neighborhoods and ways they can be exploited efficiently by Multi-Neighborhood Search (MNS) algorithms.

In contrast to the work of Rodríguez-Tello et al. [13] and Martí et al. [10], our results do not rely on numerical properties of the objective function (proximities of labels to medians of the labels of adjacent nodes), which are highly specific to the MinLA problem. Hence our theorems are valid for more general classes of problems that include the MinLA problem as a special case. Specifically, we take advantage of the adjacency and non-adjacency differentiation of nodes in the problem graph structure to give an effective means for exploiting these neighborhoods in a general setting. Then, we introduce a class of additional neighborhoods for the MinLA problem called set-based neighborhoods, and show how to identify an optimal solution over each of the neighborhoods in this class. Once again, our results do not rely on numerical properties of the MinLA objective function and thus can be applied in broader contexts. Finally, we provide a mechanism for transitioning among neighborhoods in the set-based class by means of a frequency-based critical event memory. The results justifying these neighborhoods provide a foundation for implementations of MNS methods both for MinLA and for variations that incorporate other objectives.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation. Sections 3 and 4 present the proposed new neighborhood classes and associated theoretical results, and Sect. 5 presents conclusions and opportunities for future research.

2. Problem formulation

Let $G = (N, E)$ denote a graph with node set $N = \{1, \ldots, n\}$ and edge set E consisting of a subset of (unordered) pairs $(i, j),\ i, j, \in N$.

A linear layout or labeling L of G assigns the integers $1, 2, \ldots, n$ to the nodes of G so that each node has a different label. Let $L(i)$ be the label of node i, and let $N(i)$ denote the set of nodes that are neighbors of (adjacent to) node i, hence $N(i) = \{j : (i, j) \in E\}$.

The contribution of node i to the objective function under the labeling L, which we call the value of node i, is given by

$$Val(i) = \sum_{k \in N(i)} \big|L(i) - L(k)\big|$$

The objective function value for the MinLA problem on the graph G, denoted $GVal$, is then

$$GVal = \frac{1}{2} \sum_{i \in N} Val(i) \tag{2.1}$$

or equivalently

$$GVal = \sum_{(i,j) \in E} \big|L(i) - L(j)\big| \tag{2.2}$$

The goal of the MinLA problem is to minimize *GVal* by choosing a "best assignment" of the labels to the nodes. We take the labeling L to be implicit in the preceding definitions of $Val(i)$ and $GVal$, understanding that these terms may be expressed more explicitly as $Val(L : i)$ and $GVal(L)$.

3. Foundations for a class of parameter-based neighborhoods

In this section, we identify a series of theorems concerning relationships implied by the graph structure that can be exploited by higher level metaheuristic guidance and special updating algorithms. Our results concern a parameter-based class of neighborhoods consisting of two types of neighborhoods that swap labels attached to a pair of selected nodes i and j depending on whether i and j are adjacent. Complementing this, in the next section we introduce a set-based class of neighborhoods consisting of multiple neighborhoods defined by selecting subsets N_0 of non-adjacent nodes and identifying an optimal re-assignment of their labels in polynomial time by solving an assignment problem, creating a labeling that dominates $|N_0|!$ alternatives. Our results for exploiting the structure of the graph G are particularly useful for MNS strategies that employ adaptive memory.

As a starting point we introduce two key quantities. For any two distinct nodes i and j, let $TransVal(i, j)$ denote the contribution to the objective function $GVal$ if the label for node i is reassigned ("transferred") to node j, given by

$$TransVal(i, j) = \left(\sum_{k \in N(j) - \{i\}} \big|L(i) - L(k)\big| \right) + \delta_{ij} \big|L(i) - L(j)\big| \tag{3.1}$$

where $\delta_{ij} = 1$ if nodes i and j are neighbors, and $\delta_{ij} = 0$ otherwise.

Thus $TransVal(i, j)$ identifies the quantity $Val(j)$ that would result if nodes i and j in G swap labels, causing $Val(j)$ to be replaced by $TransVal(i, j)$ in the definition of the objective function $GVal$ under this swap. Similarly $TransVal(j, i)$ identifies the quantity $Val(i)$ that would result if nodes i and j in G swap labels.

