

Speeding up the Passenger Assignment Problem in transit planning by the Direct Link Network representation

Dilay Aktaş^{a,*}, Evert Vermeir^b, Pieter Vansteenwegen^a

^a*KU Leuven Institute for Mobility - CIB, Celestijnenlaan 300, 3001 Leuven, Belgium,*

^b*De Lijn, Flemish public bus company, 3000, Leuven, Belgium,*

Abstract

When designing a public transport network with the passenger travel time in mind, the passenger assignment problem (PAP) needs to be addressed explicitly. It is a complex problem on its own and even the simplest version requires determination of the shortest path for all demand pairs in the network. It is typically treated as a subproblem of larger problems by public transport planners, such as the transit network design problem (TNDP). In the TNDP, bi-level optimization models and/or metaheuristics are used where the PAP is solved when evaluating a line plan. In doing so, the PAP is often addressed by representing the transit network with a so-called ‘Change-and-Go’ (CNG) network with dummy transfer nodes in order to model transfer penalties as part of the passenger travel time. Then, in order to solve the PAP, the all-pairs shortest path problem needs to be solved for this CNG network representation. In this paper, we present a much more efficient network representation, ‘Direct Link Network’ (DLN), where additional edges are added instead of additional nodes. We compare the theoretical complexity of both representations and the computation time required to solve the PAP by using CNG and DLN on the most commonly used benchmark networks and also several real-life networks. The results show that with the DLN representation, the PAP can be solved multiple times faster than with CNG. Consequently, DLN can significantly speed up all TNDP algorithms that solve the PAP multiple times when designing a public transport network.

Keywords: Line Planning Problem, Passenger Assignment Problem, Change and Go Network, Train Service Network

1. Introduction

The transit network design problem (TNDP) or (uncapacitated) line planning problem (LPP) both refer to one of the first stages of a multi-stage problem: designing a public transport network (Ceder and Wilson, 1986; Desaulniers and Hickman, 2007; Lusby et al., 2018). After deciding on the *infrastructure network*, i.e., stops or stations and the roads or tracks between these, in the TNDP, the set of lines of a bus, train or tram system is decided. Based on an expected Origin-Destination (OD) demand matrix, it is decided which lines will serve which stops and in which order. In the next planning stages, the frequencies of the lines, timetable and lastly the crew and vehicle scheduling are determined.

Typically, the objective is to minimize the operator costs or to maximize the quality of service or a combination of both. The operator costs are generally associated with the length of the lines and/or the maximum number of stops whereas the service quality is measured by the passenger travel time, the number of transfers, or the number of direct travellers. Since frequencies have

*Corresponding author

Email addresses: dilay.aktas@kuleuven.be (Dilay Aktaş), evert.vermeir@delijn.be (Evert Vermeir), pieter.vansteenwegen@kuleuven.be (Pieter Vansteenwegen)

a direct impact on both the passenger travel time and the operator costs, integrating frequency setting with the line planning stage, also known as “Transit network design and frequency setting problem” (TNDFSP), has also gained attention (López-Ramos, 2014). However, due to the complexity of real-life problems, many simplifying assumptions are required to be made in order to solve the TNDFSP (Durán-Micco and Vansteenwegen, 2022). Thus, there exists a large gap between the problems addressed in theory and the problems encountered in practice. As a result, line planning and frequency setting problems are rather treated as separate and sequential problems in real-life (Ceder, 2016). It should be noted that the TNDP is also considered to be an extremely complex problem where even the special cases of the subproblems are proven to be NP-Hard (Schöbel, 2012).

When passenger travel time is considered in the TNDP, the *passenger assignment problem (PAP)* needs to be addressed and explicit passenger routing has to be solved. The PAP aims to model the expected traveling behaviour of passengers to reach their destinations. The easiest and the most common approach that is used in the TNDP is the ‘all-or-nothing’ transit assignment where all passengers are assigned to their shortest paths to minimize their in-vehicle time and transfer penalties (e.g. Fan et al. (2009); Ahmed et al. (2019b); Liu et al. (2020); Vermeir et al. (2021)). However, calculating the shortest path between each origin and destination pair is still highly time consuming, especially for real-life instances, and typically hard to integrate in the optimization (Schmidt et al., 2014). For this reason, several other assumptions are sometimes made to further simplify the problem such as ignoring the burden of transfers (Pfetsch and Borndörfer, 2006; Mauttone and Urquhart, 2009; Ranjbari et al., 2020) or assuming that the operator can assign a route to each passenger (Schöbel and Scholl, 2006; Schmidt et al., 2014). Yet, most of these simplifications are considered to be unrealistic. Instead, when considering the ‘all-or-nothing’ PAP, the TNDP is solved as a bi-level optimization problem where the aim is to first come up with a line plan for a given infrastructure network, and then to solve the PAP for the obtained line plan. Moreover, many (meta-)heuristics addressing the TNDP solve this PAP in every iteration and it is claimed to be the most time consuming part of most algorithms (Farahani et al., 2013; Liu et al., 2020; Durán-Micco and Vansteenwegen, 2022).

In order to accommodate transfer penalties in evaluating the ‘all-or-nothing’ PAP for a given line plan, a typical method used in literature is to add a dummy node for every bus line that passes through a stop. This extended network is then called “Change-and-Go” (CNG) (Schöbel and Scholl, 2006) or sometimes “Train Service Network” (Fu et al., 2015) and “Trajectory Graph” (Cancela et al., 2015). This means that the number of nodes significantly increases, especially when the line plan includes many lines with many transfer possibilities. Since the complexity of the shortest path algorithms (e.g., Dijkstra’s algorithm, Floyd-Warshall algorithm) mainly depends on the number of nodes, this extended network increases the computation time required for solving the PAP.

In this paper, a much more efficient “Direct Link Network” (DLN) representation for the TNDP with integrated passenger routing is presented. This representation is based on adding links instead of nodes and is explained in detail in Section 3. It is shown that DLN is expected to run much faster than CNG with traditional shortest path algorithms. To compare the performance of the CNG and DLN representations in more detail, Floyd-Warshall and Dijkstra’s algorithms are implemented in Matlab to solve the PAP for a given line plan of several benchmark instances. The main contribution of this paper is that it introduces and explains the DLN network representation that allows to solve the ‘all-or-nothing’ PAP much faster than the existing network representations. Since solving this PAP is crucial in all algorithms that address the TNDP with passenger travel in mind, this much faster PAP solution results in a significant improvement in the computation time of all these existing and future algorithms. In addition, DLN could also be applied to other problems similar to the PAP and that can be defined by an edge-coloured graph with vertex penalties such as multi-modal shortest path problems (Lewis, 2020). In those

problems, each mode is represented by a different colour and the transfer penalties at vertices correspond to moving people or goods from one mode to another. Lewis (2020) also identifies modelling traffic delays at certain intersections, initial headway requirements at the terminals, and illegal route and turn constraints as an edge-coloured graph with vertex penalties.

The organization of the paper is as follows. In Section 2, the related literature on the transit-network design problem is discussed. In Section 3, the existing network representations and then the new Direct Link Network representation and its theoretical complexity are explained. In Section 4, the benchmark instances and the computational results are presented. Finally, conclusions and future work are discussed in Section 5.

2. Literature Review

In this literature review, a review on the TNDP or (uncapacitated) LPP, with a focus on the studies considering passenger travel time, is presented. The research on the TNDP with only the operator’s point of view is not discussed since those systems do not consider passenger travel time explicitly. In addition, the studies that integrate the frequency setting with the line planning constitutes only a small part of this review since in the presence of frequencies, typically a frequency-based method is used for the transit assignment rather than the ‘all-or-nothing’ PAP. Only a small number of studies integrating frequency setting with the TNDP, i.e., TNDFSP, also consider the ‘all-or-nothing’ PAP with a constant transfer penalty since the frequency-based transfer times lead to non-linear models (Schöbel and Scholl, 2006; Schmidt et al., 2014; Goerigk and Schmidt, 2017). For more studies on the TNDP with operator’s point of view and the TNDFSP, the reader is referred to the earlier surveys: Schöbel (2012); López-Ramos (2014); Iliopoulou et al. (2019); Durán-Micco and Vansteenwegen (2022).

