The Time-Varying Backward Hyperpath with the Maximum Capacity in the Backward Hypergraph

Zhile Xing, Shurong Zhang*, Lin Chen

Abstract—**With the rapid development of information technology, the hypergraph has gradually become an important model that can truly reflect the complex network. Especially time-varying directed hypergraphs can better represent dynamic networks and have more important research significance. As an important class of directed hypergraphs, barkward hypergraphs (BHs) have a variety of applications in urban traffic and information transmission. The problem of finding the time-varying backward hyperpath (TV-B-hyperpath) with the maximum capacity in the time-varying BH (TV-BH) is the core of many network optimization problems. However, due to the special structure of TV-BH and TV-B-hyperpath, this problem has not yet been investigated. In this paper, we consider a model of weighted TV-BH. Then, we propose polynomial algorithm to find maximum capacity TV-B-hyperpath in the weighted TV-BH. Moreover, in order to further reduce the complexity of algorithm, we use the time discretization and hypergraph pruning techniques. Simulation results show that, by using the two methods, the computation time of the algorithm are significantly reduced.**

Index Terms—**The maximum capacity, time-varying networks, backward hypergraphs, hyperpath optimization, time discretization.**

I. Introduction

As the core of many network optimization problems, the maximum capacity path problem (MCP) was proposed by Pollack [19] as early as 1960 and has widely been used to tackle a number of network problems [6], [14], [25]. Specifically, the MCP problem is to find a path between two given nodes such that the capacity of the path is maximal, where the capacity of a path is defined as the minimum capacity of edges and nodes on this path. Gabow et al. proved in [12], [16], [20] that the MCP problem is polynomially solvable in static networks, where the weights of each arc are static constants. However, many real-life problems tend to be dynamic in nature. For example, the dynamic management of a traffic system provides real-time traffic information to users, the channel bandwidth between stations in the communication network is time-dependent. Therefore, scholars began to study the time-varying maximum capacity problem (TVMCP), in which the transmission time and capacity of each arc in the network change dynamically over time. At the same time, the TVMCP may appear as a sub-problem in the process of solving other time-varying network optimization problems, such as the time-varying maximum flow problem [10] and the problem of improving network transmission capacity [9].

Manuscript received December 7, 2022; revised March 20, 2023. This work was supported by the Natural Science Foundation of Shanxi Province (202103021224058).

Z. Xing is a postgraduate student of Taiyuan University of Technology, Taiyuan, Shanxi 030024, P. R. China. (e-mail: 1634761743@qq.com).

S. Zhang is a professor of College of Mathematics, Taiyuan University of Technology, Taiyuan, Shanxi 030024, P. R. China. (e-mail: zhangshurong@tyut.edu.cn).

L. Chen is a professor of School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou, Guangdong 510275, P. R. China. (e-mail: chenlin021390@163.com)

* Corresponding author (e-mail: zhangshurong@tyut.edu.cn)

MCP and TVMCP are usually solved in general graph. In fact, the real network is a complex system composed of multiple subjects and relationships. Different system structures exhibit different properties between nodes and arcs. Then, the general graph cannot fully describe the network characteristics of the real world. In particular, in real life, the relationships between objects are often complex. For example, the general graph can indicate whether authors have collaborated, but it cannot indicate whether three or more authors have collaborated on a paper. Then solving this kind of problem can only be done in the hypergraph. As early as the 1960s, hypergraphs became an independent theory [8].

Originally, developed in France by Berge [4], the hypergraph is a general structure in discrete mathematics. Furthermore, Gallo et al. [13] gave the concept of directed hypergraphs. It can be used to deal with specific problems in computer science and combinatorial optimization. Moreover, the backward hypergraph (BH), in which each hyperarc has only one head node and multiple tail nodes, is an important kind of directed hypergraph. This topology is widely used in many practical problems [21], [23]. For example, in the communication network, the central node (head node) in each area needs to receive data information from all other nodes (tail nodes) in this area before data can be forwarded. Due to the limitation of channel bandwidth (capacity) between nodes, the problem of how to transmit more information to the destination within the specified time motivates us consider the maximum capacity backward hyperpath (B-hyperpath) problem. Since the communication network is dynamic in practical applications, we mainly consider the time-varying backward hypergraph (TV-BH). Therefore, we model the weighted TV-BH and solve the maximum capacity TV-B-hyperpath problem in this TV-BH.

A. *Related work.*

In the general graph G, the MCP problem was proposed by Pollack in 1960 [19]. The well-known algorithm used to solve this problem was proposed by Gabow [12] with the complexity of $O(\min\{m + nlogn, mlog_n W\})$, where W is the maximum edge weight of the graph G. Punnen [20] used a binary search method to propose an algorithm with the complexity of $O(m)$, where m is the number of edges in G. In [16], Kaibel et al. gave an algorithm with the complexity of $O(m)$ in undirected graphs, and proposed a modified Dijkstra algorithm with bucket structure in directed graphs to solve the MCP problem. When the weights associated with the edges in the graph change over time, the problem has been studied in [10] and shown to be NP-complete. The authors in [10] used the approach of time discretization to propose approximate algorithms. When the capacities are considered to be the generalized trapezoidal fuzzy number and no waiting is allowed at any node, an exact algorithm for finding the optimal path within a time limit D was proposed in [22]. Therefore, we can know that different time-varying weights associated with edges will affect the complexity of the problem, and the approach of time discretization is an important method to solve time-dependent problems.

The directed hypergraph, a generalization of the directed graph [7], has been widely and deeply studied in [1], [3], [4], [5], [13], [24] and quite often been used in various area of computer science and discrete mathematics as a modelling and algorithmic tool. However, the backward hyperpath in directed hypergraph is not a trivial extension of directed path. In the work of Ausiello et al. [2], the study of the shortest backward hyperpath problem for directed hypergraphs has been approached for a variety of cost measures. Ausiello et al. [3] introduced weighted directed hypergraphs and given different calculation methods about the cost of backward hyperpath. In [13], the authors porposed various algorithms in weighted hypergraphs. These algorithms are all based on static hypergraph. To the authors' knowledge, the problem of finding the maximum capacity TV-B-hyperpath in the TV-BH has not been considered. Therefore, we will study this problem in this paper.

B. Our Contributions.

The main contributions of this paper are as follows.

- **Mathematical model of TV-BH.** We model a weighted TV-BH. All channels in each area may have approximately the same transmission time and bandwidth. Then this case is modeled as the weighted TV-BH $\mathscr{H}^1$, where each hyperarc is assigned a delay and a time-varying capacity function.
- **Maximum capacity TV-B-hyperpath problems.** For the problem of information transmission in time-varying communication networks, the channel bandwidth (capacity) between nodes varies with time, and the central nodes (head node) in each area (hyperarc) must receive all information from all other nodes (tail nodes) in the area before data can be forwarded. Then, the information is sent to the central node only after all other nodes have received the information. This is defined as a Capacity Model for calculating the capacity of a TV-B-hyperpath in $\mathscr{H}^1$. According to this model, we propose a polynomial algorithm to find the maximum capacity TV-B-hyperpath in TV-BH.
- **The main technical methods in the algorithms.** Firstly, we propose polynomial algorithm by using the method of time discretization. Then, we prune the TV-BH by using the bidirectional search and update the already discretized time set by removing times that are unreachable at any node. These methods are able to optimize our algorithm. In general graphs, many scholars have used bidirectional Dijkstra Algorithm, bidirectional A* Algorithm, and bidirectional ALT Algorithm to find the shortest path [11], [15], [17], [18]. These studies have shown that bidirectional search can reduce the search space and thus reduce the operation time of the algorithm. Therefore, in this paper, the method is applied to prune the TV-BHs, which significantly speeds up the running speed of our algorithms. Then the effectiveness of the algorithms are compared and analyzed by simulation experiments.

The rest of the paper is organized as follows. In section II, we define the weighted TV-BH and develop the Capacity Model for calculating the capacity of the TV-B-hyperpath. Then we propose the mathematical model of the main problem. In Sections III, we design algorithms to find maximum capacity TV-B-hyperpath in above weighted TV-BH, and analyze their correctness and complexity. Then, we give experiments in Section IV. Finally, we conclude this paper in Section V.

II. NETWORK TOPOLOGY AND PROBLEM FORMULATION

A. Directed hypergraph.

For hypergraph-theoretical terminologies and notations defined here we follow [13]. A *hypergraph* is an ordered pair $\mathscr{H} = (\mathscr{V}, \mathscr{E})$, where $\mathscr{V}$ is the set of nodes, and $\mathscr{E}$ is a family of subsets of $\mathscr{V}$. The elements of $\mathscr{E}$ are called hyperedges. For each hyperedge $E \in \mathscr{E}$, if E is divided into two nonempty subsets $T(E)$ and $H(E)$, then E is a dircted hyperedge from $T(E)$ to $H(E)$, where $T(E)$ denotes the set of *tail nodes* and $H(E)$ denotes the set of *head nodes*. Then E is called a *hyperarc* and is noted as $(T(E), H(E))$. Furthermore, $\mathscr{H}$ is called the *directed hypergraph* (DH). The *cardinality* of a hyperarc $E \in \mathscr{E}$ is the number of nodes it contains, denoted by $|E|$. When $|E| = 2$ for each $E \in \mathscr{E}$, the DH $\mathscr{H}$ is a general directed graph. The *size* of $\mathscr{H}$ is $size(\mathscr{H}) = \sum\limits_{E\in\mathscr{E}} |E|$.