Building on (3.1) we define

$$SwapVal(i, j) = TransVal(i, j) + TransVal(j, i) - \big(Val(i) + Val(j)\big). \tag{3.2}$$

The choice of the name $SwapVal(i, j)$ is intended to suggest that (3.2) identifies the "swap value" embodied in the change in $GVal$ that results when the labels of nodes i and j are exchanged. We now show that this assertion is true, in spite of the fact that (3.2) contains no reference to the factor 1/2 in the expression (2.1), and also in spite of the fact that the swap makes different changes in $Val(i)$ and $Val(j)$ depending on whether i and j are neighbors.

Observation 1. $SwapVal(i, j)$ equals the change in $GVal$ produced by swapping the labels for nodes i and j.

Proof. By reference to the equivalent definition (2.2) of $GVal$, the combined contribution of $Val(i)$ and $Val(j)$ to the objective function before the swap can be expressed as $Val(i) + Val(j) - \delta_{ij}|L(i) - L(j)|$. This follows from the fact that when nodes i and j are neighbors, the edge (i, j) gets counted twice in $Val(i) + Val(j)$, whereas otherwise $\delta_{ij} = 0$ and the extra term vanishes. By the same token $TransVal(i, j)$ and $TransVal(j, i)$ contribute $TransVal(i, j) + TransVal(j, i) - \delta_{ij}|L(i) - L(j)|$ to $GVal$ after the swap. The quantity in (3.2) results directly by subtracting the former contribution from the latter.

The foregoing observation can also be inferred from discussions in Martí et al. [10]. It is useful for our purposes to additionally note that $SwapVal(i, j)$ is symmetric, that is, $SwapVal(i, j) = SwapVal(j, i)$, whereas in general $TransVal(i, j) \neq TransVal(j, i)$. (Swapping $L(i)$ and $L(j)$ is no different from swapping $L(j)$ and $L(i)$, but the new $Val(j) = TransVal(i, j)$ will generally differ from the new $Val(i) = TransVal(j, i)$.)

Among other things we are interested in an effective way of exploiting the graph structure to permit $SwapVal(i, j)$ as defined by (3.2) to be identified and updated highly efficiently. We will show how this can be done in the case where (i, j) is an edge of the graph, and demonstrate how this can be integrated with a critical event memory defined over edges of G to create a higher level search strategy for neighbor swaps. At the same time, we will show how to exploit partial information of non-neighboring nodes integrated with critical event memory defined over nodes of G. In this fashion, we create two alternating and interacting neighborhood search processes that form the foundation of a multi-neighborhood search approach, in one case focusing on neighbor swaps and in the other case focusing on non-neighbor swaps, which together drive the overall method to explore the solution space in an effective manner.

For the following results we address the situation where two nodes p and q are selected as a basis for swapping their labels, and assume $L(p)$ and $L(q)$ denote the values of these labels before the swap takes place. All other labels $L(i)$, $i \in N - \{p, q\}$ are assumed to refer to the corresponding values before changing any labels in the current solution.

We make use of the following definitions:

- $V(i) =$ Value of $Val(i)$ before swapping nodes p and q.
- $NewV(i) =$ Value of $Val(i)$ after swapping nodes p and q.
- $V(i : j) = TransVal(i : j)$, the value of $Val(i)$ when nodes i and j swap labels, but before swapping labels for nodes p and q.
- $NewV(i : j) = V(i : j)$ after swapping labels for nodes p and q.
- $SwapVal(i, j) =$ the change in the objective function by swapping the labels for nodes i and j before swapping the labels for nodes p and q.
- $NewSwapVal(i, j) =$ the change in the objective function by swapping the labels for nodes i and j after swapping the labels for nodes p and q.
- $CngV(i) = NewV(i) - V(i)$.
- $CngV(i : j) = NewV(i : j) - V(i : j)$.
- $CngSwapVal(i, j) = NewSwapVal(i, j) - SwapVal(i, j)$.

