An Enhanced Dynamic Discretization Discovery Algorithm for Continuous-Time Service Network Design Problem

Shengnan Shu[a],Zhou Xu[b],Roberto Baldacci[c]
[a] Department of Civil and Environmental Engineering, National University of Singapore, Singapore, shengnan@nus.edu.sg
[b]Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hong Kong, zhou.xu@polyu.edu.hk
[c] College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar, rbaldacci@hbku.edu.qa

March 3, 2025

Keywords: service network design, continuous time, dynamic discretization discovery, exact algorithm

1 INTRODUCTION

The Service Network Design Problem (SNDP) is a fundamental optimization problem focusing on planning transportation operations for carriers to handle shipments of volumes smaller than capacities of vehicles (Crainic, 2000, Wieberneit, 2008, Crainic & Hewitt, 2021). It is particularly prominent in industries such as parcel and small package delivery as well as Less-Than-Truckload (LTL) freight. In these industries, carriers frequently consolidate multiple shipments into a single vehicle dispatch to enhance cost-effectiveness, increasing vehicle utilization and reducing transportation costs associated with the vehicles. These carriers make use of a network of terminals where shipments can be transferred from incoming to outgoing vehicles, thus enabling the consolidation of multiple shipments onto a single outgoing vehicle. Each shipment is scheduled for pickup at its respective origin terminal at a proper time after it becomes available. It is then transported through the terminal network by following specific routing and consolidation plans, to its destination terminal within a specified time-frame. Shipments may be temporarily stationed at both origin and intermediate terminals along their route, waiting for consolidations.

While the classic SNDP is defined on a planning horizon discretized into a finite number of discrete time units, the Continuous-Time SNDP (CTSNDP) is defined on a continuous-time planning horizon to prevent errors arising from discretization. The CTSNDP poses a significant computational challenge due to the continuous nature of time and the increased complexity of optimization. Existing algorithms for CTSNDP and its variants primarily adopt a Dynamic Discretization Discovery (DDD) solution framework (Boland *et al.*, 2017, Marshall *et al.*, 2021, He *et al.*, 2022, Shu *et al.*, 2024). This framework iteratively refines the discretization, which is a finite set of discrete time units, constructs a partially time-expanded network based on the discretization, and leverages the network to derive relaxations and feasible CTSNDP solutions.

However, existing DDD algorithms for the CTSNDP face some challenges in achieving efficient solutions, particularly for those benchmark instances characterized by a high-cost ratio of vehicle-based costs over flow-based costs and high time flexibility, nearly half of which still need to be solved. The limitations of current DDD algorithms in efficiently solving such instances can be attributed to several factors: (i) a weak relaxation that allows numerous impossible consolidations indicated by the available times and due times of shipments; (ii) the ineffectiveness of heuristic methods for deriving upper-bound solutions; and (iii) the ineffectiveness of the refinement methods that fail to reach a reasonably fine discretization or require excessive iterations.

In this study, we develop an enhanced DDD algorithm for the CTSNDP to address the aforementioned challenges. While the proposed enhanced DDD algorithm shares a similar solution framework with previous DDD algorithms, our study distinguishes itself through three key novel improvements. (i) We introduce a new initial relaxation, based on timed-node-based time-expanded commodity networks for shipments. This involves reduction of decision variables and incorporation of a minimum set of significant time points to the initial discretization, thereby effectively eliminating impossible consolidations from the relaxation model. (ii) We employ a novel MIP-based approach for computing an upper bound at each iteration based on a collection of relaxation solutions. Unlike the heuristic methods applied by Boland *et al.* (2017) and Marshall *et al.* (2021), our EDDD algorithm utilizes an MIP model that aims to find the best CTSNDP solution among all feasible solutions that adhere to the routing plan provided by each relaxation solution. (iii) We introduce a new three-stage method to refine discretization, which involves removing newly identified structural patterns (known as minimum too-long paths) that cause the infeasibility of the relaxation solutions.

The computational study demonstrates that, with the improvements above, our enhanced DDD algorithm exhibits exceptional performance. It solves all benchmark instances of the CTSNDP to optimality within one hour, for the first time in the literature.