According to a recent survey on the TNDP and TNDFSP by Durán-Micco and Vansteenwegen (2022), the majority of the recent studies on TNDP optimize the passenger point of view while limiting the length of the lines and/or the maximum number of stops on each line to keep the operator costs under control (Fan and Mumford, 2010; Kechagiopoulos and Beligiannis, 2014; Nayeem et al., 2014; Iliopoulou and Kepaptsoglou, 2019; Vermeir et al., 2021). Some studies consider both the passengers’ and the operator’s point of view in a multi-objective manner, by simultaneously optimizing the passenger travel time and the total length of the lines (Chew et al., 2013; John et al., 2014; Fan et al., 2009; Mumford, 2013).

Although the passenger point of view is represented by the average or total travel time in most of the studies, other criteria such as maximizing ridership (Yoon and Chow, 2020) or maximizing the number of direct travellers (Suman and Bolia, 2019) have also been studied in optimizing line plans, probably in order to avoid the time consuming evaluation of the PAP during optimization. In those cases, the PAP is evaluated only at the end, for the resulting solution.

The PAP is often called the *transit assignment problem* (TAP) where a demand, a set of Origin-Destination (OD) pairs, is assigned to the transit routes. According to Liu et al. (2010), the TAP is classified under two major categories: static and dynamic transit assignment. It is a complex problem and different solution methods with different assumptions have been studied independently in order to model and optimize the traveling behaviour of passengers. Except for a few studies where dynamic assignment is used to design a public transport network (Nnene et al., 2019; Manser et al., 2020), static assignment models are used in most studies that address the TNDP and TNDFSP (Schöbel, 2012; Durán-Micco and Vansteenwegen, 2022).

The easiest and the most common PAP in the TNDP is a static assignment where passenger behaviour is independent of the time and other factors. The ‘all-or-nothing’ PAP assigns all passengers to their shortest paths considering the in-vehicle time and a transfer penalty. Many studies consider the ‘all-or-nothing’ PAP with a constant transfer penalty (typically 5 minutes) included in the shortest path calculation (Fan et al., 2009; Mumford, 2013; John et al., 2014; Ahmed et al., 2019a,b; Chew et al., 2013; Heyken Soares et al., 2019; Iliopoulou and Kepapt-

soglou, 2019; Fan et al., 2020; Vermeir et al., 2021). This means that the network representation should be appropriately expanded to model the transfer penalties. Yet, even this simple assignment is claimed to be the most time-consuming part in most TNDP solution approaches (e.g., Farahani et al. (2013); Liu et al. (2020)).

Some studies also aim to minimize the passenger travel time, by completely ignoring the burden of transfers and neglecting the transfer time (Pfetsch and Borndörfer, 2006; Mauttone and Urquhart, 2009; Ranjbari et al., 2020) or by only minimizing the in-vehicle time when determining the shortest-paths and adding the transfer penalties later, for those routes that require transfers (Fan and Mumford, 2010; Nikolić and Teodorović, 2013; Kechagiopoulos and Beligianis, 2014; Nayeem et al., 2014; Yao et al., 2014). It should be noted that if the transfer penalties are not considered when determining the shortest paths, a network representation is not necessary. However, this implies that the routes with a longer in-vehicle time, but less transfers are ignored when calculating the shortest paths. This issue is illustrated in Figure 1 on a toy network with 7 stops and 3 bus lines.

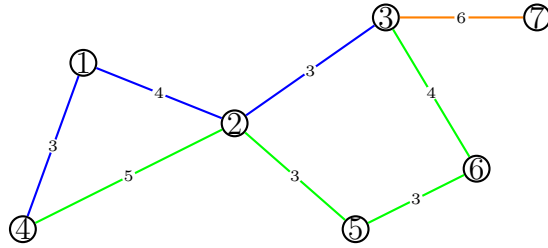


Figure 1: A toy example with 7 stops and 3 bus lines

According to Figure 1, when the passengers are assigned to their shortest paths only based on their in-vehicle time, the shortest path from stop 4 to stop 3 corresponds to the path ‘4-2-3’ with the total in-vehicle time of 8. However, if the transfer times were included in the shortest paths, any transfer penalty greater than 2 would change the shortest path to ‘4-1-2-3’ instead, which has the total in-vehicle time of 10 without any transfers. Similarly, when traveling from stop 5 to stop 7, the shortest path based on the in-vehicle time corresponds to the path ‘5-2-3-7’ with the total in-vehicle time of 12, and 2 transfers, whereas the path ‘5-6-3-7’ has a total in-vehicle time of 13 with only 1 transfer. This means that ignoring the transfer times or adding them at the end after finding the shortest paths might yield different conclusions for the exact same line plan simply because they might end up with different passenger assignments. For this reason, the transfer penalties should be taken into account when determining the shortest paths in order to properly model the additional cost of transfers in passenger travel times.

Due to the complexity of addressing the PAP within the optimization (Schmidt et al., 2014), only a few studies use exact optimization techniques when the passenger travel time is integrated in the line planning stage (Vermeir et al., 2021; Wu et al., 2020; De-Los-Santos et al., 2021). However, many different (meta-)heuristics that evaluate the PAP only when a line plan is found have been developed: bee colony optimization (Nikolić and Teodorović, 2013), simulated annealing (Fan and Mumford, 2010; Feng et al., 2018), multi-objective evolutionary algorithms (Fan et al., 2009; John et al., 2014), genetic algorithm (Nayeem et al., 2014; Feng et al., 2019), tabu search (Yao et al., 2014), etc. Iliopoulou et al. (2019) review different applications of metaheuristics to solve the TNDP as well as a comparison on a commonly used benchmark, Mandl’s Swiss Network (Mandl, 1979). Different from the existing studies, Vermeir et al. (2022) recently develop a fast and local evaluation technique in order to re-evaluate the PAP locally only considering the changes to the line plan.

When the ‘all-or-nothing’ PAP is evaluated for a given line plan, all shortest paths between all OD pairs need to be determined considering the transfer time. Among the research on TNDP, Fan and Mumford (2010); Islam et al. (2019); Ahmed et al. (2019b) explicitly mention that Dijkstra’s

algorithm (Dijkstra, 1959) is used whereas Mumford (2013); Kechagiopoulos and Beligiannis (2014); Liu et al. (2020); Vermeir et al. (2021) use the Floyd-Warshall algorithm (Floyd, 1962) when calculating all-pairs shortest path. In order to be able to apply these algorithms to solve the ‘all-or-nothing’ PAP, or in fact any other traditional shortest path algorithm, the network representation should be appropriately expanded when considering transfers in the passenger travel time.

In this paper a new and much faster network representation, the Direct Link Network (DLN) is proposed and tested for addressing the passenger assignment in the TNDP. Its theoretical complexity and the computation time required to solve the PAP are compared with the most commonly used network representation in the TNDP. In the next section, the existing network representations and the proposed Direct Link Network representation are explained in detail.

3. Direct Link Network

A typical and well-known approach used in the public transport literature is the “Change-and-Go Network” (CNG) by Schöbel and Scholl (2006). Although sometimes different names are used such as “Train Service Network” (TSN) in Fu et al. (2015); Liu et al. (2020) or “Trajectory Graph (TG)” in Cancela et al. (2015), these representations result in the same network representation. Also in, e.g., Mumford (2013), John et al. (2014), Ahmed et al. (2019b), Fan et al. (2020), Heyken Soares et al. (2019), the line plans are represented by this so-called CNG network expansion.

Much earlier, Kirby and Potts (1969) proposed a network expansion for calculating minimum routes on a graph with turn penalties. According to the “Kirby-Potts Expansion” (KPE), for each node and for each line, two duplicates are created to represent an entering and an exiting node. However with CNG, a single duplicate is made for each line passing through a node and the original node is connected to the duplicates to reflect the transfer penalties. Therefore, CNG is expected to create a much smaller network than KPE, which is sufficient for the uncapacitated line planning problem. Figure 2 below shows a comparison between KPE and CNG for a toy network with three lines serving six stops.