Defining δ_{ij} as before, where $\delta_{ij} = 1$ if nodes i and j are neighbors, and $\delta_{ij} = 1$ otherwise, we may summarize the result of the preceding definitions in the following expressions.

$$V(i) = \sum_{k \in N(i)} \big|L(i) - L(k)\big|$$

$$V(i : j) = \Bigg(\sum_{k \in N(j) - \{i\}} \big|L(i) - L(k)\big|\Bigg) + \delta_{ij}\big|L(i) - L(j)\big|$$

For convenience we also express the Basic Observation in the form:

Observation 2.

$$SwapVal(i, j) = V(i : j) + V(j : i) - \big(V(i) + V(j)\big),$$

and note that the observation implies:

$$NewSwapVal(i, j) = NewV(i : j) + NewV(j : i) - \big(NewV(i) + NewV(j)\big)$$
$$CngSwapVal(i, j) = CngV(i : j) + CngV(j : i) - \big(CngV(i) + CngV(j)\big)$$

The goal is to identify $(i : j)$, $CngV(j : i)$, $CngV(i)$ and $CngV(j)$, which will therefore give the updated form of $SwapVal(i, j)$ after swapping labels for p and q:

$$NewSwapVal(i, j) = SwapVal(i, j) + CngV(i : j) + CngV(j : i) - \big(CngV(i) + CngV(j)\big)$$

Different outcomes apply to different cases.

Theorem 1. *If $i \neq p, q$*

Case 1: $CngV(i) = 0$ *if* $i \in N(p) \cap N(q)$
Case 2: $CngV(i) = |L(i) - L(q)| - |L(i) - L(j)|$ *if* $i \in N(p) - N(q)$
Case 3: $CngV(i) = |L(i) - L(p)| - |L(i) - L(q)|$ *if* $i \in N(q) - N(p)$
Case 4: $CngV(i) = 0$ *if* $i \notin N(p) \cup N(q)$

Proof. From the definition of $V(i)$ we can write

$$V(i) = \Bigg(\sum_{k \in N(i) - \{p,q\}} \big|L(i) - L(k)\big|\Bigg) + \delta_{ip}\big|L(i) - L(p)\big| + \delta_{iq}\big|L(i) - L(q)\big|$$

After swapping $L(p)$ and $L(q)$ for nodes p and q we have $|L(i) - L(p)| \to |L(i) - L(q)|$ and $|L(i) - L(q)| \to |L(i) - L(p)|$, and consequently, assuming $i \neq p, q$ (so that $|L(i) - L(k))|$ does not change)

$$NewV(i) = \Bigg(\sum_{k \in N(i) - \{p,q\}} \big|L(i) - L(k)\big|\Bigg) + \delta_{ip}\big|L(i) - L(q)\big| + \delta_{iq}\big|L(i) - L(p)\big|$$

This yields

$$\begin{aligned} CngV(i) &= \delta_{ip}\big|L(i) - L(q)\big| + \delta_{iq}\big|L(i) - L(p)\big| - \big(\delta_{ip}\big|L(i) - L(p)\big| + \delta_{iq}\big|L(i) - L(q)\big|\big) \\ &= \delta_{ip}\big(\big|L(i) - L(q)\big| - \big|L(i) - L(p)\big|\big) + \delta_{iq}\big(\big|L(i) - L(p)\big| - \big|L(i) - L(q)\big|\big) \end{aligned}$$

An enumeration of cases gives the result of the Theorem, according to the possible values of δ_{ip} and δ_{iq}. □

Theorem 2. *When p and q are neighbors*

$$NewV(p) = \Bigg(\sum_{i \in N(p)} \big|L(q) - L(i)\big|\Bigg) + \big|L(q) - L(p)\big|$$

$$NewV(q) = \Bigg(\sum_{i \in N(q)} \big|L(p) - L(i)\big|\Bigg) + \big|L(p) - L(q)\big|$$

and when p and q are not neighbors

$$NewV(p) = \sum_{i \in N(p)} \big|L(q) - L(i)\big|$$

$$NewV(q) = \sum_{i \in N(q)} \big|L(p) - L(i)\big|$$

Proof. Write $V(p)$ in the form

$$V(p) = \Bigg(\sum_{i \in N(p) - \{q\}} \big|L(p) - L(i)\big|\Bigg) + \delta_{pq}\big|L(p) - L(q)\big|$$