2 Problem Description

The CTSNDP is defined as follows. We are given a directed graph $\mathcal{D} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ represents the physical terminal set and set $\mathcal{A}$ indicates the arc set, and a set $\mathcal{K}$ of commodities which must be routed through the network $\mathcal{D}$. In the network $\mathcal{D}$, each arc (i, j) is characterized by its travel time, a per-unit-of-flow cost for each commodity $k \in \mathcal{K}$, a fixed cost for each dispatch of a vehicle on the arc, and a capacity for each vehicle served on the arc. Each commodity $k \in \mathcal{K}$ is defined by a single origin, a single destination, a transportation demand, and a time window defined by its earliest available time at its origin and the due time for arriving at its destination. The demand of each commodity must be delivered from the origin to the destination following a single delivery path or route. Multiple demands can be consolidated to pass through arc $(i, j) \in \mathcal{A}$, reducing the total fixed cost for using the service on arc (i, j). For all practical purposes, we allow the travel times and time window restrictions to be integers in minutes.

The CTSNDP is to determine the routing and consolidation plans for the commodities and the required services/resources to execute them. It aims to minimize total flow and fixed costs while ensuring compliance with time constraints on the commodities. In alignment with Boland *et al.* (2017), we assume that holding shipments at a terminal incurs no additional costs. However, our enhanced method can be easily adapted to the case with non-zero holding costs.

As shown in Boland *et al.* (2017), the CTSNDP can be formulated (approximately) by a time-indexed mixed integer programming (MIP) model based on a time-expanded network $\mathcal{D}^{\Delta}_{\mathcal{T}} = (\mathcal{N}^{\Delta}_{\mathcal{T}}, \mathcal{A}^{\Delta}_{\mathcal{T}} \cup \mathcal{H}^{\Delta}_{\mathcal{T}})$. The accuracy of this modeling approach depends the given discretization parameter Δ. Here, $\mathcal{T} = \{\mathcal{T}_i\}_{i \in \mathcal{N}}$ represents the discretization, where $\mathcal{T}_i = \{0, \Delta, ..., \Delta\lceil \max_{k \in \mathcal{K}} l^k/\Delta \rceil\}$ is a set of discrete time points for each $i \in \mathcal{N}$. The node set $\mathcal{N}^{\Delta}_{\mathcal{T}}$ consists of a node (i, t) for each $i \in \mathcal{N}$ and $t \in \mathcal{T}_i$. The set of arcs in $\mathcal{D}^{\Delta}_{\mathcal{T}}$ includes two subsets of timed-arcs. (i) Holding arcs $\mathcal{H}^{\Delta}_{\mathcal{T}}$: For every terminal $i \in \mathcal{N}$ and every $n \in \{1, ..., n_i - 1\}$, there exists an arc $((i, t^i_n), (i, t^i_{n+1}))$ representing a holding of $(t^i_{n+1} - t^i_n)$ time units at terminal i. (ii) Service arcs $\mathcal{A}^{\Delta}_{\mathcal{T}}$: For every arc $(i, j) \in \mathcal{A}$ and every node $(i, t) \in \mathcal{N}^{\Delta}_{\mathcal{T}}$, there exists an arc $((i, t), (j, \bar{t}))$ with $\bar{t} = t + \Delta\lceil \tau_{ij}/\Delta \rceil$, representing a dispatch from terminal i at time t arriving at time $\bar{t}$ at terminal j.

Let SND($\mathcal{D}^{\Delta}_{\mathcal{T}}$) indicate the time-indexed MIP model of the CTSNDP based on $\mathcal{D}^{\Delta}_{\mathcal{T}}$. Its objective function is to minimize the total cost, calculated as the sum of fixed and flow costs. Its constraints include flow conservation constraints and capacity constraints.

3 Methodology

The optimal solution for the CTSNDP can be obtained by solving the time-indexed MIP model SND($\mathcal{D}^{\hat{\Delta}}_{\mathcal{T}}$) based on the fully time-expanded network $\mathcal{D}^{\hat{\Delta}}_{\mathcal{T}}$ that contains all necessary time points.