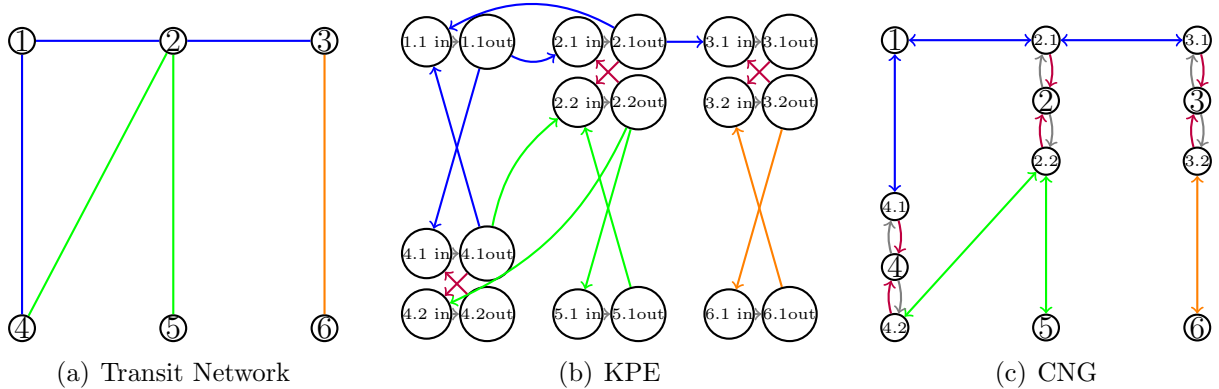


Figure 2: Comparison of KPE and CNG on a toy network

Figure 2a represents a simple transit network with three lines: a blue line (4-1-2-3), a green line (4-2-5) and an orange line (3-6). In Figure 2b, the KPE representation has an entering and an exiting node for each line that passes through a stop to incorporate the transfer penalties on the directed edges between the exiting and entering nodes of different lines (indicated in red). Overall, the KPE representation uses 18 nodes where the shortest paths should be calculated from the ‘in’ nodes as origins and to the ‘out’ nodes as destinations. In CNG, since only a single dummy node is included for each line that passes through a node and the original node is also kept, a smaller network consisting of 12 nodes is created. The transfer penalties can be included as the weights of the directed edges between the dummies and the original node (indicated in red). The gray arcs are assigned a weight of zero to facilitate the entrance to different lines.

This means that, in the end, one excess transfer penalty should be deducted to only consider the ‘real’ transfers.

Different than the existing approaches, Lewis (2020) proposes two modified versions of Dijkstra’s algorithm that are updated to incorporate the transfer penalties when directly applying the algorithm to the transit network, instead of a network expansion. However, the results of their study show that the KPE network expansion provides much faster evaluation of all-pairs shortest path than the modified Dijkstra’s algorithms. Yet, it provides a new line of research where the additional computational burden of the network expansions such as KPE, CNG, etc. are questioned and aimed to be avoided.

In the rest of this section, the direct link network representation is explained in detail and compared with the CNG representation. The DLN was first implemented and briefly introduced in Vermeir et al. (2021). In that paper, by partly the same authors as this paper, DLN was implemented to determine, for the first time, the optimal solutions of the TNDP for Mandl’s Network. Although that illustrated the potential of the DLN, its actual complexity and its performance on a variety of instances were not investigated. The impact it could have on a variety of algorithms was also not quantified, the DLN was just implemented for one algorithm on one network. Now, the DLN will be discussed in detail, its theoretical complexity will be analyzed and the experimental results will quantify the benefits when addressing the PAP and related problems.

In DLN, the set of vertices corresponds to the vertices in the original infrastructure network. Then, a link between a pair of vertices is added if and only if there exists a *direct link*, i.e., a single bus line that serves the two stops, without any transfers. Then, the weight of the link corresponds to the *direct* travel time between the two stops. In case multiple bus lines serve the two stops, the edge-weight or travel time corresponds to the minimum of the direct travel times. With DLN, the number of links required for a passenger to reach his/her destination according to the shortest path corresponds to the number of bus lines used. Therefore, it is very easy to keep track of the number of transfers. Thus, when a shortest path algorithm is applied, the shortest path itself stores the information on the number of transfers, as well. With this in mind, when a constant transfer penalty is considered, the transfer penalty (tp) can actually be added directly to each edge-weight of the DLN. Then, this DLN is used to calculate the shortest paths. This means that, in order to correctly determine the time required for the transfers, one transfer penalty needs to be subtracted from each shortest path in the end.

Figure 3 presents the transit network for a line plan consisting of 3 lines on a part of Mandl’s Network (Mandl, 1979), and the corresponding CNG and DLN network representation.

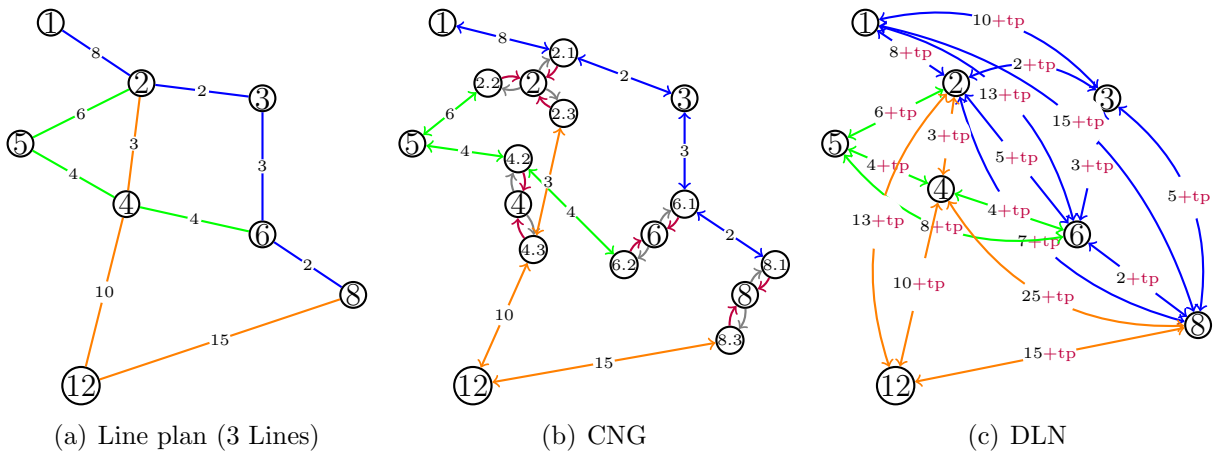


Figure 3: An example of DLN and CNG on a part of Mandl’s Network

It should be noted that in DLN, when multiple lines serve the same pair of stops, the minimum direct travel time is included as the edge-weight. For instance, both the blue and the green lines

serve stops 2 and 6 without any transfers. Therefore, the link between these two nodes has the weight of 5 as the length of the trip by the blue line is shorter than the length of the trip by the green line which is 14. Another example are the nodes 2 and 8 where the blue and orange lines serve the two stops without transfers. In this case, the weight of the edge between the two nodes correspond to 7 as the blue line is faster than the orange line which takes 28. Finally, the transfer penalty (tp) is added to the weight of each link on DLN.

To illustrate the use of traditional shortest path algorithms on the CNG and the DLN, consider the shortest path from node 4 to node 8. For CNG, the total cost of following the path “4-12-8” corresponds to visiting the nodes “4-4.3-12-8.3-8” which has the total length of $25 + tp$ ($0 + 10 + 15 + tp$) while the total cost of following the path “4-6-8” corresponds to visiting the nodes “4-4.2-6.2-6.1-8.1” with the total length of $6 + 2 * tp$ ($0 + 4 + tp + 0 + 2 + tp$). The first path includes no transfers, but one transfer penalty is incurred, while the second path includes one transfer and two transfer penalties are incurred. Therefore, in the end, from all of the shortest paths, one excess transfer penalty should be removed. Similarly, for DLN, the total cost of reaching node 8 from node 4 without any transfers corresponds to the edge-weight between 4 and 8, which is $25 + tp$, while the total cost of transferring at node 6 corresponds to $6 + 2 * tp$ ($4 + tp + 2 + tp$) even though only one transfer is required. This means that, one excess transfer penalty should be subtracted from each shortest path in the end. This example also illustrates that not allowing transfers or only minimizing the use of transfers, resulting in path 4-12-8, might lead to (much) longer paths than necessary.

It should also be noted that if transfer penalties are not constant but station-dependent, the same DLN or CNG approach can still be applied. The corresponding transfer penalties incurring due to transferring in a given station is added to the weight of each outgoing link from the station. Again, this means that in the end, an excess transfer is considered thus one transfer penalty corresponding to the departure station should be subtracted from each shortest path.