This implies, after swapping labels for nodes p and q:

$$NewV(p) = \Big(\sum_{i \in N(p)-\{q\}} \big|L(q) - L(i)\big| \Big) + \delta_{pq} \big|L(p) - L(q)\big|$$

If p and q are not neighbors, this reduces to

$$NewV(p) = \sum_{i \in N(p)} \big|L(q) - L(i)\big|$$

While if they are neighbors, it reduces to

$$NewV(p) = \Big(\sum_{i \in N(p)} \big|L(q) - L(i)\big| \Big) + \big|L(p) - L(q)\big|$$

since allowing $i \in N(p)$ in place of $i \in N(p) - \{q\}$ only adds the term $L(p, q) = 0$. □

Theorem 3. *If $i \neq p, q$ and $j \neq p, q$, then*

Case 1 $(i : j)$: $CngV(i : j) = 0$ *if* $j \in N(p) \cap N(q)$
Case 2 $(i : j)$: $CngV(i : j) = |L(i) - L(q)| - |L(i) - L(j)|$ *if* $j \in N(p) - N(q)$
Case 3 $(i : j)$: $CngV(i : j) = |L(i) - L(p)| - |L(i) - L(q)|$ *if* $j \in N(q) - N(p)$
Case 4 $(i : j)$: $CngV(i : j) = 0$ *if* $j \notin N(p) \cup N(q)$

Proof. Rewrite $V(i : j)$ in the form

$$V(i : j) = \Big(\sum_{k \in N(j)-\{i,p,q\}} \big|L(i) - L(k)\big| \Big) + \delta_{ij} \big|L(i) - L(j)\big| + \delta_{ip} \big|L(i) - L(p)\big| + \delta_{jq} \big|L(i) - L(q)\big|$$

After swapping label $L(p)$ and label $L(q)$ for nodes p and q we have

$$NewV(i : j) = \Big(\sum_{k \in N(j)-\{i,p,q\}} \big|L(i) - L(k)\big| \Big) + \delta_{ij} \big|L(i) - L(j)\big| + \delta_{ip} \big|L(i) - L(q)\big| + \delta_{jq} \big|L(i) - L(p)\big|$$

Hence

$$\begin{aligned}
CngV(i : j) &= L_{ij}(i, j) + L_{jp}(i, q) + L_{jq}(i, p) - \big(L_{ij}(i, j) + L_{jp}(i, p) + L_{jq}(i, q)\big) CngV(i : j) \\
&= \delta_{ij} \big|L(i) - L(j)\big| + \delta_{jp} \big|L(i) - L(q)\big| + \delta_{jq} \big|L(i) - L(p)\big| \\
&\quad - \big(\delta_{ij} \big|L(i) - L(j)\big| + \delta_{jp} \big|L(i) - L(p)\big| + \delta_{jq} \big|L(i) - L(q)\big|\big) \\
&= \delta_{jp} \big|L(i) - L(q)\big| + \delta_{jq} \big|L(i) - L(p)\big| - \big(\delta_{jp} \big|L(i) - L(p)\big| + \delta_{jq} \big|L(i) - L(q)\big|\big) \\
&= \delta_{jp} \big(\big|L(i) - L(q)\big| - \big|L(i) - L(p)\big|\big) + \delta_{jq} \big(\big|L(i) - L(p)\big| - \big|L(i) - L(q)\big|\big)
\end{aligned}$$

An enumeration of cases gives the result of the Theorem, according to the four possible combinations of values for δ_{jp} and δ_{jq}. □

Theorem 4. *If $i \neq p, q$ and $i \in N(p)$:*

$$CngV(p : i) = \Big(\sum_{j \in N(i)-\{p,q\}} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \Big) + \big|L(i) - L(q)\big| - \big|L(i) - L(p)\big|$$

Moreover, if $L(p)$ is temporarily reset by $L(p) = L(i)$, restricted to encountering p in the set of nodes $j \in N(i)$ then

$$CngV(p : i) = \Big(\sum_{j \in N(i)} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \Big) + \delta_{iq} \big|L(p) - L(q)\big|$$