However, the size of the fully time-expanded network $\mathcal{D}_{\mathcal{T}}^{\hat{\Delta}}$ can be prohibitively large for practical instances, and solving the resulting SND($\mathcal{D}_{\mathcal{T}}^{\hat{\Delta}}$) using conventional techniques becomes challenging. Therefore, Boland *et al.* (2017) proposed a DDD algorithm, which iteratively expands the size of the time-expanded network. However, existing DDD algorithms for the CTSNDP still face challenges in achieving efficient solutions.

In this study, we propose an enhanced DDD algorithm for the CTSNDP, denoted as EDDD, which follows the DDD framework but incorporates a new initial relaxation model to strengthen the lower bound, a new MIP-based approach to compute the upper bound, and a new three-stage method to refine discretization.

The EDDD algorithm begins by enhancing the initial discretization $\mathcal{T}$ through the incorporation of a minimum set of significant time points to eliminate impossible consolidations. These time points are then utilized to construct the corresponding timed-node-based time-expanded commodity networks $\underline{\mathcal{D}_{\mathcal{T}}^{\mathcal{K}}}$ and the initial relaxation model SND($\underline{\mathcal{D}_{\mathcal{T}}^{\mathcal{K}}}$). Model SND($\underline{\mathcal{D}_{\mathcal{T}}^{\mathcal{K}}}$) is obtained by substituting the network $\mathcal{D}_{\mathcal{T}}^{\Delta}$ with $\underline{\mathcal{D}_{\mathcal{T}}^{\mathcal{K}}}$ in the time-indexed MIP model SND($\overline{\mathcal{D}_{\mathcal{T}}^{\Delta}}$).

Next, the EDDD algorithm solves the relaxation model SND($\underline{\mathcal{D}_{\mathcal{T}}^{\mathcal{K}}}$) iteratively to compute a collection of relaxation solutions in each iteration, which includes the optimal relaxation solution with its objective serving as a valid lower bound for the CTSNDP. For each obtained relaxation solution, an upper bound for the CTSNDP is then derived by solving a new MIP model. This MIP model aims to achieve a feasible CTSNDP solution, aligning with the routing plan derived from the relaxation solution, with the total cost minimized.

Accordingly, our algorithm updates the best lower bound and upper bound obtained. If the two bounds meet a given optimality tolerance, the EDDD algorithm terminates, as an optimal solution to the CTSNDP is found (within the imposed tolerance). Otherwise, we apply a new three-stage method to refine the discretization $\mathcal{T}$ based on the obtained relaxation solutions. It involves adding new time points to $\mathcal{T}$ to eliminate the newly identified structural patterns (referred to as minimum too-long paths) that cause infeasibility of the relaxation solutions.

4 Computational Results and Conclusions

We conducted computational experiments on a PC with a 3.2GHz CPU and a 64 GB RAM to demonstrate the effectiveness and efficiency of the newly proposed EDDD algorithm and its key components. We compare its computational results with the results reported in Marshall *et al.* (2021) for BHMS and MBSH, the two DDD algorithms developed by Boland *et al.* (2017) and Marshall *et al.* (2021), respectively. To further assess the performance of EDDD, we also implemented and tested the DDD algorithm of Boland *et al.* (2017) for the CTSNDP, referred to as IDDD. We evaluated the performance of these different DDD algorithms by solving the same 558 CTSNDP instances used in Marshall *et al.* (2021), which are further categorized according to time flexibility and cost ratio as **HC/LF**", "**HC/HF**", "**LC/LF**" and "**LC/HF**". In the following experiments, we solved each CTSNDP instance to an optimality tolerance of 0.01 with a time limit of one hour for both EDDD and IDDD, as done for BHMS and MBSH.

Table 1 summarizes the computational results. For each group of instances, it displays the number of instances in the group. For each algorithm, it displays the average gap between the best upper bound (UB) and the best lower bound (LB) obtained ("**%Gap**"), i.e., $(UB - LB)/UB \times 100\%$, the average computing time in seconds ("**Time**(s)"), the average number of iterations ("**#Iterations**"), and the percentage of instances solved to optimality ("**%Optimal**") within the given optimality tolerance and time limit.