3.1 Shortest Path Algorithms and Theoretical Complexity

As already mentioned in the literature, the Floyd-Warshall and Dijkstra’s algorithms are typically used to solve the PAP in TNDP. Dijkstra’s algorithm finds the shortest-path tree from a source node (Dijkstra, 1959). With the proper representation of priority queues, the theoretical complexity of finding a shortest-tree from a source node with Dijkstra’s algorithm can be reduced to $O(m \cdot \log(n))$ instead of $O(n^2)$ (Lewis, 2020), where m denotes the number of edges and n denotes the number of nodes. For the all-pairs shortest path, the algorithm should be run n times.

The Floyd-Warshall (FW) algorithm directly calculates the all-pairs shortest paths (Floyd, 1962). The algorithm runs by inserting each node as an intermediate node between a pair of nodes if it makes it shorter to pass by it. Although the theoretical complexity of the basic Floyd-Warshall algorithm is $O(n^3)$ (Lewis, 2020), it can easily be coded by using vector structures and it is observed to run faster than the basic algorithm. A Matlab code for the “Vectorized-Floyd Warshall” algorithm is shared publicly by Giorgos (2018) and that is implemented in this paper, as well. A comparison with other shortest path algorithms and more details on the theoretical complexities can be found in Lewis (2020).

The exact number of nodes and edges in the CNG and the DLN representations depends on the detailed characteristics of the line plan, such as the number of stops per line and the number of lines on a given node. Still, the following formulas can be derived to show the worst-case theoretical complexity of the representations.

Let n be the number of nodes and m be the number of edges in the infrastructure network, n_k be the number of stops on line k and k_j be the number of different lines passing through the stop j in the transit network. Suppose l is the total number of lines and n_l is the maximum allowed line length, expressed as a maximum number of stops.

In the CNG representation, in addition to the original nodes, a single duplicate is created for each bus line passing through a node to represent the different lines at transfer stations. When only one line passes through a node, a duplication is in fact not necessary. Therefore, the total number of nodes in the CNG representation, n_{CNG} can be written in terms of the number of transfer stations, n_T , i.e., the number of stops with at least two lines passing, and the number of lines passing through each node as follows:

$$n_{CNG} = n_T + \sum_{j=1}^n k_j \quad (1)$$

Since each node served by each different line becomes a different duplicate in the CNG representation, the total number of nodes can also be written based on the line length as follows:

$$n_{CNG} = n_T + \sum_{k=1}^l n_k \quad (2)$$

In the worst case, n_T is equal to n where all stops are transfer stations with at least two lines serving. Also, for a given number of lines l , if all lines pass through every node, this gives the theoretical maximum possible number of duplicates or the maximum total line length. However, typically a maximum line length, n_l , is considered, which is much smaller than the number of nodes, n . Therefore, in the worst case, the most number of duplicates would be observed when the total line length is the maximum, i.e., when each line serves the maximum number of stops allowed. Then, the total number of duplicates and the total line length would be $l n_l$. The total number of nodes in the CNG representation for the worst case can therefore be written in terms of n , n_l , and l as follows:

$$n_{CNG} = n + l n_l \quad (3)$$

The directed edges in the CNG representation, m_{CNG} , includes all the original edges in the transit network, i.e., the edges connecting different nodes by different lines. Note that if the line k is bi-directional, the number of directed edges line k brings to the CNG representation equals $2(n_k - 1)$. If line k is loop-shaped, then the number of directed edges due to this line equals n_k . In addition to the original edges of the transit network, the original nodes and each duplicate should be connected by two directed edges in CNG. Then, the total number of directed edges in the CNG network is calculated as follows:

$$m_{CNG} = \sum_{k \in K_2} 2(n_k - 1) + \sum_{k \in K_1} n_k + \sum_{j=1}^n 2k_j \quad (4)$$

where K_1 is the set of loop-shaped lines and K_2 is the set of bi-directional lines.

The worst case for the number of edges due to the line plan is observed when each line has the maximum line length, n_l . Thus, n_k can be replaced by n_l in Equation 4. Moreover, when Equations 1 and 2 are considered together, it is observed that the total of the number of lines passing through all nodes is equal to the total line length. Therefore, the last part of the Equation 4 can also be written in terms of the total line length. This results in the following formula for m_{CNG} :

$$m_{CNG} = \sum_{k \in K_2} 2(n_l - 1) + \sum_{k \in K_1} n_l + \sum_{k=1}^l 2n_l \quad (5)$$

Lastly, as long as a line serves at least two stops, there exists more directed edges for a bi-directional line than a loop-shaped line. Thus, the most number of directed edges in the CNG representation can be observed if all the lines are bi-directional. Assuming that all lines are bi-directional, the total number of directed edges in the CNG representation for the worst case can be written in terms of n_l and l as follows:

$$m_{CNG} = 4l n_l - 2l \quad (6)$$

In the DLN representation, the number of nodes, n_{DLN} equals the number of nodes in the transit network or the infrastructure network, n , assuming that all nodes are somehow served. However, the representation adds additional links between the nodes if there exists a single line that serves the two stops.

Unlike the CNG, the number of directed edges in the DLN representation is the same regardless of the line being bi-directional or loop-shaped. In both cases, there exists a direct link between all pairs that are served by the line and therefore the number of directed edges originated due to the line k equals $2 \binom{n_k}{2}$. Then, the total number of directed edges in the DLN representation is as follows:

$$m_{DLN} = \sum_{k=1}^l 2 \binom{n_k}{2} \quad (7)$$

Similar to the CNG, the most number of edges can be observed when each line has the maximum line length, n_l . Thus, n_k can be replaced by n_l . Then, the total number of directed edges in the DLN representation for the worst case can be written in terms of n_l and l as follows:

$$m_{DLN} = l n_l (n_l - 1) \quad (8)$$

These (worst case) numbers of nodes and edges for the CNG and DLN representation are interesting from a theoretical perspective, but they do not give much practical insights. Therefore, the growth of complexity is now illustrated by using fictitious line plan configurations where the maximum line length, n_l , varies between 10 and 25 stops, and the number of lines, l , varies between 50 and 80. The number of nodes, the number of directed edges and the worst-case theoretical complexity for both Floyd-Warshall and Dijkstra's algorithm are analyzed considering an infrastructure network of 127 nodes.