Proof. Write $V(p : i)$ in the form:

$$V(p : i) = \Big(\sum_{j \in N(i)-\{p,q\}} \big|L(p) - L(j)\big| \Big) + \delta_{ip} \big|L(i) - L(p)\big| + \delta_{iq} \big|L(p) - L(q)\big|$$

Since i and p are neighbors, we have:

$$V(p:i) = \Big(\sum_{j \in N(i) - \{p,q\}} \big|L(p) - L(j)\big| \Big) + \big|L(i) - L(p)\big| + \delta_{iq} \big|L(p) - L(q)\big|$$

After swapping labels for p and q this gives

$$NewV(p:i) = \Big(\sum_{j \in N(i) - \{p,q\}} \big|L(q) - L(j)\big| \Big) + \big|L(i) - L(q)\big| + \delta_{iq} \big|L(p) - L(q)\big|$$

Consequently

$$CngV(p:i) = \Big(\sum_{j \in N(i) - \{p,q\}} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \Big) + \big|L(i) - L(q)\big| - \big|L(i) - L(p)\big|$$

which is the first result identified in the theorem. Now if we set $L(p) = L(i)$, restricted to accessing p as one of the elements $j \in N(i)$, then the expression

$$\sum_{j \in N(i) - \{q\}} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \tag{3.3}$$

can be written

$$\Big(\sum_{j \in N(i) - \{p,q\}} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \Big) + \big|L(q) - L(i)\big| - \big|L(p) - L(i)\big|$$

and the latter is identical to the value derived for $CngV(p:i)$. Moreover, expression (3.3) is the same as

$$\Big(\sum_{j \in N(i)} \big(\big|L(q) - L(j)\big| - \big|L(p) - L(j)\big|\big) \Big) - \delta_{iq} \big(\big|L(q) - L(q)\big| - \big|L(p) - L(q)\big|\big)$$

and upon dropping the 0 term $|L(q) - L(q)|$, we obtain the second expression identified in the theorem. □

Theorem 5. $NewSwapVal(p,q) = -SwapVal(p,q)$.

Proof. The theorem results from expanding the terms that arise upon applying the definition of $SwapVal(p,q)$ to express $NewSwapVal(p,q)$. The theorem is also justified by the following simple argument. When the labels for p and q that have been swapped to produce $SwapVal(p,q)$ are swapped back to produce $NewSwapVal(p,q)$, the net change in the objective function must be 0 since we have returned to the starting point. Consequently, $NewSwapVal(p,q)$ must be the negative of $SwapVal(p,q)$. □

The foregoing theorems that give expressions for the quantities $CngV(i:j)$ and $CngV(p:i)$, etc., also similarly give expressions for the quantities $CngV(j:i)$ and $CngV(q:i)$, etc., simply by interchanging the roles of i and j and of p and q. In this fashion, on the assumption that a move is first made that swaps the labels for nodes p and q, the foregoing results cover all possible cases for identifying outcomes of moves that swap labels $L(i)$ and $L(j)$ for additional pairs of nodes i and j.

Evidently, by applying the foregoing results to determine $CngSwapVal(i,j) = CngV(i:j) + CngV(j:i) - (CngV(i) + CngV(j))$, the amount of computation to determine $NewSwapVal(i,j) = SwapVal(i,j) + CngVal(i,j)$, is exceedingly small compared to the amount required to determine $NewSwapVal(i,j)$ from its original definition. We identify the effort involved in this updating computation more precisely as follows.

Theorem 6 *(Computational effort). Define the following two sets of nodes for which the quantities $Val(i)$ and $SwapVal(i,j)$, change as a result of swapping the labels for nodes p and q:*

$$ValCngSet = \big\{i : NewVal(i) \neq Val(i)\big\}$$
$$SwapCngSet = \big\{(i,j) : NewSwapVal(i,j) \neq SwapVal(i,j)\big\}$$

Then

$$ValCngSet = \{p,q\} \cup \big(N(p) - N(q)\big) \cup \big(N(q) - N(p)\big)$$
$$SwapCngSet = \big\{(p,q)\big\} \cup \big\{(i,j) : i \in ValCngSet,\ j \in N(i)\big\}$$

Begin with $N_0 = \emptyset$ and $N_1 = N$.
While $N_1 \neq \emptyset$
    Select a node $i \in N_1$ and update N_0 and N_1 as follows:
    $N_0 := N_0 \cup \{i\}$
    $N_1 := N_1 \backslash (N(i) \cup \{i\})$
End While

Fig. 1. Algorithm for Generating N_0.