Define a hyperarc $E = (T(E), H(E))$ with $|H(E)| = 1$ as a *backward arc* (B-arc). A *backward hypergraph*, or simply BH, is a directed hypergraph whose hyperarcs are all B-arcs. An example of a BH is illustrated in Fig. 1, which contains 10 nodes $v_1, v_2, \ldots, v_{10}$ and 16 B-arcs $E_1, E_2, \ldots, E_{16}$, and B-arcs E_1 and E_4 have 1 and 3 tail nodes, respectively.

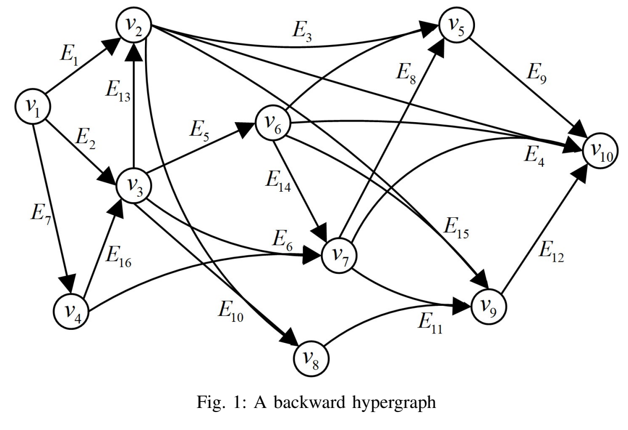


Fig. 1: A backward hypergraph

Let $FS(v) = \{E \in \mathscr{E} | v \in T(E)\}$ and $BS(v) = \{E \in \mathscr{E} | v \in H(E)\}$ be the forward star and backward star of v, respectively. In a DH $\mathscr{H}$, a *hyperpath* $P_{v_s v_t}$ of length q is a sequence of nodes and hyperarcs:

$$P_{v_s v_t} = (v_s = v_1, E_1, v_2, E_2, \ldots, E_q, v_{q+1} = v_t)$$

where $E_i \in \mathscr{E}$ for $i = 1, 2, \ldots, q$, $v_s \in T(E_1)$, $v_t \in H(E_q)$ and $v_i \in H(E_{i-1}) \cap T(E_i)$ for each $i \in [2, q]$. Node v_t is connected to node v_s in $\mathscr{H}$ if a hyperpath $P_{v_s v_t}$ exists. If $v_t \in T(E_1)$, then $P_{v_s v_t}$ is a cycle in $\mathscr{H}$. The hyperpath $P_{v_s v_t}$ is *cycle-free* if it does not contain any sub-hyperpath which is a cycle. If $\mathscr{H}$ contains no cycles, it is called *acyclic*.

B. The TV-BH and a transmission model.

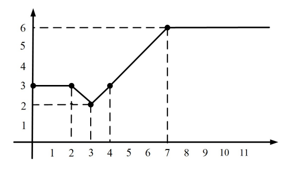
As explained in the Intruduction, we are interested in the TV-BH, which is defined as follows.

Definition 1 (**TV-BH** $\mathscr{H}^1$)**.** *Given a BH $\mathscr{H} = (\mathscr{V}, \mathscr{E})$ with a node set $\mathscr{V} = \{v_1, v_2, \ldots, v_n\}$ and a B-arc set $\mathscr{E} = \{E_1, E_2, \ldots, E_m\}$, then the TV-BH with time-varying B-arc capacity is defined by a quintuple $(\mathscr{V}, \mathscr{E}, T^*, \Delta, \Lambda)$, where*

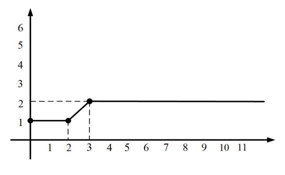
- *T^* is the time horizon;*
- *$\Delta = \{d(E_i) : E_i \in \mathscr{E}\}$ and $d(E_i) \in \mathbb{N}^+$ is the delay to traverse each $E_i \in \mathscr{E}$;*
- *$\Lambda = \{c_i(E_i, t) : E_i \in \mathscr{E}\}$ and $c_i(E_i, t) > 0$ is the time-varying capacity function for each $E_i \in \mathscr{E}$, where $t \in [0, T^*]$.*

The characteristic of time-varying capacity functions in TV-BH $\mathscr{H}^1$ is that each B-arc is assigned exactly one capacity function. For the BH $\mathscr{H}$ given in Fig. 1, in following example, we will define the weight function for each B-arc in $\mathscr{H}$ and obtain a $\mathscr{H}^1$.

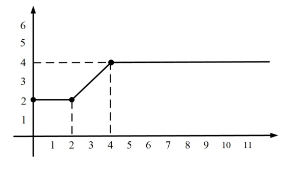
Example 1. Observe that the hypergraph $\mathscr{H}$ in Fig. 1 is a TV-BH with 10 nodes and 16 B-arcs. The time-varying capacity functions of some B-arcs E_1, E_5, E_6, E_8, E_9, E_{10}, E_{11}, E_{12}, E_{13}, E_{14}, E_{15} are shown in Fig. 2 (a)-(d) and the time-varying capacity functions of other B-arcs are defined as the following constant functions: $c_2(E_2, t) = c_3(E_3, t) = c_7(E_7, t) = c_{16}(E_{16}, t) = 3$, $c_4(E_4, t) = 5$, where $t \in [0, T^*]$. The delay of each B-arc is set as follows: $d(E_1) = 2$, $d(E_2) = 1$, $d(E_i) = 2$ for $i = 3, 4, \ldots, 15$, $d(E_{16}) = 1$. Then, the TV-BH $\mathscr{H}^1$ can be obtained.



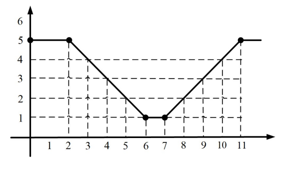
(a) The time-varying capacity functions of B-arcs $E_1, E_5, E_{12}, E_{13}, E_{15}$



(b) The time-varying capacity functions of B-arcs E_6 and E_8



(c) The time-varying capacity functions of B-arcs E_9 and E_{11}



(d) The time-varying capacity functions of B-arcs E_{10} and E_{14}

Fig. 2: The time-varying capacity functions of some B-arcs

In this paper, we mainly consider the case that the time-varying capacity functions are piecewise linear functions. Based on the application background from the Introduction, we can know the way of information transmission in $\mathscr{H}^1$. In the following, we describe the model of this information transmission.

• [**Capacity Model**] For any TV-BH $\mathscr{H}^1$ and $E_i = (T(E_i), \{v\}) \in \mathscr{E}$, without loss of generality, suppose E_i has two tail nodes v_j and v_k, which receive the information at times t_1 and t_2, respectively, where $t_1 < t_2$. So v_j receives the information first. At this time it cannot transmit the information to v immediately and needs to wait until time t_2. That is v_j waits until the other tail node v_k has received the information. At time t_2, v_j and v_k start transmitting information to v. The time to arrive at v is $\max\{t_1, t_2\} + d(E_i)$.

C. Time-varying backward hyperpath.

According to the above transmission model, for any B-arc $E = (T(E), \{v\})$, v can receive information only if the information will be transmited to all tail nodes in $T(E)$. This leads us to focus on the backward hyperpath which is a special kind of the hyperpath and is defined as follows.

Definition 2 (**B-hyperpath [13]**)**.** *Given a DH $\mathscr{H} = (\mathscr{V}, \mathscr{E})$ with a source v_s and a sink v_t, a backward hyperpath (B-hyperpath) $\pi_{v_s v_t}$ is a DH $(\mathscr{V}_\pi, \mathscr{E}_\pi)$ with the least number of hyperarcs and satisfying the following conditions:*

- *$\mathscr{E}_\pi \subseteq \mathscr{E}, \mathscr{V}_\pi = \bigcup_{E \in \mathscr{E}_\pi} E \subseteq \mathscr{V}$;*
- *$v_s, v_t \in \mathscr{V}_\pi$;*
- *for each $v \in \mathscr{V}_\pi$, $\pi_{v_s v_t}$ has a cycle-free hyperpath from v_s to v.*

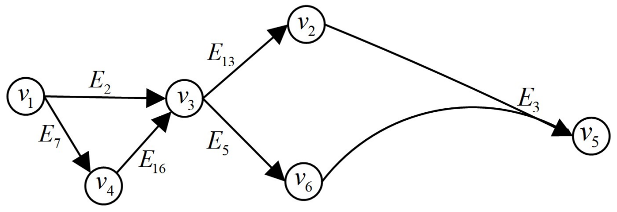


Fig. 3: BH $\mathscr{H}$

Given a BH $\mathscr{H}$ in Fig. 3, then a TV-BH $\mathscr{H}^1$ can be obtained. There are two hyperpaths $P_{v_1 v_2} = (v_1, E_2, v_3, E_{13}, v_2)$ and $P_{v_1 v_6} = (v_1, E_7, v_4, E_{16}, v_3, E_5, v_6)$ with departure time t_1. Assume that, passing through $P_{v_1 v_2}$, the times of arriving nodes v_3 and v_2 are t_3 and t_2, respectively. Moreover, passing through $P_{v_1 v_6}$, the times of arriving at nodes v_4, v_3 and v_6 are t_4, t_3' and t_6, respectively. Note that E_2, E_5, E_7, E_{13} and E_{16} are general edges. Then $P_{v_1 v_2}$ and $P_{v_1 v_6}$ are general paths as well. Combining this with the definition of the time-varying path capacity [10], suppose that $P_{v_1 v_2}$ and $P_{v_1 v_6}$ are the maximum capacity paths arriving at v_2 and v_6, respectively. Assume that $t_6 > t_2$ and the minimum capacity value of B-arc E_3 in time period $[t_6, t_6 + d(E_3))$ is larger than any other time periods. Then, if the information start from v_1 at time t_1 and pass through $\mathscr{H} = P_{v_1 v_2} + P_{v_1 v_6} + E_3$, then the amount of information transmitted to v_5 can be maximized, which leads to reaching v_3 twice at times t_3 and t_3' passing through $P_{v_1 v_2}$ and $P_{v_1 v_6}$, respectively. Then, we can observe that $\mathscr{H}$ is not a B-hyperpath because it contradicts the minimality in the definition of the B-hyperpath. In addition, if v_3 at times t_3 and t_3' are seemed as two different nodes, then $P_{v_1 v_2}$ and $P_{v_1 v_6}$ are two internally disjoint paths and $\mathscr{H}$ becomes a

B-hyperpath. This will be explained in Example 2. Now we give the useful terminologies as follows. After that, we will propose an appropriate general definition of the time-varying backward hyperpath.