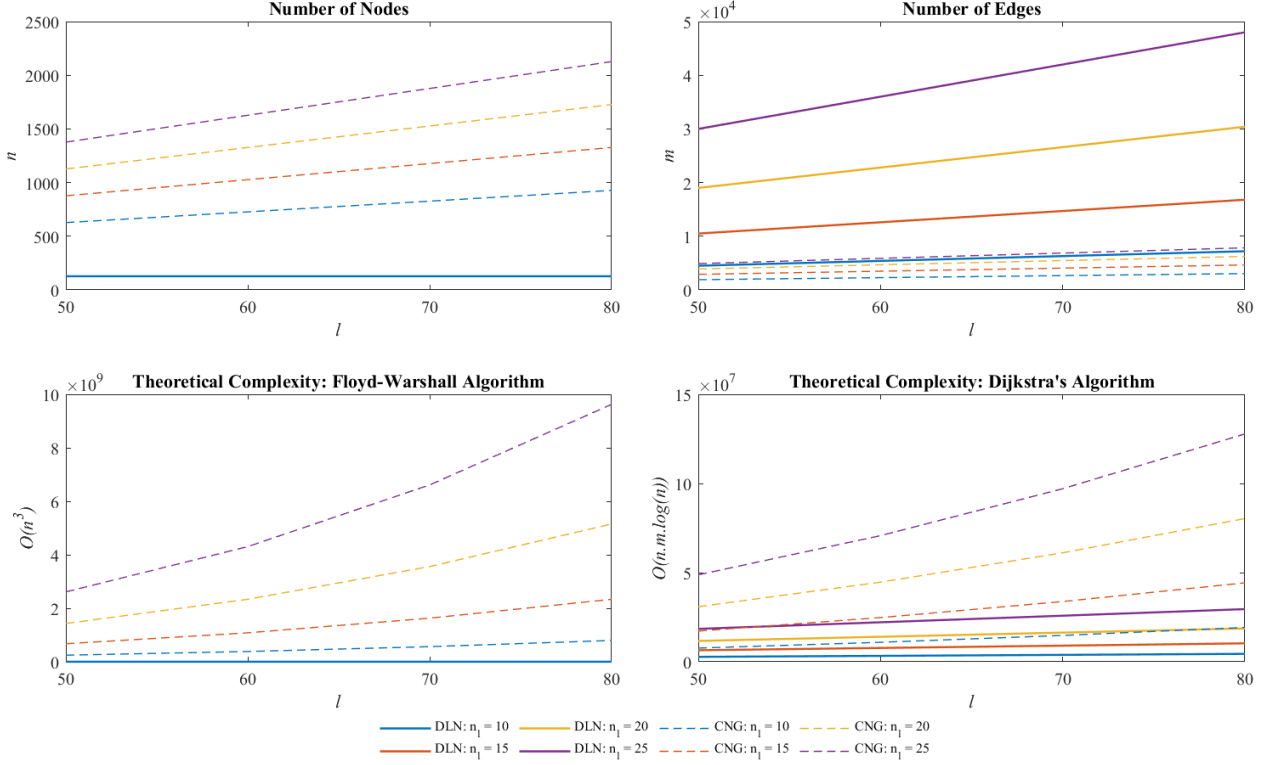


Figure 4: Worst-case Theoretical Complexity: DLN vs. CNG

As can be seen from Figure 4, in the CNG, the number of nodes increases as more lines are considered in the line plan. As expected, the CNG representation always has a smaller number of directed edges than the DLN representation for the same line configuration. However, the number of nodes in the DLN is always smaller than the number of nodes in CNG, in fact n_{DLN} is always equal to the number of nodes in the infrastructure network. Thus, the theoretical complexity of solving the traditional shortest path algorithms with DLN is always smaller than the one of CNG, even when the Dijkstra's algorithm is used with priority queues where the theoretical complexity not only depends on the number of nodes but also depends on the number of edges. These results illustrate the potential of DLN to run much faster than CNG with both algorithms when solving the PAP. Yet, the line plans considered in this part are fictitious considering the worst-case scenario and only for a given infrastructure network. Thus, in the next section, the shortest path algorithms are run on a set of benchmark networks and the real computation times are compared.

4. Experiments

In this section, first, the benchmark instances used in the experiments are presented. Then, the all-pairs shortest paths are calculated by using Floyd-Warshall and Dijkstra's algorithms using the DLN and CNG network representations on the benchmark instances. The computational experiments that evaluate the 'all-or-nothing' PAP consider a 5 minutes transfer time penalty.

4.1 Benchmark Instances

In order to conduct research on the TNDP, an infrastructure network and a demand matrix is required. Mandl's Network (Mandl, 1979) in Figure 5 below, is one of the first publicly available and still the most commonly used benchmark network in line planning (Iliopoulou et al., 2019). Even though it is a small network with fifteen nodes and twenty-one bi-directional links (or forty-two unidirectional edges), much research has been conducted on it to improve the best solutions found for different instances on this network (e.g., Fan et al. (2009), Fan and Mumford (2010), Nikolić and Teodorović (2013), John et al. (2014), Kechagiopoulos and Beligiannis (2014),

Nayeem et al. (2014), Ahmed et al. (2019b), Chew et al. (2013), Fan et al. (2020)) until the optimal solutions were determined by Vermeir et al. (2021).

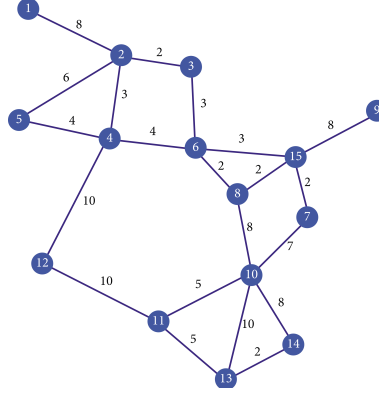


Figure 5: Mandl's Swiss Network (Mandl, 1979): fifteen nodes and twenty-one links with the travel time indicated

Mumford (2013) published larger instances, Mumford 0, 1, 2, 3 in Figure 6 below, based on Yubei in China, and Brighton and Cardiff in the UK. These have also been used since then by several researchers (e.g., Mumford (2013), Nikolić and Teodorović (2013), John et al. (2014), Nayeem et al. (2014), Ahmed et al. (2019b), Iliopoulou and Kepaptsoglou (2019), Fan et al. (2020)).

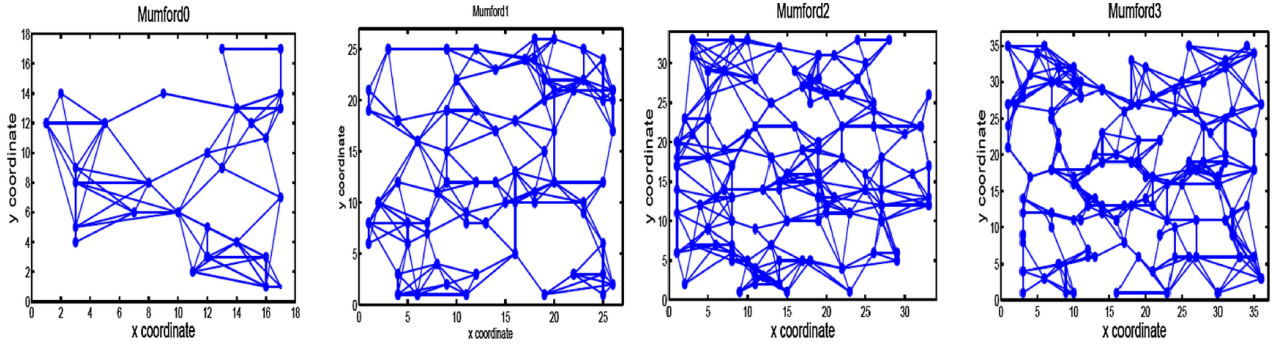


Figure 6: Mumford 0, 1, 2, 3 Networks (Mumford, 2013)

Other than these instances, different instances based on real cities have been used by different studies such as Utrecht (Durán-Micco et al., 2022), Sevilla (De-Los-Santos et al., 2021) and Nottingham (Ahmed et al., 2019b). Yet, these are not as widely used as the Mandl's Network or Mumford Networks and a fair comparison between different studies is not yet possible on these networks. For this reason, Schiewe et al. (2022) present a database, namely "LinTim", to provide benchmark instances and even a tool to solve the different stages to design a public transport network.

In this paper, in order to evaluate the 'all-or-nothing' PAP for a given line plan with the CNG and DLN representations, the Mandl's Network with optimal line plans ranging from 3 to 14 lines determined by Vermeir et al. (2021), Mumford 0, 1, 2, 3 networks with the line plans presented in Mumford (2013), and also several real-life networks are considered: Athens' metro system and Göttingen's bus system together with the line plans obtained from "LinTim" (Schiewe et al., 2022), and lastly Utrecht's bus system with the city's current line plan used in Durán-Micco et al. (2022).

These networks and line plans are selected as instances here, because these are widely used (Mandl and Mumford) or because they correspond to real size networks (Utrecht, Athens, Göttingen). These instances allow to evaluate the DLN representation on (benchmark) instances with different characteristics and size. Table 1 presents the characteristics related to the infrastructure network: the number of nodes, the number of directed edges, and the percentage

of OD pairs for which a directed edge from the origin to the destination exists in the infrastructure network. The characteristics of the line plans are also indicated: the number of lines, the percentage of OD pairs where at least one line, i.e., a direct link, serves both the origin and the destination, the percentage of OD pairs where a direct link exists but the passengers are better off making a transfer, i.e., the shortest path includes at least one transfer despite the transfer penalty, and the number of non-served stops and the type of lines used.