Moreover, the amount of computational effort to perform the updates identified in Theorems 1–5 is given by the size of ValCngSet and SwapCngSet, and hence are given respectively by $O(v_0)$ and $O(v_1)$ where

$$v_0 = \big|N(p)\big| + \big|N(q)\big|$$
$$v_1 = \big|N(p)\big| + \big|N(q)\big| + \sum_{i \in N(p) \cup N(q)} \big|N(i)\big|.$$

Proof. The identification of $ValCngSet$ and $SwapCngSet$ results follows from Theorems 1 and 3 for the cases where $i \neq p, q$ and $j \neq p, q$ and follows from Theorems 2 and 4 for the cases where one of i and j is the same as p or q. The determination of v follows from the fact that $O(|N(p)| + |N(q)|)$ effort is required to determine $NewVal(i)$ for $i \in N(p) \cup N(q)$ (hence for $i \in ValCngSet$) by Theorem 2 and also to determine $NewCngVal(p : i)$ and $NewCngVal(q : i)$ by Theorem 4. The component $\sum_{i \in N(p) \cup N(q)} |N(i)|$ in the definition of v is a bound on $|SwapCngSet|$, and each of the computations for elements of this set, as given in Theorem 3, is of order $O(1)$. Likewise, the update of Theorem 5 is $O(1)$. □

Complexity analysis. Because of Theorem 6, we see that the time complexity of performing a swap operation that exchanges the labels for nodes p and q in accordance with the stipulations of Theorems 1 to 5 is precisely $O(v_0) + O(v_1)$, for v_0 and v_1 as defined above. This quantity, which is composed of simple sums over the sets $N(p)$ and $N(q)$, and over the sets $N(i)$ for i in the union of $N(p)$ and $N(q)$, can examine each edge in these sets at most twice (once for each of the edge's endpoints) and hence is bounded above by twice the number of edges in E.

As demonstrated by the preceding computational analysis, Theorems 1–5 yield exceedingly efficient updates as the foundation of an algorithm that employs the quantities $Val(i)$ and $SwapVal(i, j)$ for evaluating potential swap moves. The fact that $ValCngSet$ is somewhat smaller than $SwapCngSet$ (v_0 versus v_1) gives rise to a variation of our method described below.

4. A class of set-based neighborhoods

We now introduce a class of set-based neighborhoods for the MinLA problem and identify rules for transitioning efficiently from one neighborhood to another. Set-based neighborhoods may be contrasted with the commonly employed parameter-based neighborhoods, such as k-flip neighborhoods for binary optimization problems and k-opt neighborhoods for traveling salesman problems where a value of the parameter k is selected (typically a small one such as 1, 2, 3, etc.) and then the neighborhood consists of all (non-overlapping) legitimate moves that can be generated for this value of k. In such k-flip neighborhoods, not all choices of k may be legitimate or need to be generated when k is successively incremented to overcome local optimality. For example, there are no 1-opt moves for traveling salesman problems and it is well-known that a number of k-opt moves can be generated by sequences of more elementary 2-opt moves, but the standard approach for any selected k is to systematically enumerate the available move configurations as a means to select one of them and obtain the next solution in the neighborhood.

A set-based neighborhood, by contrast, consists of selecting some set S of elements that are used to define a solution, and to generate new solutions that can be obtained by restricting attention to operations on the set S. Thus the choice of S replaces the choice of k as the object of interest, and as in the case of k, some choices of S may be excluded from consideration. Likewise, some structures generated relative to a given S may be excluded (just as in the case for a given k).

Our interest in set-based neighborhoods in the present context is motivated by the fact that we are able to identify a particular class of such neighborhoods, i.e., a particular characterization of the admissible sets S, for which it is unnecessary to enumerate configurations over S to obtain one that may be selected to give the next solution. Instead, we can identify the best configuration in a single step by means of a polynomial-time algorithm, even though the number of these configurations is exponentially large. To do this we once again draw on the adjacent versus non-adjacent differentiation underlying the analyses of the preceding section, but do so in a different manner.