In a TV-BH, similar to the node v_3 in Fig. 3, each node v may have multiple arrival times passing through different hyperpaths. We denote the set of times when the information can arrive at v by T_v. For any $t \in T_v$, the B-arc $E \in BS(v)$ that enables the information to pass through E and arrive node v at time t is called a *B-arc of v at time t*. We denote the set of all these B-arcs by $BS(v,t)$. Therefore, for each $t \in T_v$, we have a two-tuples $(t, BS(v,t))$.

Next we can construct the *corresponding BH* $\mathscr{H}'$ of a TV-BH $\mathscr{H}^1$ as follows. Firstly, for each node v with $T_v = \{t_1, t_2, \ldots, t_r\}$ in $\mathscr{H}^1$ for some $r \geq 1$, we replace the node v with $|T_v| = r$ copies which are denoted by $v(t_1), v(t_2), \ldots, v(t_r)$, respectively. For any B-arc $E \in BS(v,t_i)$, the head node of E (i.e. v) is replaced by $v(t_i)$. Then the corresponding BH $\mathscr{H}'$ is obtained and an example is as follows.

Example 2. Recall that in the $\mathscr{H}$ given in Fig. 3, when the departure time is t_1, the hyperpaths $P_{v_1v_2}$ and $P_{v_1v_6}$ pass through v_3 at times t_3 and t'_3, respectively. Then $T_{v_3} = \{t_3, t'_3\}$ and we discretize v_3 into two nodes $v_3(t_3)$ and $v_3(t'_3)$. For each node v_i, where $i \in \{1,2,4,5,6\}$, since $|T_{v_i}| = 1$ and $T_{v_i} = \{t_i\}$, each v_i is replaced by the node $v_i(t_i)$. Then, the set of nodes of the corresponding BH $\mathscr{H}'$ is obtained and we can note that the B-arcs E_2, E_7, E_{16}, E_3 are replaced by $(\{v_1(t_1)\}, \{v_3(t_3)\})$, $(\{v_1(t_1)\}, \{v_4(t_4)\})$, $(\{v_4(t_4)\}, \{v_3(t'_3)\})$ and $(\{v_2(t_2), v_6(t_6)\}, \{v_5(t_5)\})$, respectively. As the time passing through v_3 along $P_{v_1v_2}$ is t_3 and the path $P_{v_1v_6}$ passes through E_5 in the time period $[t'_3, t_6]$, E_{13} and E_5 are replaced with $(\{v_3(t_3)\}, \{v_2(t_2)\})$ and $(\{v_3(t'_3)\}, \{v_6(t_6)\})$, respectively. Then the corresponding BH $\mathscr{H}'$ of TV-BH $\mathscr{H}^1$ defined based on BH $\mathscr{H}$ in Fig. 3 is constructed and is shown in Fig. 4. We can observe that $\mathscr{H}'$ is a static B-hyperpath which has the maximum information capacity from v_1 to v_5 with departure time t_1 although $\mathscr{H}$ in Fig. 3 is not a B-hyperpath. Therefore, we need to propose the definition of time-varying B-hyperpath to maximize capacity.

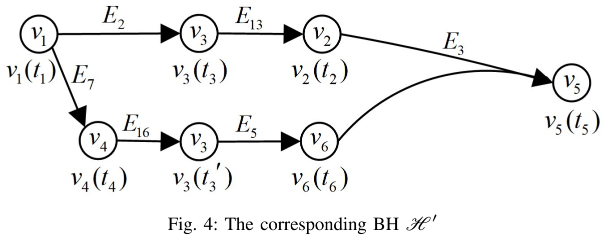


Fig. 4: The corresponding BH $\mathscr{H}'$

Definition 3 (**TV-B-hyperpath**). *Given a TV-BH* $\mathscr{H}^1 = (\mathscr{V}, \mathscr{E}, T^*, \Delta_1, \Lambda_1)$ *with a source* v_s *and a sink* v_t*, the departure time is* t_0*. If the corresponding BH of* $\mathscr{H}^1$ *is a B-hyperpath from* v_s *to* v_t*, then* $\mathscr{H}^1$ *is called a time-varying B-hyperpath (or, for short, TV-B-hyperpath) and is denoted by* $\Pi_{v_sv_t}$.

If there is a TV-B-hyperpath $\Pi_{v_sv_t}$ from v_s to v_t in a TV-BH, then v_t is reachable. Let the departure time and arrival time of $\Pi_{v_sv_t}$ be t_0 and t, respectively. We define the delay of $\Pi_{v_sv_t}$ as $D(\Pi_{v_sv_t}, t) = t - t_0$. For any node v in $\Pi_{v_sv_t}$, let $T_v(\Pi_{v_sv_t})$ be the set of arrival times of node v in $\Pi_{v_sv_t}$. Since the corresponding BH of $\Pi_{v_sv_t}$ is a B-hyperpath, according to the minimality of the B-hyperpath, there is exactly one TV-B-hyperpath in $\Pi_{v_sv_t}$ from v_s to v at each time $t \in T_v(\Pi_{v_sv_t})$ and it is called a TV-B-subhyperpath which is denoted by $\Pi_{v_sv}(t)$. Similarly, following result can be obtain directly.

Observation 1. *For any* v *in a TV-B-hyperpath* $\Pi_{v_sv_t}$*, if* $t \in T_v(\Pi_{v_sv_t})$*, then there is only one B-arc* $E \in BS(v,t)$ *in* $\Pi_{v_sv_t}$*. Furthermore, for each* $w \in T(E)$*,* $T_w(\Pi_{v_sv}(t))$ *has exactly one time.*

D. The capacity of a TV-B-hyperpath and problem formulation.

For any BH $\mathscr{H}$, since the capacity function in TV-BH $\mathscr{H}^1$ is time-varying, the minimum capacity value of the B-arc is time dependent. So we firstly give following notation.

In the TV-BH $\mathscr{H}^1$, for any $E_i = (T(E_i), \{v\}) \in \mathscr{E}$, the minimum capacity value of the B-arc E_i in time period $[t_1, t_2]$ is $l(E_i, t_1, t_2) = \min\limits_{t \in [t_1,t_2]} \big\{c_i(E_i, t)\big\}$.

Example 3. As shown in Fig. 2, we have:

$$l(E_1, 2, 4) = \min_{t \in [2,4]} \big\{c_1(E_1, t)\big\} = 2.$$

Next, combining the Capacity Model and Observation 1, we will give definition of the capacity of the TV-B-hyperpath $\Pi_{v_sv_t}$ in $\mathscr{H}^1$.

Definition 4 ($C(\Pi_{v_sv_t})$)**.** *Given a TV-B-hyperpath* $\Pi_{v_sv_t}$ *from* v_s *to* v_t *at time exactly* t^**, the capacity of* $\Pi_{v_sv_t}$ *is denoted as* $C(\Pi_{v_sv_t})$*. For each node* v *in* $\Pi_{v_sv_t}$ *and* $t \in T_v(\Pi_{v_sv_t})$*, the capacity of the TV-B-subhyperpath* $\Pi_{v_sv}(t)$ *is denoted by* $C(\Pi_{v_sv}(t))$*. Let* $C(\Pi_{v_sv_s}) = \infty$*. Then the iterative formula for computing* $C(\Pi_{v_sv_t})$ *is* $C(\Pi_{v_sv_t}) = \min\Big\{\min\limits_{v_k \in T(E_j)} \big\{C(\Pi_{v_sv_k}(t_k))\big\}, l(E_j, t - d(E_j), t)\Big\}$*, where* E_j *is the only one B-arc in both* $BS(v_t, t^*)$ *and* $\Pi_{v_sv_t}$*, and for each* $v_k \in T(E_j)$*, let* $T_{v_k}(\Pi_{v_sv_t}) = \{t_k\}$.

Next, we formulate an optimization problem about the time-varying maximum capacity B-hyperpath (TV-MCBH) in $\mathscr{H}^1$.

Problem 1 (TV-MCBH I)**.** *Given a TV-BH* $\mathscr{H}^1$*, a source* v_s*, a sink* v_t*, and a positive integer* D*, we want to find the maximum capacity TV-B-hyperpath* $\Pi_{v_sv_t}$ *with departure time* t_0 *from* v_s *to* v_t *in* $\mathscr{H}^1$ *such that the delay is not more than* D.

III. Algorithm Design for TV-MCBH I

In this section, we will discuss how to find the TV-MCBH in $\mathscr{H}^1 = (\mathscr{V}, \mathscr{E}, T^*, \Delta, \Lambda)$, where $\mathscr{E} = \{E_1, E_2, \ldots, E_m\}$ is the set of B-arcs in $\mathscr{H}^1$. Firstly, according to the Δ in Definition 1 and the D in Problem 1, we discretize the interval $[t_0, t_0+D]$ into a set $\mathscr{T} = \{t_0, t_0+1, t_0+2, \ldots, t_0+D\}$. For any $t \in \mathscr{T}$ and $v \in \mathscr{V}$, if there exists $E_j \in BS(v,t)$, then the latest time to reach the set of tail nodes is $F_j = t - d(E_j)$. The maximum capacity of the TV-B-hyperpath from v_s to v at time exactly t is denoted by $\xi(v,t)$. In order to calculate $\xi(v,t)$, we give the following theorem.