The results in Table 1 demonstrate that EDDD successfully solves all 558 instances within one hour. The other three methods fail to solve all instances in groups "**HC/LF**" and "**HC/HF**". EDDD achieves this with significantly fewer iterations and shorter computational times. Notably, all 558 CTSNDP instances are solved optimally within five iterations, and for groups "**HC/LF**", "**LC/LF**", and "**LC/HF**", the majority of instances (nearly 80%) are solved optimally within a

Table 1 – *Summary Results on the CTSNDP Instances*

Group	**Algorithm**	**%Gap**	**Times**(s)	**#Iterations**	**%Optimal**
HC/LF	BHMS	0.08	1391.1	5.3	77.1
183	MBSH	0.12	677.8	14.8	85.8
	IDDD	0.99	285.5	13.6	95.6
	EDDD	**0.73**	**9.1**	**1.5**	**100.0**
HC/HF	BHMS	0.56	1966.7	6.0	53.7
177	MBSH	0.84	1693.8	17.5	56.5
	IDDD	2.34	1377.7	14.8	70.1
	EDDD	**0.85**	**131.1**	**2.7**	**100.0**
LC/LF	BHMS	0.00	28.6	3.7	100.0
94	MBSH	0.00	0.6	6.5	100.0
	IDDD	0.71	0.4	3.8	100.0
	EDDD	**0.33**	**0.2**	**1.0**	**100.0**
LC/HF	BHMS	0.00	1.5	2.5	100.0
104	MBSH	0.00	0.1	3.2	100.0
	IDDD	0.51	0.1	1.3	100.0
	EDDD	**0.08**	**0.1**	**1.0**	**100.0**

single iteration. Moreover, take the algorithm IDDD as a baseline algorithm, in the first iteration, EDDD achieves maximum improvements in the lower and upper bounds of 22.92% and 37.70%, respectively. Additionally, the percentage ratio between the number of time points in the final discretization produced by EDDD and the complete discretization is less than approximately 4%. These highlight the superior performance of EDDD compared to other existing DDD methods for the CTSNDP, emphasizing the effectiveness of the new initial relaxation, the new MIP-based method for the upper bound, and the new three-stage refinement method.

In summary, this study introduced an enhanced dynamic discretization discovery (EDDD) algorithm to solve the continuous-time service network design problem (CTSNDP). The enhanced DDD algorithm follows a dynamic framework as in previous DDD algorithms but stands out through innovative components. It exhibits exceptional performance, achieving optimality within one hour for all classic CTSNDP instances. In contrast, existing DDD algorithms struggle to solve nearly half of the high-cost-ratio and high-time-flexibility instances. These innovative components provide a solid foundation for enhancing DDD algorithms for different variants of the CTSNDP and various other transportation problems.

References

Boland, Natashia, Hewitt, Mike, Marshall, Luke, & Savelsbergh, Martin. 2017. The continuous-time service network design problem. *Operations Research*, **65**(5), 1303–1321.

Crainic, Teodor Gabriel. 2000. Service network design in freight transportation. *European Journal of Operational Research*, **122**(2), 272–288.

Crainic, Teodor Gabriel, & Hewitt, Mike. 2021. Service network design. *Network Design with Applications to Transportation and Logistics*, 347–382.

He, Edward, Boland, Natashia, Nemhauser, George, & Savelsbergh, Martin. 2022. An exact algorithm for the service network design problem with hub capacity constraints. *Networks*, **80**(4), 572–596.

Marshall, Luke, Boland, Natashia, Savelsbergh, Martin, & Hewitt, Mike. 2021. Interval-based dynamic discretization discovery for solving the continuous-time service network design problem. *Transportation Science*, **55**(1), 29–51.

Shu, Shengnan, Xu, Zhou, & Baldacci, Roberto. 2024. Incorporating holding costs in continuous-time service network design: new Model, relaxation, and exact Algorithm. *Transportation Science*, **58**(2), 412–433.

Wieberneit, Nicole. 2008. Service network design for freight transportation: a review. *OR Spectrum*, **30**(1), 77–112.