Generating sets for MinLA and determining best solutions over them. Consider a current labeling $L = \{L(i) : i \in N\}$ and generate a node set $N_0 \subset N$ by the procedure in Fig. 1.

To identify a best solution relative to N_0, let $L_0 = \{L(i) : i \in N_0\}$. We now create a bipartite network assignment problem P_0 on the graph $G_0 = (N, L_0, E_0)$ where $E_0 = \{(p, q) : p \in N_0,\ q \in L_0\}$, and the cost $c(p, q)$ for edge $(p, q) \in E_0$ is given by $\sum_{k \in N(p)} |q - L(k)|$. We identify an assignment solution by $(p, q(p)),\ p \in N_0$, where $q = q(p) \in L_0$.

Theorem 7. *An optimal min cost solution to P_0 consisting of edges $(p, q(p)),\ p \in N_0,\ q(p) \in L_0$, identifies a feasible labeling of G that retains the labels of $N \backslash N_0$ unchanged, and assigns the label $q(p)$ to each node $p \in N_0$. Moreover, this labeling is an optimal labeling of G subject to the labeling over $N \backslash N_0$, and is the best among $n_0!$ labelings of G for $n_0 = |N_0|$.*

Proof. The Algorithm for Generating N_0 evidently yields N_0 as a maximal independent (stable) set of nodes in G, such that no edge (j, j) exists in E such that $i, j \in N_0$. By the structure of G_0 determined by the non-adjacency of nodes in N_0, any relabeling of G using the labels of L_0 over N_0, while retaining the labels of $N \backslash N_0$ unchanged, must be feasible labeling of G. The cost $c(p, q)$ equals the quantity $Val(p)$ if the label q is assigned to node $p \in N_0$, (in place of its current label $L(p)$). Consequently, the redefinition of the $Val(p)$ quantities determined by an assignment $(p, q(p))$ is clearly valid, and verifies the optimality of the min cost solution $(p, q(p))$ to P_0 for relabeling the nodes of N_0. This solution evidently is best among $n_0!$ labelings of G. □

We note that the exponential value $n_0!$ can be exceedingly large even for relatively small values of n_0. (For example $n_0! = 1{,}307{,}674{,}368{,}000$ for $n_0 = 15$, and $n_0!$ is more than 10^6 larger for $n_0 = 20$.) The combinatorial leverage that results from the ability to obtain solutions that dominate exponential numbers of alternatives by the polynomial effort of solving a network assignment problem is an attractive feature of our present class of set-based neighborhoods.[1]

Rules for transitioning between the set-based neighborhoods. We now address the challenge of identifying useful rules for transitioning from one of the N_0-based neighborhoods to another. In the case of parameter-based neighborhoods, the possible transition rules are based on simple protocols such as round-robin (token ring) selection, repeated reversion to the simplest neighborhood when improvement occurs, randomized selection, and so on. Here, a biased randomized protocol is a viable option, but we are additionally motivated to account for the composition of the sets N_0 as a foundation for transitioning from one to another. Specifically, we seek to generate new sets N_0 whose composition differs significantly from those previously generated, or from those generated over a recent time span (when labeling changes may make it relevant to re-visit previous N_0 sets).

To generate N_0 sets that are diverse, we employ a type of frequency-based memory called critical event memory, also called tabu memory for strategic oscillation [5], where the critical events correspond here to establishing a complete labeling over a current N_0. Hence we are interested in the frequencies that a given node has participated in one of these relabeling events. For this purpose, we maintain a count $C_i =$ the number of times node $i \in N$ has been selected to belong to one of the r most recent node sets N_0, where r can vary to become smaller when an improved labeling has been found. To manage this memory we maintain a record of the increments $C(h)$ for the $h = 1$ to r most recent sets N_0 generated, where $C_i(h) = 1$ if $i \in N_0$ for the h most recent of these sets and 0 otherwise. Then, as long as r is unchanged, the appropriate update is given by $C_i := C_i - C_i(r) + C_i(1)$, where $C_i(r)$ refers to the increment for the r^{th} (oldest) N_0 before examining the new N_0 and $C_i(1)$ refers to the increment determined by the new N_0. (Here, we implicitly renumber the increment vectors so they always range from 1 to r, though this can be handled by having a shifting pointer to the vector that is first.)