Theorem 1. *Given a TV-BH $\mathscr{H}^1$, for any $t \in \mathscr{T}$ and $v_i \in \mathscr{V}$, assume that there is at least one B-arc $E_j \in BS(v_i, t)$, let*

$$\xi_{v_k} = \max_{t' \le t-d(E_j)} \{\xi(v_k, t')\}, \tag{1}$$

$$\xi_j(v) = \min\{\xi(v, t-d(E_j)), \min_{v_j \in T(E_j)\setminus\{v\}} \xi_{v_j}\}, \quad \textit{for each } v \in Q, \tag{2}$$

$$\xi_{T(E_j)} = \max_{v \in Q}\{\xi_j(v)\}, \tag{3}$$

where Q is the set of nodes in $T(E_j)$ that can be reached at time $t - d(E_j)$. Then, the maximum capacity of the TV-B-hyperpath from v_s to v_t at time exactly t is

$$\xi(v_i, t) = \max_{E_j \in BS(v_i,t)} \min\{\xi_{T(E_j)}, l(E_j, t-d(E_j), t)\}. \tag{4}$$

Proof: First, we claim that there exists a TV-B-hyperpath from v_s to v_i at time exactly t with capacity $\xi(v_i, t)$. Without loss of generality, let $E_j = argmax\{\min\{\xi_{T(E)}, l(E, t-d(E), t) : E \in BS(v_i, t)\}\}$ (see Fig. 5 (a)) and let $v_l = argmax\{\xi_j(v) : v \in Q\}$. Then, by Eq. (2) and Eq. (3),

$$\begin{aligned}\xi_{T(E)} &= \xi_j(v_l)\\ &= \min\{\xi(v_l, t-d(E_j)), \min_{v_j \in T(E_j)\setminus\{v_l\}} \xi_{v_j}\}\end{aligned} \tag{5}$$

Furthermore, by the existence of E_j and the definition of TV-B-hyperpath, there is a TV-B-hyperpath $\Pi_{v_s v_k}(t_k)$ from v_s to each $v_k \in T(E_j)\setminus\{v_l\}$ at some time $t_k \le t - d(E_j)$ with capacity ξ_{v_k}. By the choice of v_l, there exists a TV-B-hyperpath $\Pi_{v_s v_l}(t - d(E_j))$ from v_s to v_l at time $t - d(E_j)$ with capacity $\xi(v_l, t - d(E_j))$.

Let the union of all TV-B-hyperpaths in the set $P = \{\Pi_{v_s v_l}(t - d(E_j))\} \cup \{\Pi_{v_s v_k}(t_k) : v_k \in T(E_j)\setminus\{v_l\}\}$ be the TV-BH $\Pi(v_s \to T(E_j))$. Set $t_l = t - d(E_j)$. That is $\Pi_{v_s v_l}(t - d(E_j))$ can be rewritten as $\Pi_{v_s v_l}(t_l)$. Therefore, the minimum capacity of TV-B-hyperpaths in P is

$$\begin{aligned}&\min_{v_k \in T(E_j)} C(\Pi_{v_s v_k}(t_k))\\ &= \min\{C(\Pi_{v_s v_l}(t_l), \min_{v_k \in T(E_j)\setminus\{v_l\}} C(\Pi_{v_s v_k}(t_k)))\}\\ &= \min\{\xi(v_l, t-d(E_j)), \min_{v_k \in T(E_j)\setminus\{v_l\}} \xi_{v_k}\}.\end{aligned}$$

Then, combining this with Eq. (5), we have that

$$\min_{v_k \in T(E_j)} C(\Pi_{v_s v_k}(t_k)) = \xi_{T(E_j)}. \tag{6}$$

According to the definition of TV-B-hyperpath, it can be seen that the TV-BH consisting of $\Pi(v_s \to T(E_j))$ and E_j is a TV-B-hyperpath from v_s to v_i at time t and it is denoted by $\Pi_{v_s v_i}$. The node v_l is called the fixed node of E_j in $\Pi_{v_s v_i}$. Now, by Definition 4 and Eq. (6), we can see that $C(\Pi_{v_s v_i}) = \min\{\min_{v_k \in T(E_j)} C(\Pi_{v_s v_k}(t_k)), l(E_j, t - d(E_j), t)\} = \min\{\xi_{T(E_j)}, l(E_j, t-d(E_j), t)\}$.

Then, the choice of E_j implies that

$$C(\Pi_{v_s v_i}) = \max_{E_j \in BS(v_i,t)} \min\{\xi_{T(E_j)}, l(E_j, t-d(E_j), t)\}$$

Therefore, the claim holds.

The above proof yields a TV-B-hyperpath $\Pi_{v_s v_i}$ from v_s to v_i at time t and $C(\Pi_{v_s v_i}) = \xi(v_i, t)$. In the following, it's enough to prove that for any other TV-B-hyperpath $\Pi^*_{v_s v_i}$ from v_s to v_i at time t, $C(\Pi^*_{v_s v_i}) \le \xi(v_i, t)$. Now, this will be proved by induction on the number of B-arcs of the TV-B-subhyperpath in $\Pi^*_{v_s v_i}$. According to the Definitions 2 and 3, for each v_j in $\Pi^*_{v_s v_i}$ and $t_j \in T_{v_j}(\Pi^*_{v_s v_i})$, there is only one TV-B-hyperpath $\Pi^*_{v_s v_j}(t_j)$ in $\Pi^*_{v_s v_i}$ from v_s to v_j at time t_j (see Fig. 5 (b)).

First, the result holds obviously when $v_i = v_s$. The only one B-arc of $BS(v_i, t)$ in $\Pi^*_{v_s v_i}$ is denoted by E_k. By the Capacity Model, without loss of generality, let $t_{l^*} = t - d(E_k)$. That is v_{l^*} is the fixed node of E_k in $\Pi^*_{v_s v_i}$. Observe that $\Pi^*_{v_s v_i}$ is the union of all TV-B-hyperpaths in the set $P^* = \{\Pi^*_{v_s v_{l^*}}(t - d(E_k))\} \cup \{\Pi^*_{v_s v_j}(t_j) : v_j \in T(E_k)\setminus\{v_{l^*}\}\}$.

By induction, we have that $C(\Pi^*_{v_s v_{l^*}}(t_{l^*})) = C(\Pi^*_{v_s v_{l^*}}(t - d(E_k))) \le \xi(v_{l^*}, t - d(E_k)) = \xi(v_{l^*}, t_{l^*})$ and $C(\Pi^*_{v_s v_j}(t_j)) \le \xi(v_j, t_j)$. Hence,

$$\begin{aligned}&\min_{v_j \in T(E_k)}\{C(\Pi^*_{v_s v_j}(t_j))\}\\ &\le \min\{\xi(v_{l^*}, t_{l^*}), \min_{v_j \in T(E_j)\setminus\{v_{l^*}\}}\{\xi(v_j, t_j)\}\}\\ &\le \min\{\xi(v_{l^*}, t_{l^*}), \min_{v_j \in T(E_j)\setminus\{v_{l^*}\}}\{\max_{t' \le t-d(E_k)} \xi(v_j, t')\}\}\\ &= \min\{\xi(v_{l^*}, t_{l^*}), \min_{v_j \in T(E_k)\setminus\{v_{l^*}\}} \xi^*_{v_j}\} = \xi_k(v_{l^*})\end{aligned} \tag{7}$$

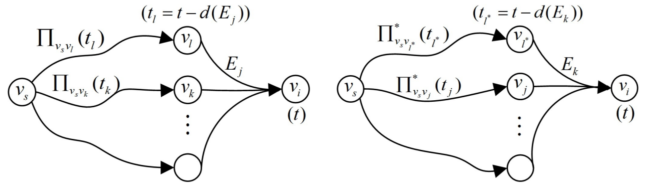
where $\xi^*_{v_j} = \max_{t' \le t_{l^*}} \xi(v_j, t')$. Thus, by the Definition 4 and Eq. (7), we have that

$$\begin{aligned}C(\Pi^*_{v_s v_i}) &= \min\{\min_{v_k \in T(E_k)} C(\Pi^*_{v_s v_k}(t_k), l(E_k, t_{l^*}, t))\}\\ &\le \min\{\xi_k(v_{l^*}), l(E_k, t - d(E_k), t)\}.\end{aligned}$$

Then, by the Eq. (3), $C(\Pi^*_{v_s v_i}) \le \min\{\xi_{T(E_k)}, l(E_k, t_{l^*}, t)\}$. Therefore, the choice of E_j implies that

$$\begin{aligned}C(\Pi^*_{v_s v_i}) &\le \max_{E_j \in BS(v_i,t)} \min\{\xi_{T(E_j)}, l(E_j, t-d(E_j), t)\}\\ &= \xi(v_i, t) = C(\Pi_{v_s v_i}).\end{aligned}$$

Then, $\Pi_{v_s v_i}$ is the maximum capacity of the TV-B-hyperpath from v_s to v_i at time exactly t and so the result holds. ∎



(a) $\Pi_{v_s v_i}$ (b) $\Pi^*_{v_s v_i}$

Fig. 5: Illustration of Theorem 1

According to Theorem 1, if we want to get $\xi(v_i, t)$, we need to calculate $\xi(v_k, t')$ for each $v_k \in T(E_j)$ and $t' \le F_j$, where $E_j \in BS(v_i)$. The computational complexity of this process is undoubtedly huge. Due to the restriction of D, some nodes are unreachable at some times. Therefore, we will remove these nodes in next subsection and can greatly reduce the computational complexity.