Table 1: Instance Characteristics

Instance	Infrastructure Network			Line Plan				
	# of nodes	# of directed edges	% of OD pairs with a directed edge	# of lines	% of OD pairs with a direct link	% of OD pairs better off with transfer	# of nonserved stops	line type
Mandl3	15	42	20	3	69	5	-	bi-directional
Mandl4	15	42	20	4	72	6	-	bi-directional
Mandl5	15	42	20	5	80	6	-	bi-directional
Mandl6	15	42	20	6	88	4	-	bi-directional
Mandl7	15	42	20	7	92	1	-	bi-directional
Mandl8	15	42	20	8	95	1	-	bi-directional
Mandl9	15	42	20	9	94	2	-	bi-directional
Mandl10	15	42	20	10	96	2	-	bi-directional
Mandl11	15	42	20	11	96	1	-	bi-directional
Mandl12	15	42	20	12	97	0	-	bi-directional
Mandl13	15	42	20	13	98	0	-	bi-directional
Mandl14	15	42	20	14	98	1	-	bi-directional
Mumford0	30	180	21	12	93	30	-	bi-directional
Mumford1	70	420	9	14	92	56	-	bi-directional
Mumford2	110	770	6	56	73	41	-	bi-directional
Mumford3	127	850	5	60	71	44	-	bi-directional
Athens	51	104	4	12	22	0	7	bi-directional
Göttingen	257	656	1	63	23	0	-	bi-directional
Utrecht	271	940	1	52	19	6	-	bi-directional & loop

As can be seen from Table 1, Mandl’s Network corresponds to the smallest network while Utrecht’s bus system corresponds to the largest based on the number of nodes and edges. Although Athens’ metro falls between Mumford0 and Mumford1 based on the number of nodes, it is considered in the same group as Göttingen and Utrecht’s bus systems since its density is as small as the two others. When larger and more realistic networks are considered, the percentage of OD pairs that have a direct link in the infrastructure network without an intermediate stop decreases. When the different line plans are considered for Mandl’s Network, the percentage of OD pairs with a direct link increases as the number of lines increases. Similarly, since more and more direct links are present between different OD pairs, the percentage of OD pairs that are better off with at least one transfer decreases. According to the line plan obtained from “LinTim” (Schiewe et al., 2022), Athens’ metro line has seven nonserved stops while the other instances somehow serve all OD pairs. Moreover, except for the Utrecht’s current bus line plan where loop-shaped lines are allowed, the lines are always bi-directional in the other instances.

4.2 Computational Results

In this paper, we implemented Dijkstra’s algorithm with priority queues, and Floyd-Warshall algorithm with vector structures in Matlab on a computer equipped with a processor AMD Ryzen 7 PRO 2700U w/ Radeon Vega Mobile Gfx 2.20 GHz with 16 GB RAM. The average computation time over 10 runs required to calculate the ‘all-or-nothing’ PAP on CNG and DLN representations using the FW and Dijkstra’s algorithm are presented in Table 2 below in seconds. Obviously, for each OD pair, the shortest paths obtained by the different algorithms have the same length. Also, the computation time required to generate the CNG and DLN network representations are reported (column “Initialization time”), together with the number of nodes and directed edges in both representations.

Table 2: Comparison of the computation time with CNG and DLN

Instance	Change-and-go (CNG)						Direct Link Network (DLN)					
	# of nodes	# of directed edges	% of OD pairs with a directed edge	Initialization time (s)	FW (s)	Dijkstra (s)	# of nodes	# of directed edges	% of OD pairs with a directed edge	Initialization time (s)	FW (s)	Dijkstra (s)
Mandl3	39	66	4.5	<0.0005	<0.0005	0.013	15	144	68.6	<0.0005	<0.0005	0.013
Mandl4	47	88	4.1	<0.0005	0.001	0.014	15	152	72.4	<0.0005	<0.0005	0.009
Mandl5	55	110	3.7	<0.0005	0.001	0.015	15	168	80.0	<0.0005	<0.0005	0.010
Mandl6	63	132	3.4	0.001	0.002	0.017	15	184	87.6	<0.0005	<0.0005	0.011
Mandl7	71	154	3.1	0.001	0.003	0.019	15	194	92.4	<0.0005	<0.0005	0.009
Mandl8	79	176	2.9	0.001	0.004	0.021	15	200	95.2	<0.0005	<0.0005	0.010
Mandl9	85	192	2.7	0.001	0.004	0.024	15	198	94.3	<0.0005	<0.0005	0.011
Mandl10	93	214	2.5	0.001	0.005	0.024	15	202	96.2	<0.0005	<0.0005	0.010
Mandl11	100	233	2.4	0.001	0.007	0.031	15	202	96.2	<0.0005	<0.0005	0.013
Mandl12	108	255	2.2	0.001	0.007	0.036	15	204	97.1	<0.0005	<0.0005	0.014
Mandl13	116	277	2.1	0.001	0.008	0.027	15	206	98.1	<0.0005	<0.0005	0.010
Mandl14	124	299	2.0	0.001	0.009	0.029	15	206	98.1	<0.0005	<0.0005	0.010
Average	82	183	3.0	0.001	0.004	0.022	15	188	89.7	<0.0005	<0.0005	0.011
Mumford0	210	516	1.2	0.001	0.041	0.082	30	812	93.3	<0.0005	<0.0005	0.036
Mumford1	488	1226	0.5	0.004	0.738	0.478	70	4438	91.9	0.001	0.003	0.488
Mumford2	1342	3584	0.2	0.021	17.064	5.925	110	8696	72.5	0.002	0.008	1.562
Mumford3	1627	4380	0.2	0.028	32.467	9.451	127	11330	70.8	0.003	0.012	2.591
Average	917	2427	0.5	0.014	12.578	3.984	84	6319	82.1	0.002	0.006	1.169
Athens	161	306	1.2	0.002	0.016	0.102	51	556	21.8	<0.0005	0.001	0.043
Göttingen	1594	3885	0.2	0.048	30.077	22.604	257	14944	22.7	0.004	0.058	4.849
Utrecht	1127	2425	0.2	0.033	9.897	11.177	271	13632	18.6	0.002	0.062	5.629
Average	961	2205	0.5	0.028	13.330	11.294	193	9711	21.0	0.002	0.041	3.507
Overall	396	975	2.1	0.008	4.755	2.636	58	2944	77.2	0.001	0.008	0.807

As can be seen from Table 2, when the line plans on Mandl’s Network are considered, according to CNG, the number of nodes and edges increases significantly when the number of lines increases. For example, Mandl14 with 14 lines has 124 nodes and 299 directed edges in CNG and 15 nodes and 206 directed edges in DLN while only 15 nodes and 42 directed edges in the infrastructure network. Although DLN is based on adding additional edges instead of additional nodes, for the largest five Mandl instances (Mandl14, Mandl13, Mandl12, Mandl11, and Mandl10), more edges are required for CNG than for DLN. When the resulting network densities are considered, it is observed that CNG always produces a more sparse network (at most 5% of OD pairs has a directed edge) whereas the resulting network with DLN is typically more dense (at least 19% and at most 98% of OD pairs has a directed edge). Another conclusion is that both network representations require negligible time for the initialization although a slightly longer time is required to initialize the real networks (Athens, Göttingen, Utrecht) for both CNG and, to a smaller extent, for DLN.

Overall, when the results with CNG are considered, the computation time required for the ‘all-or-nothing’ PAP for Mandl’s Network is rather small (less than 0.04 seconds) for all the different line plans and different algorithms. However, for Mumford 0, 1, 2, 3, and the real networks, the computation time increases drastically (up to 33 seconds with FW and 10 seconds with Dijkstra’s algorithm). It is also observed that with larger networks, usually Dijkstra’s algorithm performs better than FW with CNG.

When the results using DLN are considered, especially with FW, the computation times are always smaller than a second. With DLN and Dijkstra’s algorithm, the computation time increases for larger instances however it remains always smaller than 6 seconds.

Lastly, when CNG and DLN representations are compared for each of the chosen shortest path algorithms, it is observed that the computation time with DLN is always as small as the computation time when CNG is used. Moreover, each algorithm runs multiple times faster with DLN. Table 3 summarizes the ratio of the best computation times when CNG and DLN representations are used. Since the computation times are very small for all Mandl’s instances, resulting ratios would not be meaningful. Therefore, these results are omitted from Table 3.