Using the count vector C in this way, we now generate new node sets N_0 by referring to C in the choice of node i at each execution of the Algorithm for Generating N_0. In particular, the choice step that says "Select a node $i \in N_1$" becomes

Select a node $i \in N_1$ such that $i = \arg\min\{C_j : j \in N_1\}$, breaking ties randomly.

A MNS algorithm for MinLA can then alternate between selecting the set-based N_0 neighborhoods and the coordinated neighborhoods of the preceding sections, as by transferring from one class of neighborhoods to the other when improvements via the given class become difficult to find.

5. Conclusions

We have introduced theorems for exploiting the structure of the graph G of the MinLA problem by identifying properties of complementary neighborhoods that can be incorporated into multi-neighborhood search algorithms. We have further introduced a class of set-based neighborhoods that provide an opportunity to generate labelings for the MinLA problem in polynomial time that dominate exponential numbers of other labelings.

Our results are independent of the numerical properties of the MinLA objective function, which relate to identifying the medians of labels assigned to neighbors of a given label, and thus can be used as a foundation for MNS approaches for other more general problems. A variety of implementations of our results are possible depending on the protocol of the specific form of MNS employed, inviting future research to explore such alternatives to determine which ones prove most effective for particular problem classes.

[1] A different but related class of set-based neighborhoods arises for the TSP problem by segregating odd and even nodes in a TSP tour. Then an assignment problem that re-assigns the positions of either the odd or even nodes produces an optimal tour subject to the restriction that the other nodes retain their odd-even position. In this case the transition between neighborhoods is trivial, progressing from odd to even and back, and the search terminates when neither option yields and improvement. Variations arise by treating node sequences as single nodes [4,7].

References

[1] D. Adolphson, T.C. Hu, Optimal linear ordering, SIAM J. Appl. Math. 25 (3) (1973) 403–423.
[2] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of Completeness, W. H. Freeman and Company, New York, 1979.
[3] F. Glover, Heuristics for integer programming using surrogate constraints, Decis. Sci. 8 (1) (1977) 156–166.
[4] F. Glover, Multilevel Tabu Search and Embedded Search Neighborhoods for the Traveling Salesman Problem, Research Report, University of Colorado, Boulder, 1991.
[5] F. Glover, Tabu thresholding: improved search by nonmonotonic trajectories, ORSA J. Comput. 7 (4) (1995) 426–442.
[6] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, Boston, 1997.
[7] F. Glover, C. Rego, New assignment-based neighborhoods for traveling salesman and routing problems, Networks (2017), in press.
[8] L.H. Harper, Optimal assignment of numbers to vertices, SIAM J. Appl. Math. 12 (1) (1964) 131–135.
[9] M. Juvan, B. Mohar, Optimal linear labelings and eigenvalues of graphs, Discrete Appl. Math. 36 (2) (1992) 153–168.
[10] R. Martí, J.J. Pantrigo, A. Duarte, V. Campos, F. Glover, Scatter search and path relinking. A tutorial on the linear arrangement problem, Int. J. Swarm Intell. Res. 2 (2) (2011) 1–21.
[11] A.J. McAllister, A New Heuristic Algorithm for the Linear Arrangement Problem, Technical Report TR-99-126a, Faculty of Computer Science, University of New Brunswick, 1999.
[12] J. Petit, Experiments on the minimum linear arrangement problem, ACM J. Exp. Algorithmics 8 (2003) 2–3.
[13] E. Rodríguez-Tello, J.-K. Hao, J. Torres-Jimenez, An effective two-stage simulated annealing algorithm for the minimum linear arrangement problem, Comput. Oper. Res. 35 (10) (2008) 3331–3346.
[14] I. Safro, D. Ron, A. Brandt, Graph minimum linear arrangement by multilevel weighted edge contractions, J. Algorithms 60 (2006) 24–41.