A. Hypergraph Pruning and Time Discretization.

1) Hypergraph Pruning: The primary goal of using Hypergraph Pruning is to remove as many nodes as possible that do not belong to any TV-B-hyperpath. In this subsection, we propose a bidirectional pruning algorithm (see Algorithm 1).

Recall that the sink v_t should be arrived in the time period $[t_0, t_0+D]$. Under this constraint, let f_v be the earliest arrival time at node v, g_v be the latest departure time at node v. $[f_v, g_v]$ is called the reachable time interval of v. We use forward search and backward search to obtain the f_v and g_v respectively. In the forward search, the calculation of f_v is mainly based on procedure SBT in [13]. In the backward search, we use the method similar to Dijkstra's Algorithm to calculate $\hat{g}_v$, which is a upper bound of g_v. This is because the Dijkstra's Algorithm can only be used to construct general paths and maybe not suitable for constructing B-hyperpaths. When f_v and $\hat{g}_v$ are obtained for each node v, $[f_v, \hat{g}_v]$ is an estimate of the reachable time interval and contains all the arrival and departure times of v. Noting that $\hat{g}_v \geq g_v$, we can check and delete some useless nodes in the main algorithm. Furthermore, in order to better achieve the purpose of pruning, we removes the following two categories of nodes:

- The node that cannot be reached from v_s before time $t_0 + D$;
- The node v that satisfies $\hat{g}_v < f_v$.

According to the above discussion, we propose Algorithm 1. The main steps of the algorithm are as follows.

Algorithm 1. First, the algorithm perform initialization in lines 1-9. Let $\overrightarrow{Q} = \{v_s\}$ and $\overleftarrow{Q} = \{v_t\}$. For each B-arc E_j, k_j is the counter which is used to provide the number of its tail nodes already removed from $\overrightarrow{Q}$. We initialize k_j to 0. Then, we perform the forward search in lines 10-22 in Algorithm 1, and it is based on Procedure SBT in [13]. The difference from Procedure SBT is that if v is not reachable from v_s within D, then $f_v - t_0 > D$ and we can safely remove v from $\mathscr{H}^1$. Lines 23-32 in Algorithm 1 are backward search. Since we consider the latest departure time g_v at each node v and we have got the $\hat{g}_v$ which is the upper bound of g_v, the algorithm first select and remove the node v with the largest $\hat{g}_v$ from the set $\overleftarrow{Q}$ (see line 24). if there exists a node v such that $\hat{g}_v < f_v$, that is, for the TV-B-hyperpath passing through the node v at time f_v, this hyperpath cannot transmit information to v_t before time $t_0 + D$, then we can thus remove v from $\mathscr{H}^1$. The above pruning process removes the second category of nodes (see lines 25-27). Otherwise, the algorithm checks and possibly updates $\hat{g}_y$ for every node $y \in T(E_j)\backslash\{v_s\}$ and $E_j \in BS(v)$. More precisely, a new time $\hat{g}_v - d(E_j)$ is compared to the currently time $\hat{g}_y$. If $\hat{g}_v - d(E_j)$ is larger, then the algorithm adds node y to the set $\overleftarrow{Q}$ and increases the value of $\hat{g}_y$ (see line 32). Finally, the algorithm terminates when $\overleftarrow{Q} = \phi$.

Correctness. According to the correctness of the Procedure SBT, f_v obtained by the forward search is indeed the earliest arrival time at node v. Moreover, since the node v such that $f_v - t_0 > D$ is not reachable within D, removing these nodes will not affect the calculation of f_v.

For backward search, we claim that $\hat{g}_v$ obtained by the Alogrithm 1 is the upper bound of the latest departure time g_v at v. By contradiction, suppose that there is a TV-B-hyperpath $\Pi_{v_i v_t}$ from v_i to v_t satisfying the following two conditions:

- the time to arrive at v_t is $t \in [t_0, t_0 + D]$;

Algorithm 1: Hypergraph pruning in $\mathscr{H}^1$

```
Input: $\mathscr{H}^1$, $t_0$, $D$, $v_s$, $v_t$
Output: $\mathscr{H}^1_{t_0}$ and $[f_{v_i}, \hat{g}_{v_i}]$ for each $v_i \in \mathscr{V}\backslash\{v_s\}$
1  Initialization: $\overrightarrow{Q} = \{v_s\}$; $\overleftarrow{Q} = \{v_t\}$; $k_j = 0$ for each $E_j \in \mathscr{E}$;
2  for each $v \in \mathscr{V}$ do
3      if $v = v_s$ then
4          $f_v = t_0$; $\hat{g}_v = -1$;
5      else
6          if $v = v_t$ then
7              $\hat{g}_v = t_0 + D$; $f_v = \infty$;
8          else
9              $f_v = \infty$; $\hat{g}_v = -1$;
10 while $\overrightarrow{Q} \neq \phi$ do
11     $v = argmin\{f_u : u \in \overrightarrow{Q}\}$; $\overrightarrow{Q} = \overrightarrow{Q}\backslash\{v\}$;
12     if $f_v - t_0 > D$ then
13         remove $v$ and the related B-arcs;
14     else
15         for each $E_j = (T(E_j), \{y\}) \in FS(v)$ do
16             $k_j = k_j + 1$;
17             if $k_j = |T(E_j)|$ then
18                 $t' = \max_{v_i \in T(E_j)} f_{v_i}$;
19                 if $t' + d(E_j) < f_y$ then
20                     if $y \notin \overrightarrow{Q}$ then
21                         $\overrightarrow{Q} = \overrightarrow{Q} \cup \{y\}$;
22                     $f_y = t' + d(E_j)$;
23 while $\overleftarrow{Q} \neq \phi$ do
24     $v = argmax\{\hat{g}_u : u \in \overleftarrow{Q}\}$; $\overleftarrow{Q} = \overleftarrow{Q}\backslash\{v\}$;
25     if $\hat{g}_v < f_v$ then
26         remove $v$ and the related B-arcs;
27     else
28         for each $E_j \in BS(v)$ do
29             for each $y \in T(E_j)$ such that $\hat{g}_v - d(E_j) > \hat{g}_y$ and $y \neq v_s$ do
30                 if $y \notin \overleftarrow{Q}$ then
31                     $\overleftarrow{Q} = \overleftarrow{Q} \cup \{y\}$;
32                 $\hat{g}_y = \hat{g}_v - d(E_j)$;
```

- there is a internal node u in $\Pi_{v_i v_t}$ such that the departure time at u is $g'_u > \hat{g}_u$ and for any other node v in the TV-B-subhyperpath $\Pi_{u v_t}(t)$, $g'_v \leq \hat{g}_v$.

Assume that E_j is the B-arc in $\Pi_{v_i v_t}$ such that $u \in T(E_j)$. Then $\hat{g}_{H(E_j)} - d(E_j) \geq g'_{H(E_j)} - d(E_j) \geq g'_u > \hat{g}_u$. Therefore, $\hat{g}_u < \hat{g}_{H(E_j)} - d(E_j)$, which contradicts line 32. Then the claim holds.

Complexity. The initialization requires $O(n + m)$ time. Since the complexity of forward search is the same as that of the Procedure SBT, lines 10-22 need $O(\max\{n^2, size(\mathscr{H}^1)\})$ time, so the complexity of forward search is $O(\max\{n^2, size(\mathscr{H}^1)\})$. For the backward search, we firstly need to choose a node v from $\overleftarrow{Q}$. Then we consider all of its neighbours. These steps take at most $O(n^2)$ time.

So, the overall time needed to compute $\hat{g}_v$ for each node v is $O(n^2)$. Therefore, the complexity of Alogrithm 1 is $O(m+n+\max\{n^2, size(\mathscr{H}^1)\}+n^2)$=$O(\max\{n^2, size(\mathscr{H}^1)\})$.

2) Time Discretization: By Algorithm 1, we obtain the time interval $[f_{v_i}, \hat{g}_{v_i}]$ for each node $v_i \in \mathscr{V}$. Then we propose the improvement of $\mathscr{T}$ as follows.

Observation 2. *For each $v_i \in \mathscr{V}$, if $v_i = v_s$, let $\mathscr{T}_{v_i} = \{t_0\}$; otherwise, let $\mathscr{T}_{v_i} = \{f_{v_i}, f_{v_i}+1, f_{v_i}+2, \ldots, \hat{g}_{v_i}\}$. Then $\mathscr{T}$ can be updated to $\bigcup_{v_i \in \mathscr{V}} \mathscr{T}_{v_i}$ which contains a finite number of discrete times.*

Note that $\hat{g}_{v_i}$ is the upper bound of g_{v_i}. There may be some time $t \in \mathscr{T}_{v_i}$ such that v_i can not be arrived at time t. However, by Observation 2, the updated $\mathscr{T}$ can also be used to reduce the time complexity significantly. Now we arrang all different times in $\mathscr{T}$ in ascending order and then $\mathscr{T}$ is written as $\{t_0, t_1, \ldots, t_n\}$, where $t_0 \le t_1 \le \cdots \le t_n$ and $n = |\mathscr{T}|$.

B. Algorithm scheme.

Armed with the pruned hypergraph $\mathscr{H}^1_{t_0}$ output by Algorithm 1 and the set $\mathscr{T}$ derived in last subsection, we develop Algorithm 2 in this subsection to get the maximum capacity of TV-B-hyperpath from v_s to v_t at time exactly t. The main steps of the algorithm are as follows.

Algorithm 2. We describe the algorithm steps according to the following three parts.