Table 3: Best Algorithm for CNG vs Best Algorithm for DLN

Instance	Change-and-go (CNG)		Direct Link Network (DLN)		Ratio
	Best Algorithm	Computation Time (s)	Best Algorithm	Computation Time	
Mumford0	FW	0.041	FW	0.000	231.43
Mumford1	Dijkstra	0.478	FW	0.003	153.22
Mumford2	Dijkstra	5.925	FW	0.008	726.64
Mumford3	Dijkstra	9.451	FW	0.012	756.54
Average	Dijkstra/FW	3.984	FW	0.006	466.96
Athens	FW	0.016	FW	0.001	27.06
Göttingen	Dijkstra	22.604	FW	0.058	386.72
Utrecht	FW	9.897	FW	0.062	158.41
Average	Dijkstra/FW	11.294	FW	0.041	190.73
Overall	Dijkstra/FW	2.636	FW	0.008	348.57

Overall, the DLN representation allows to evaluate the ‘all-or-nothing’ PAP about 350 times faster. This can be explained by looking back at the computational complexity of FW and Dijkstra’s algorithm. On the one hand, the FW algorithm has the theoretical worst-case complexity of $O(n^3)$ to solve the all-pairs shortest path problem, which depends only on the number of nodes. On the other hand, the theoretical complexity of a single run of Dijkstra’s algorithm that calculates the shortest-tree from a source node can be reduced to $O(m.log(n))$ with priority queues. Yet, the Dijkstra’s algorithm needs to be run n times in order to solve all-pairs shortest path problem, thus it results in an overall computational complexity of $O(n.m.log(n))$, which depends on both the number of edges and nodes. Nevertheless, the complexity of these shortest path algorithms depends on the number of nodes on a higher degree than the number of edges. This can be seen in Figure 7 below where the computational time required to evaluate the ‘all-or-nothing’ PAP on the Mumford 0, 1, 2, 3, and Athens, Göttingen, and Utrecht instances are shown with respect to the number of nodes in their resulting CNG and DLN representations.

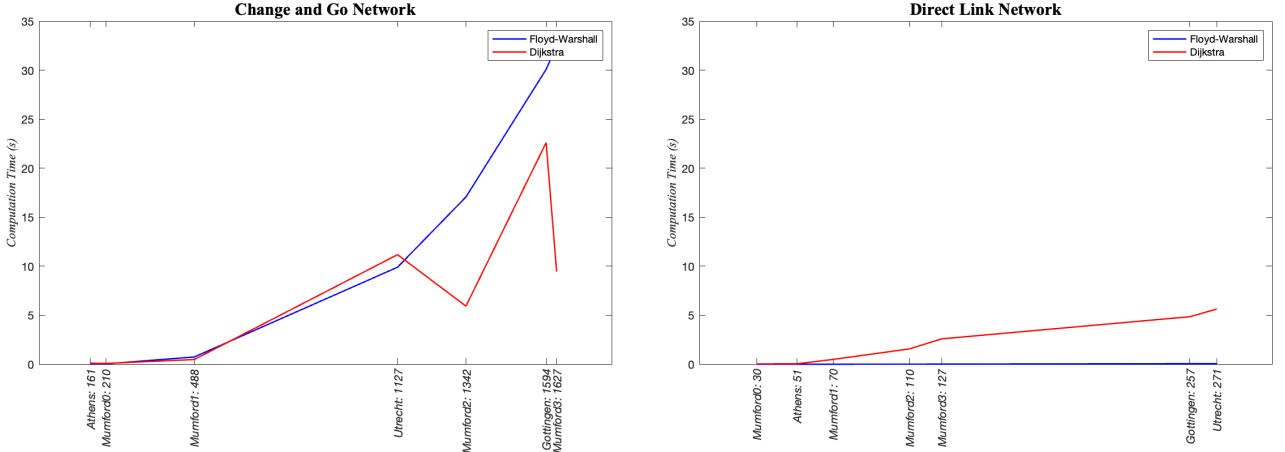


Figure 7: Computational Time: CNG vs. DLN

As can be seen from Figure 7, the computational time required by the FW and Dijkstra’s algorithm increases as the number of nodes in the corresponding network representations increase. Two exceptions occur, Mumford 2 and Mumford 3, when Dijkstra’s Algorithm is considered with CNG. In this case, the computational time for Mumford 2 and Mumford 3 is smaller when compared to Utrecht and Göttingen. This can be explained by the relatively high number of lines considered in the Mumford instances and by the fact that when Dijkstra’s Algorithm is used to find the shortest-tree, the algorithm needs to be run on the CNG representation, considering all the transfer nodes and additional edges. However, the algorithm does not need to run for each

node of this expanded network to calculate all-pairs shortest paths. Instead, the algorithm runs only for the original set of nodes in the infrastructure network. Since the number of nodes in the infrastructure network of Mumford 2 is smaller than the one of Utrecht and similarly the number of nodes in the infrastructure network of Mumford 3 is smaller than the one of Göttingen, smaller computational times are observed even though the total number of nodes in the CNG network representation are larger. Note that this does not happen with DLN since the number of nodes in the infrastructure network and in the DLN are always equal. Nonetheless, the main conclusion from this figure is that the number of nodes, both in the infrastructure network and the network representation, impacts the required computational time significantly. And this is exactly why the DLN is always expected to work so much faster than CNG, despite the many more additional edges.

Another reason why DLN works so much faster compared to the CNG could be that during the initialization of DLN, the edge-weights are assigned such that it corresponds to the shortest direct link when multiple bus lines serve the same pair of stops. This implies that a part of the comparisons needed to calculate the shortest-path is no longer required, by considering directly the faster line if multiple lines serve the two stops. And it should be noted that still the initialization time for the DLN is observed to be rather negligible, and more importantly, this initialization time is less than the initialization time for CNG.

5. Conclusions

In this paper, a much more efficient “Direct Link Network” (DLN) representation is presented for the Transit Network Design Problem (TNDP) or line planning problem. Typically, the ‘all-or-nothing’ passenger assignment (PAP) is used where all passengers are assigned to their shortest paths based on the in-vehicle time and a constant penalty for each transfer. Since even this most basic version of the PAP is not always easy to integrate in the optimization, bi-level optimization models and meta-heuristics are proposed. Only after finding a line plan, the ‘all-or-nothing’ PAP is evaluated. Yet, it is still claimed to be the most time consuming part of line planning algorithms (Farahani et al., 2013; Liu et al., 2020; Durán-Micco and Vansteenwegen, 2022).

Contrary to the commonly used “Change-and-Go” network representation (Schöbel and Scholl, 2006) and its variants: “Train-Service-Network” (Fu et al., 2015), and “Trajectory Graph” (Cancela et al., 2015), or earlier “Kirby-Potts Expansion” (Kirby and Potts, 1969), the DLN adds additional edges instead of additional transfer nodes. Therefore, the DLN results in a network with much less nodes and sometimes even less edges. Since the complexity of traditional shortest path algorithms depends more on the number of nodes than on the number of edges, the DLN representation is always much faster than the node-expansion methods. As a direct result, the traditional shortest path algorithms run on average 350 times faster on the DLN. Knowing that current metaheuristics easily solve thousands of PAPs when improving a line plan step by step, the DLN promises a significant improvement in the computational efficiency of all existing and future transit network design algorithms that evaluate the ‘all-or-nothing’ PAP in each iteration.

As already mentioned in the paper, the approach is explained and tested only for the commonly used assumption of a constant transfer penalty. When station-dependent transfer penalties are considered, the same method can be applied. The only difference is that the excess transfer penalty that should be eliminated to obtain the shortest path, should correspond to the penalty of the origin station. As future work, frequency-dependent transfer penalties, which can easily be considered in CNG with some modifications, could also be considered with DLN. However, this time, a modification to the shortest path algorithm is required, instead of adding the transfer penalties to each direct link. Furthermore, we are convinced that using the DLN instead of CNG in a metaheuristic addressing the TNDP, could have additional benefits. Evaluating the impact

of a small change to the line plan, might not require a complete evaluation of the PAP since most direct links will not be affected. This should be implemented and evaluated in detail in future work.

Data Availability

All the data of the experiments and the instances are available at: <https://www.mech.kuleuven.be/en/cib/lp/#section-28>.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This project is supported by the FWO (Research Foundation Flanders), project G.0759.19N.