• Input and Output: The input of this algorithm contains the pruned hypergraph $\mathscr{H}^1_{t_0}$, start time t_0 and $\mathscr{T}$. Algorithm 2 outputs a capacity table M, such as TABLE I. The first column in this table represents each time $t \in \mathscr{T}$, and the first row represents each node $v_i \in \mathscr{V}$. Each entry $\xi(v,t)$ in this table represents the the maximum capacity of the TV-B-hyperpath from v_s to v at time exactly t.

• Initialization: For any $t \in \mathscr{T}$, the set of nodes that may be arrived at time t is denoted by $\mathscr{V}_t$. Let $\xi_{v_i} = 0$ for each node $v_i \in \mathscr{V}$ and $\xi_{T(E_j)} = 0$ for each B-arc $E_j \in \mathscr{E}$. Since v_s can only be reached at time t_0, let $\xi(v_s, t_0) = \infty$, $\mathscr{V}_{t_0} = \{v_s\}$ and $\xi(v_s, t) = NULL$ for each $t \in \mathscr{T} \backslash \{t_0\}$. For each $v_i \in \mathscr{V} \backslash \{v_s\}$, v_i may be reached at each time $t \in \mathscr{T}_{v_i}$. Therefore, the algorithm initializes $\xi(v_i, t)$ to 0 for each $t \in \mathscr{T}_{v_i}$. In addition, by using $\mathscr{T}_{v_i}$ of each v_i, we can obtain the set $\mathscr{V}_t$ for each $t \in \mathscr{T}$. In line 5, we set $\xi(v_i, t) = NULL$ for each $t \in \mathscr{T} \backslash \mathscr{T}_{v_i}$.

• Main Steps: Firstly, we introduce two important labels. For any $v_i \in \mathscr{V}$ and $t \in \mathscr{T}$, the maximum capacity TV-B-hyperpath to v_i at time exactly t is denoted by $\Pi_{v_s v_i}$. In this hyperpath, the B-arc whose head node is v_i is denoted by $Pe(v_i, t)$. Then, for example, suppose that $Pe(v_i, t) = E_j$, let $Pv(E_j, t) = \underset{v \in Q}{argmax}\{\xi_j(v)\}$, where Q is the set of nodes in $T(E_j)$ that can be reached at time $F_j = t - d(E_j)$.

Algorithm 2 performs the following steps for each time $t = t_1, t_2, \ldots, t_n$. For any $v_i \in \mathscr{V}_t$, the algorithm visits each B-arc E_j in $BS(v_i)$. The latest departure time at tail nodes of E_j is $F_j = t - d(E_j)$. Then we can get $Q = \{v_k : v_k \in T(E_j), F_j \in \mathscr{T}_{v_k}, \xi(v_k, F_j) \neq NULL\}$ (see lines 11-12). Next we calculate ξ_{v_k} according to Eq. (1) in Theorem 1 (line 13).

Here, we will have two cases that make it impossible to pass through the B-arc $E_j \in BS(v_i)$ to reach v_i.

Algorithm 2: Solving the TV-MCBH

```
   Input: $\mathscr{H}^1_{t_0}$ , $t_0$, $\mathscr{T}$
   Output:  A capacity table $M$
1  Initialization: $\xi_{T(E_j)} = 0$ for each $E_j \in \mathscr{E}$;
2  $\xi_{v_i} = 0$ for each $v_i \in \mathscr{V}$; $\xi(v_s, t_0) = \infty$; $\mathscr{V}_{t_0} = \{v_s\}$;
3  $\xi(v_s, t) = NULL$ for each $t \in \mathscr{T} \backslash \{t_0\}$;
4  $\xi(v_i, t) = 0$, $\mathscr{V}_t = \mathscr{V}_t \cup \{v_i\}$ for each $v_i \in \mathscr{V} \backslash \{v_s\}$,
   $t \in \mathscr{T}_{v_i}$;
5  $\xi(v_i, t) = NULL$ for each $v_i \in \mathscr{V} \backslash \{v_s\}$, $t \in \mathscr{T} \backslash \mathscr{T}_{v_i}$;
6  for each $t = t_1, \ldots, t_n$ do
7      for each $v_i \in \mathscr{V}_t$ do
8          for each $E_j \in BS(v_i)$ do
9              $Q = \phi$; $F_j = t - d(E_j)$
10             for each $v_k \in T(E_j)$ do
11                 if $F_j \in \mathscr{T}_{v_k}$ and $\xi(v_k, F_j) \neq NULL$
                   then
12                     $Q = Q \cup \{v_k\}$;
13                 $\xi_{v_k} = \max_{t' \le F_j} \{\xi(v_k, t')\}$;
14             if $Q = \phi$ then
15                 $\xi_{T(E_j)} = NULL$; continue;
16             else
17                 for each $v \in Q$ do
18                     if $T(E_j) \backslash \{v\} = \phi$ then
19                         $\xi_{T(E_j)} = \xi(v, F_j)$;
20                         $Pv(E_j, t) = v$;
21                     else
22                         for each $v_j \in T(E_j) \backslash \{v\}$ do
23                             if $\xi_{v_j} = 0$ then
24                                 $\xi_{T(E_j)} = NULL$;
25                                 break for loop in line 17;
26                         $\xi_j(v) =$
                           $\min\{\xi(v, F_j), \min_{v_j \in T(E_j) \backslash \{v\}} \xi_{v_j}\}$;
27                         if $\xi_j(v) > \xi_{T(E_j)}$ then
28                             $\xi_{T(E_j)} = \xi_j(v)$;
29                             $Pv(E_j, t) = v$;
30             $\xi'(v_i, t) =$
               $\min_{\xi_{T(E_j)} \neq NULL} \{\xi_{T(E_j)}, l(E_j, F_j, t)\}$;
31             if $\xi'(v_i, t) > \xi(v_i, t)$ then
32                 $\xi(v_i, t) = \xi'(v_i, t)$;
33                 $Pe(v_i, t) = E_j$;
34         if $\xi(v_i, t) = 0$ then
35             $\xi(v_i, t) = NULL$;
```

- *case 1*: There is no tail node can be reached at time F_j (see lines 14-15).
- *case 2*: Although there is at least one node $v \in T(E_j)$ that can be reached at time F_j, $T(E_j) \backslash \{v\}$ has a tail node v_j such that the TV-B-hyperpath to v_j at any time $t' \le F_j$ does not exist (see lines 23-25).

Based on the above two cases, we give the following steps to compute the maximum capacity $\xi(v_i, t)$.

When $Q = \phi$ (i.e. *case 1*), no node in $T(E_j)$ can be reached at time F_j. According to the Capacity Model, we can

TABLE I: The capacity table M in $\mathscr{H}^1$

$t\backslash v$	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
0	∞	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$
1	$NULL$	$NULL$	**3**	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$
2	$NULL$	**3**	**NULL**	**3**	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$
3	$NULL$	**2**	**3**	**NULL**	$NULL$	**2**	$NULL$	$NULL$	$NULL$	$NULL$
4	$NULL$	**NULL**	**NULL**	**NULL**	$NULL$	**NULL**	**1**	**3**	$NULL$	$NULL$
5	$NULL$	**2**	**NULL**	**NULL**	**2**	**2**	**2**	**2**	**2**	$NULL$
6	$NULL$	**NULL**	$NULL$	$NULL$	**1**	**NULL**	**NULL**	$NULL$	**1**	**1**
7	$NULL$	**NULL**	$NULL$	$NULL$	**2**	**NULL**	**1**	$NULL$	**2**	**2**
8	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	**1**
9	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	$NULL$	**2**

only wait until all tail nodes have received the information. So we cannot transmit information to v_i at time t passing through the B-arc E_j. Now $\xi_{T(E_j)} = NULL$ and we will go back to line 8 to continue considering the next B-arc in $BS(v_i)$.

When $Q \neq \phi$, for each $v \in Q$, if $T(E_j)\backslash\{v\} = \phi$, then $\xi_{T(E_j)} = \xi(v, F_j)$ and $Pv(E_j, t) = v$ (lines 19-20). Otherwise, if there is a node v_j in $T(E_j)\backslash\{v\}$ such that $\xi_{v_j} = 0$ (i.e. *case* 2), then there is no TV-B-hyperpath to v_j at each time $t' \leq F_j$. According to the Capacity Model, the information can be transmitted to the head node v_i only after all tail nodes have received the information. Therefore, it is impossible to arrive v_i at time t passing through the B-arc E_j. So we get $\xi_{T(E_j)} = NULL$ at time t and break out of the for loop in line 17.

In addition, if $Q \neq \phi$ and $\xi_{v_j} \neq 0$ for each tail node v_j in $T(E_j)\backslash\{v\}$, then we can reach v_i at time t passing through E_j. Thus by Theorem 1, we can calculate $\xi_{T(E_j)}$ and $\xi(v_i, t)$. At the same time, $Pv(E_j, t)$ and $Pe(v_i, t)$ can be obtained. (see lines 26-33).

By Observation 2, the set $\mathscr{T}_{v_i}$ of each v_i contains all arrival and departure times at v_i. But there may be a time $t \in \mathscr{T}_{v_i}$ such that there is no TV-B-hyperpath to v_i at time exactly t because of the using of $\hat{g}_{v_i}$ in $\mathscr{T}_{v_i}$. Then we cannot reach v_i at time t passing through each B-arc $E_j \in BS(v_i)$. In this case, after the for loop in line 8, $\xi(v_i, t)$ is still equal to the initial value 0, and then let $\xi(v_i, t) = NULL$ (see lines 34-35). Finally, Algorithm 2 will return the true value of $\xi(v_i, t)$.

Example 4. For the TV-BH $\mathscr{H}^1$ in Example 1, assume that $t_0 = 0$ and $D = 9$. Let v_1 be the source node. Then we will seek the maximum capacity TV-B-hyperpath from v_1 to v_{10} and the results obtained by Algorithms 1 and 2 are shown in TABLE I.

In this table, for each $v_i \in \mathscr{V}$ and $t \in \mathscr{T}$, if $t \in \mathscr{T}_{v_i}$, then the font of $\xi(v_i, t)$ is enlarged and bolded, which means that this value needs to be considered in Algorithm 2. Otherwise, the value is $NULL$ with the normal font. It can be seen that the hypergraph prunning in Algorithm 1 can effectively reduce the complexity of Algorithm 2. Moreover, since $\hat{g}_{v_i} \geq g_{v_i}$, there exist some $NULL$s in the table whose fonts are enlarged and bolded. Take the node v_2 as an example, the set $\mathscr{T}_{v_2} = \{2, 3, 4, 5, 6, 7\}$ and the node v_2 is reachable at times 2, 3, 5, but not at times 4, 6, 7. So we get $\xi(v_2, 4) = \xi(v_2, 6) = \xi(v_2, 7) = NULL$

Now, we will prove the correctness and complexity of Algorithm 2 in the following theorems.

Theorem 2. *For each $v_i \in \mathscr{V}$ and $t \in \mathscr{T}$, the $\xi(v_i, t)$ obtained by Algorithm 2 is the maximum capacity of the TV-B-hyperpath from v_s to v_i at time exactly t.*

Proof: For any $v_i \in \mathscr{V}$ and $t \in \mathscr{T}$, since the delay for each B-arc $E_j \in BS(v_i)$ is $d(E_j) > 0$, the arrival time to be considered for each tail node of E_j must be less than t when we compute $\xi(v_i, t)$. We iteratively consider time t in ascending order in the for loop in line 6, assume that the maximum capacity $\xi(v_k, t')$ for each $v_k \in T(E_j)$ and $t' < t$ has been calculated. Therefore, for some tail node v_k and some $t' < t$, if $\xi(v_k, t') = NULL$ or $\xi(v_k, t') = 0$, then there is no TV-B-hyperpath that can reach v_k at time t'.

According to Theorem 1, if there is a B-arc $E_j \in BS(v_i)$ such that $Q \neq \phi$ and $\xi_{v_k} \neq 0$ for each $v_k \in T(E_j)$, then the $\xi(v_i, t)$ obtained by lines 13, 26-29 and 30-33 in Algorithm 2 is the maximum capacity of the TV-B-hyperpath from v_s to v_i at time exactly t.

Now, it is enough to consider the case that for each $E_j \in BS(v_i)$, either $Q = \phi$ or $\xi_{v_k} = 0$ for some $v_k \in T(E_j)\backslash Q$.

If $Q = \phi$, then there is no tail node in E_j that can be reached at time $F_j = t - d(E_j)$. By lines 14-15, we get $\xi_{T(E_j)} = NULL$.

If $\xi_{v_k} = 0$ for some $v_k \in T(E_j)\backslash Q$, then there is at least one node in $T(E_j)\backslash Q$ that cannot be reached at time $t' \leq F_j$. By lines 23-25, $\xi_{T(E_j)} = NULL$.

Therefore, for each $E_j \in BS(v_i)$, $\xi_{T(E_j)} = NULL$ in above two cases. According to Capacity Model, v_i cannot be reached at time t. By lines 34-35, $\xi(v_i, t)$ is updated from initial value 0 to $NULL$, which completes the proof. ■

Theorem 3. *The time complexity of Algorithm 2 is $O(|\mathscr{T}|nm(|\mathscr{T}|l + l^2))$.*

Proof: Assume that $|T(E_j)| \leq l$ for each B-arc E_j. Since we need to initialize $\xi(v_i, t)$ for each $v_i \in \mathscr{V}$ and $t \in \mathscr{T}$ and initialize $\xi_{T(E_j)}$ for each $E_j \in \mathscr{E}$, our initialization can be done in $O(|\mathscr{T}|n + m)$ time.

For each $t \in \mathscr{T}$ and $v_i \in \mathscr{V}_t$, we need to visit each B-arc $E_j \in BS(v_i)$. Then Algorithm 2 firstly computes the ξ_{v_k} by considering each $v_k \in T(E_j)$ and $t \leq F_j = t - d(E_j)$ in line 13. This process requires at most $O(|\mathscr{T}|l)$ time. Next, the algorithm computes $\xi_{T(E_j)}$ by iterating over each $v \in Q$

and $v_j \in T(E_j)\backslash\{v\}$ in lines 17-29. These can be done in $O(l^2)$ time. Therefore, the complexity of the Algorithm 2 is $O(|\mathscr{T}|n+m+|\mathscr{T}|nm(|\mathscr{T}|l+l^2)) = O(|\mathscr{T}|nm(|\mathscr{T}|l+l^2))$. ■

Hyperpath Recovery. According to the capacity table M obtained by Algorithm 2, we can get that the maximum capacity of the TV-B-hyperpath satisfying the delay constraint from v_s to v_t is $\xi(v_t) = \max_{t\in\mathscr{T}_{v_t}}\{\xi(v_t,t) : \xi(v_t,t) \neq NULL\}$. If there are two times $t_1, t_2 \in \mathscr{T}_{v_t}$ such that $\xi(v_t,t_1) = \xi(v_t,t_2) = \xi(v_t)$, then, without loss of generality, suppose that $t_1 - t_0 < t_2 - t_0$. Let the optimal arrival time at v_t be $t^* = t_1$. That is, when the corresponding capacities of the t_1 and t_2 are the same, we choose to the time with the shorter delay as the optimal arrival time.

Note that Algorithm 2 outputs the $Pe(v_i,t)$ and $Pv(Pe(v_i,t),t)$ for each $t \in \mathscr{T}$ and $v_i \in \mathscr{V}_t$. By backtracking from the node v_t at time t^*, we can get the maximum capacity TV-B-hyperpath that reaches v_t at time t^*. The specific steps of hyperpath recovery are shown in the Appendix and an example is given as follows.

Example 5. By the capacity table M in TABLE I,

$$\begin{aligned}\xi(v_{10}) &= \max\{\xi(v_{10},6), \xi(v_{10},7), \xi(v_{10},8), \xi(v_{10},9)\}\\ &= 2.\end{aligned}$$

Since $t_0 = 0$, $\xi(v_{10},7) = \xi(v_{10},9) = 2$ and $7-t_0 = 7 < 9 = 9-t_0$, the optimal arrival time of v_{10} is $t^* = 7$. According to the $Pe(v_i,t)$ and $Pv(Pe(v_i,t),t)$ obtained by Algorithm 2 for each $t \in \mathscr{T}$ and $v_i \in \mathscr{V}_t$, we get the maximum capacity TV-B-hyperpath that reaches v_{10} at time 7. The corresponding BH of this TV-B-hyperpath is shown in Fig. 6.

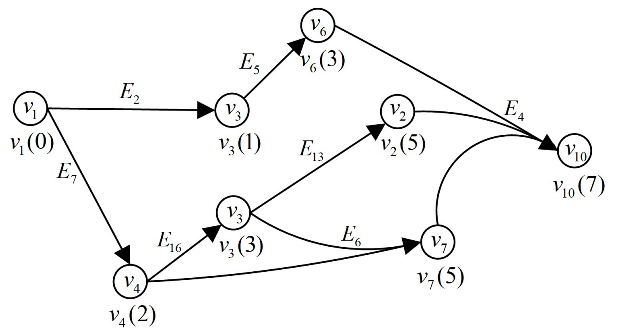


Fig. 6: The corresponding BH of TV-MCBH for Problem 1

IV. EXPERIMENTS

In this section, we test the algorithms mentioned above. For Problem 1, given a TV-BH $\mathscr{H}^1$ and $\mathscr{T} = \{t_0, t_0+1, t_0+2, \ldots, t_0+D\}$, we can get $\mathscr{H}^1_{t_0}$ and the improvement of $\mathscr{T}$ by Algorithm 1 and Observation 2. When the input of Algorithm 2 includes a TV-BH $\mathscr{H}^1$ and $\mathscr{T}$, the time to find the desired TV-B-hyperpath is denoted by MC1B. And when the input includes a TV-BH $\mathscr{H}^1_{t_0}$ and the updated $\mathscr{T}$, the time to find the desired TV-B-hyperpath is denoted by MC1-BH.

A. *Experiment setup.*

For the TV-MCBH, we randomly generate the ten classes of TV-BHs given in TABLE II with the number of nodes varying from 20 to 300 and the number of B-arcs varying from 60 to 900. In addition, the number of cycles in the TV-BH does not exceed 10% of the number of B-arcs. Assume that the number of tail nodes of each B-arc is not larger than $l = 3$. The third row of TABLE II refers to the average proportion of B-arcs with more than one tail node. The range of time-varying capacity function for each B-arc is $[50, 100]$. We set the delay of each B-arc between 10 and 20. Note that the average proportion of B-arcs with more than one tail node is larger than 55% and the number of B-arcs is 3 times the number of nodes. Therefore, this simulation can truly reflect the effect of the B-arc with more than one tail node on the complexity of the Algorithm 2.

B. *Experiment* 1 *(Hypergraph-Scalability).*

We set $D = 200$ for Problem 1. Then we consider following 3 parameters.

(1) The size of TV-BHs.

If the size of a TV-BH belonging to class $\mathscr{H}^1(i)$ is denoted by S_i for $i = 1, 2, \ldots, 10$, then with the size increasing from S_1 to S_{10}, the average processing time is shown in Fig. 7. Note that the average processing time is proportional to the value of size and MC1-BH clearly outperforms MC1B. It shows that using hypergraph pruning (i.e. Algorithm 1) can greatly reduce the average processing time of Algorithm 2.