References

- Ahmed, L., Heyken-Soares, P., Mumford, C., Mao, Y., 2019a. Optimising bus routes with fixed terminal nodes: comparing hyper-heuristics with nsgaii on realistic transportation networks, in: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1102–1110.
- Ahmed, L., Mumford, C., Kheiri, A., 2019b. Solving urban transit route design problem using selection hyper-heuristics. *European Journal of Operational Research* 274, 545–559.
- Cancela, H., Mauttone, A., Urquhart, M.E., 2015. Mathematical programming formulations for transit network design. *Transportation Research Part B: Methodological* 77, 17–37.
- Ceder, A., 2016. Public transit planning and operation: Modeling, practice and behavior. CRC press.
- Ceder, A., Wilson, N.H., 1986. Bus network design. *Transportation Research Part B: Methodological* 20, 331–344.
- Chew, J.S.C., Lee, L.S., Seow, H.V., 2013. Genetic algorithm for biobjective urban transit routing problem. *Journal of Applied Mathematics* 2013.
- De-Los-Santos, A., Canca, D., Barrena, E., 2021. Mathematical formulations for the bi-modal bus-pedestrian social welfare network design problem. *Transportation research part B: methodological* 145, 302–323.
- Desaulniers, G., Hickman, M.D., 2007. Public transit. *Handbooks in operations research and management science* 14, 69–127.
- Dijkstra, E., 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1, 269–271.
- Durán-Micco, J., van Kooten Niekerk, M., Vansteenwegen, P., 2022. Designing bus line plans for realistic cases-the utrecht case study. *Expert Systems with Applications* 187, 115918.
- Durán-Micco, J., Vansteenwegen, P., 2022. A survey on the transit network design and frequency setting problem. *Public Transport* 14, 155–190.
- Fan, L., Chen, H., Gao, Y., 2020. An improved flower pollination algorithm to the urban transit routing problem. *Soft Computing* 24, 5043–5052.
- Fan, L., Mumford, C.L., 2010. A metaheuristic approach to the urban transit routing problem. *Journal of Heuristics* 16, 353–372.

- Fan, L., Mumford, C.L., Evans, D., 2009. A simple multi-objective optimization algorithm for the urban transit routing problem, in: 2009 IEEE Congress on Evolutionary Computation, IEEE. pp. 1–7.
- Farahani, R.Z., Miandoabchi, E., Szeto, W.Y., Rashidi, H., 2013. A review of urban transportation network design problems. *European Journal of Operational Research* 229, 281–302.
- Feng, X., Ruan, Z., Zhu, X., Zhang, L., 2018. Multi-objective transport network design with a reversible simulated annealing algorithm. *Advances in Production Engineering & Management* 13.
- Feng, X., Zhu, X., Qian, X., Jie, Y., Ma, F., Niu, X., 2019. A new transit network design study in consideration of transfer time composition. *Transportation Research Part D: Transport and Environment* 66, 85–94.
- Floyd, R.W., 1962. Algorithm 97: shortest path. *Communications of the ACM* 5, 345.
- Fu, H., Nie, L., Meng, L., Sperry, B.R., He, Z., 2015. A hierarchical line planning approach for a large-scale high speed rail network: the china case. *Transportation Research Part A: Policy and Practice* 75, 61–83.
- Giorgos, D., 2018. Floyd-warshall (vectorized). URL: <https://www.mathworks.com/matlabcentral/fileexchange/67503-floyd-warshall-vectorized>. MATLAB Central File Exchange Accessed: 2 December 2022.
- Goerigk, M., Schmidt, M., 2017. Line planning with user-optimal route choice. *European Journal of Operational Research* 259, 424–436.
- Heyken Soares, P., Mumford, C.L., Amponsah, K., Mao, Y., 2019. An adaptive scaled network for public transport route optimisation. *Public Transport* 11, 379–412.
- Iliopoulou, C., Kepaptsoglou, K., 2019. Integrated transit route network design and infrastructure planning for on-line electric vehicles. *Transportation Research Part D: Transport and Environment* 77, 178–197.
- Iliopoulou, C., Kepaptsoglou, K., Vlahogianni, E., 2019. Metaheuristics for the transit route network design problem: a review and comparative analysis. *Public Transport* 11, 487–521.
- Islam, K.A., Moosa, I.M., Mobin, J., Nayeem, M.A., Rahman, M.S., 2019. A heuristic aided stochastic beam search algorithm for solving the transit network design problem. *Swarm and Evolutionary Computation* 46, 154–170.
- John, M.P., Mumford, C.L., Lewis, R., 2014. An improved multi-objective algorithm for the urban transit routing problem, in: *European Conference on Evolutionary Computation in Combinatorial Optimization*, Springer. pp. 49–60.
- Kechagiopoulos, P.N., Beligiannis, G.N., 2014. Solving the urban transit routing problem using a particle swarm optimization based algorithm. *Applied Soft Computing* 21, 654–676.
- Kirby, R.F., Potts, R.B., 1969. The minimum route problem for networks with turn penalties and prohibitions. *Transportation Research* 3, 397–408.
- Lewis, R., 2020. Algorithms for finding shortest paths in networks with vertex transfer penalties. *Algorithms* 13, 269.

- Liu, D., Durán Micco, J., Lu, G., Peng, Q., Ning, J., Vansteenwegen, P., 2020. A matheuristic iterative approach for profit-oriented line planning applied to the chinese high-speed railway network. *Journal of Advanced Transportation* 2020.
- Liu, Y., Bunker, J., Ferreira, L., 2010. Transit users' route-choice modelling in transit assignment: A review. *Transport Reviews* 30, 753–769.
- López-Ramos, F., 2014. Integrating network design and frequency setting in public transportation networks: a survey. *SORT-Statistics and Operations Research Transactions* , 181–214.
- Lusby, R.M., Larsen, J., Bull, S., 2018. A survey on robustness in railway planning. *European Journal of Operational Research* 266, 1–15.
- Mandl, C., 1979. Applied network optimization. Academic press, London.
- Manser, P., Becker, H., Hörl, S., Axhausen, K.W., 2020. Designing a large-scale public transport network using agent-based microsimulation. *Transportation Research Part A: Policy and Practice* 137, 1–15.
- Mauttone, A., Urquhart, M.E., 2009. A multi-objective metaheuristic approach for the transit network design problem. *Public Transport* 1, 253–273.
- Mumford, C.L., 2013. New heuristic and evolutionary operators for the multi-objective urban transit routing problem, in: 2013 IEEE congress on evolutionary computation, IEEE. pp. 939–946.
- Nayeem, M.A., Rahman, M.K., Rahman, M.S., 2014. Transit network design by genetic algorithm with elitism. *Transportation Research Part C: Emerging Technologies* 46, 30–45.
- Nikolić, M., Teodorović, D., 2013. Transit network design by bee colony optimization. *Expert Systems with Applications* 40, 5945–5955.
- Nnene, O.A., Joubert, J.W., Zuidgeest, M.H., 2019. Transit network design with meta-heuristic algorithms and agent based simulation. *IFAC-PapersOnLine* 52, 13–18.
- Pfetsch, M.E., Borndörfer, R., 2006. Routing in line planning for public transport, in: *Operations research proceedings 2005*. Springer, pp. 405–410.
- Ranjbari, A., Hickman, M., Chiu, Y.C., 2020. A network design problem formulation and solution procedure for intercity transit services. *Transportmetrica A: Transport Science* 16, 1156–1175.
- Schiewe, A., Albert, S., Schiewe, P., Schöbel, A., Spühler, F., Stinzenhöfer, M., 2022. Documentation for lintim 2022.08. URL: <http://nbn-resolving.de/urn:nbn:de:hbz:386-kluedo-69236>.
- Schmidt, M.E., et al., 2014. Integrating routing decisions in public transportation problems. Springer.
- Schöbel, A., 2012. Line planning in public transportation: models and methods. *OR spectrum* 34, 491–510.
- Schöbel, A., Scholl, S., 2006. Line planning with minimal traveling time, in: 5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'05), Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

- Suman, H.K., Bolia, N.B., 2019. Improvement in direct bus services through route planning. *Transport Policy* 81, 263–274.
- Vermeir, E., Durán-Micco, J., Vansteenwegen, P., 2022. The grid based approach, a fast local evaluation technique for line planning. *4OR* 20, 603–635.
- Vermeir, E., Engelen, W., Philips, J., Vansteenwegen, P., 2021. An exact solution approach for the bus line planning problem with integrated passenger routing. *Journal of Advanced Transportation* 2021.
- Wu, P., Xu, L., Che, A., Chu, F., 2020. A bi-objective decision model and method for the integrated optimization of bus line planning and lane reservation. *Journal of Combinatorial Optimization* , 1–30.
- Yao, B., Hu, P., Lu, X., Gao, J., Zhang, M., 2014. Transit network design based on travel time reliability. *Transportation Research Part C: Emerging Technologies* 43, 233–248.
- Yoon, G., Chow, J.Y., 2020. Contextual bandit-based sequential transit route design under demand uncertainty. *Transportation Research Record* 2674, 613–625.