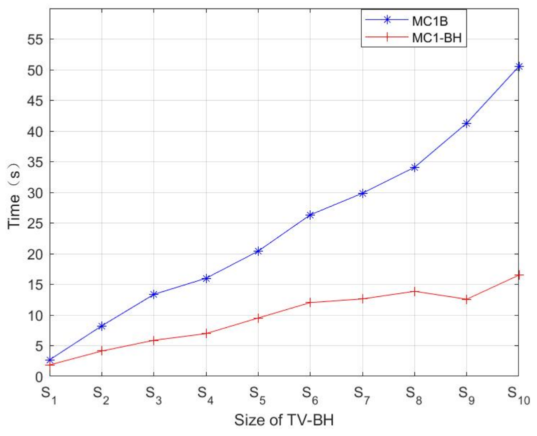


Fig. 7: Vary $size(\mathscr{H}^1)$ (Time).

(2) The number of B-arcs.

We vary the density of the TV-BH $\mathscr{H}^1$ by fixing the number of nodes as 100 and changing the number of B-arcs. 10 classes of TV-BHs are generated with 200, 300, 400, 450, 500, 550, 600, 700, 800 and 900 B-arcs, respectively. We report the average processing time in Fig. 8. The average processing time increases when the number of B-arcs increases. Note that the more B-arcs in TV-BH, the larger number of discrete times in the interval $[f_v, \hat{g}_v]$ of node v and so $\mathscr{T}$ is larger as well. This will affect the effect of the pruning algorithm. Therefore, when the number of B-arcs changes from 200 to 900, the average processing time MC1B and MC1-BH increase gradually. However, MC1-BH is also outperforms MC1B, obviously.

(3) The upper bound l of the number of tail nodes in each B-arc.

We will consider the TV-BHs belonging to the class $\mathscr{H}^1(4)$ which has been defined in TABLE II. Then, by

TABLE II: Datasets 1

	$\mathscr{H}^1(1)$	$\mathscr{H}^1(2)$	$\mathscr{H}^1(3)$	$\mathscr{H}^1(4)$	$\mathscr{H}^1(5)$	$\mathscr{H}^1(6)$	$\mathscr{H}^1(7)$	$\mathscr{H}^1(8)$	$\mathscr{H}^1(9)$	$\mathscr{H}^1(10)$
$\|\mathscr{V}\|$	20	50	80	100	120	150	180	200	250	300
$\|\mathscr{E}\|$	60	150	240	300	360	450	540	600	750	900
Average rate(%)	57.3	60.0	61.5	60.7	59.7	61.7	60.1	58.6	58.0	59.4

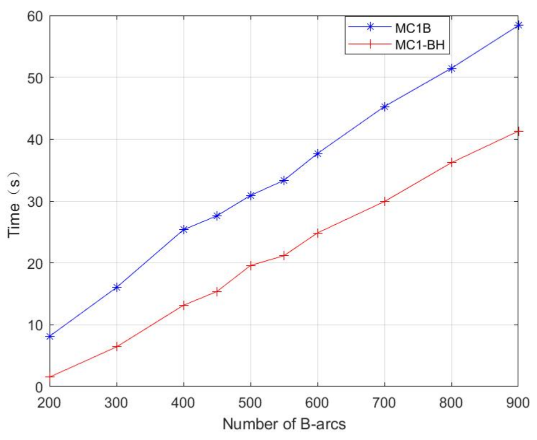


Fig. 8: Vary $|\mathscr{E}|$ in $\mathscr{H}^1$ (Time).

changing l, 8 classes of TV-BHs are generated with $l = 2$, $l = 3$, $l = 4$, $l = 5$, $l = 6$, $l = 7$, $l = 8$ and $l = 9$, respectively. In Fig. 9, MC1-BH outperforms MC1B and the average processing time increases when l increases. This is because that when l is larger, more tail nodes in each B-arc. Therefore, we can see that l is an important parameter that affects the running speed of Algorithm 2.

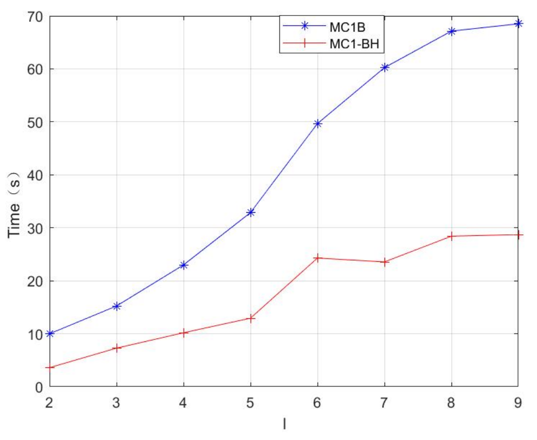


Fig. 9: Vary l in $\mathscr{H}^1$ (Time)

C. Experiment 2 (Query-Scalability).

We further test the effectiveness of algorithm by changing the parameter D of the Problem 1. Let $l = 3$ and $D > f_{v_t}$ which can ensure that there exists TV-B-hyperpath to v_t.

We change the parameter D from 100 to 600 for the TV-BHs belonging to $\mathscr{H}^1(4)$ and report the average processing time in Fig. 10. The number of times in $\mathscr{T}$ increases caused by the increased D. So the average processing time of MC1B and MC1-BH are proportional to D. As shown in Fig. 10, we can get that MC1-BH clearly outperforms MC1B. In addition, as D increases, the effect of the restriction of D gradually weakens, and so the average processing time of Algorithm 2 tends to be stable.

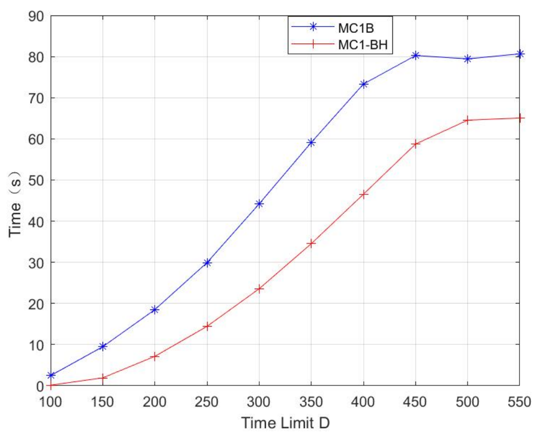


Fig. 10: Vary D in $\mathscr{H}^1$ (Time)

According to the above analysis, when the TV-BH is as sparse as possible and the fewer the number of tail nodes in each B-arc, the average processing time of Algorithm 2 is shorter. In addition, from the above Fig. 7-Fig. 10, we can know that using hypergraph pruning can greatly reduce the computation time.

V. Conclusion

In this paper, according to the application background, we designed the TV-BH $\mathscr{H}^1$ and studied the maximum capacity TV-B-hyperpath problem in $\mathscr{H}^1$ by using the model of information transmission. Then we proposed effective algorithm to compute the maximum capacity. Next, in order to further reduce the complexity of computation, we gave Algorithm 1 to prune hypergraphs. After that, our algorithm was improved significantly. Then, by the method of the hyperpath backtracking, we can get the maximum capacity TV-B-hyperpath $\Pi_{v_s v_t}$ with the optimal arrival time t^* and the delay of $\Pi_{v_s v_t}$ is not more than D. Finally, we conducted extensive experiments and confirmed that our algorithm can obtain the maximum capacity TV-B-hyperpath efficiently.

The results in this paper provide a theoretical basis for further research, especially the calculation of hyperpath capacity in time-varying hypergraph. In addition, for the hypernetwork with long delay time, we will research the multi-criteria time-varying scenarios, where both the capacity function and delay function are time-varying.

Appendix

In this section, we will give the processes of the hyperpath backtracking in $\mathscr{H}^1$.

Algorithm 3 returns the maximum capacity TV-B-hyperpath $\Pi_{v_s v_t}$ from v_s to v_t at time exactly t^* in $\mathscr{H}^1$. The main steps of the algorithm are as follows.

Input $\xi(v_t)$, the optimal arrival time t^* and the capacity table M obtained by Algorithm 2. Output the following sets: the node set $\mathscr{V}_{\Pi_t}$ of the required TV-B-hyperpath $\Pi_{v_s v_t}$, the set T_{v_i} of each $v_i \in \mathscr{V}_{\Pi_t}$, the set $BS_{\Pi_t}(v_i, t)$ consists of all B-arcs of $BS(v_i, t)$ in $\Pi_{v_s v_t}$.

In the while loop in line 2, a new set Q_2 and $T^1_{v_i}$ will be used to ensure that we can traverse all candidate nodes of Q_1 and the times in T_{v_i}. If $Q_1 \neq \phi$, then let $Q_2 = \phi$ and the algorithm performs the following steps for each $v_i \in Q_1$.

(1) If $v_i \notin \mathscr{V}_{\Pi_t}$, then we put it in $\mathscr{V}_{\Pi_t}$.

(2) For each $t \in T^1_{v_i}$, the algorithm removes t from $T^1_{v_i}$. If $Pe(v_i, t)$ exists, then let $BS_{\Pi_t}(v_i, t) = Pe(v_i, t)$ and $u = Pv(Pe(v_i, t), v)$. In the next step, we add all nodes in $T(Pe(v_i, t))$ to Q_2. Then, we get the arrival time t_u at u in line 13, and add t_u to T^1_u and T_u. After that, the algorithm calculates the arrival time t_{v_k} at each node v_k in $T(Pe(v_i, t))\backslash\{u\}$ and then puts them into both $T^1_{v_k}$ and T_{v_k} (see lines 16-18).

Finally, after the for loop in line 4, the nodes in Q_1 will be replaced by the new candidate nodes in Q_2. So the Q_1 is updated. Then the algorithm stops when $Q_1 = Q_2 = \phi$.

Algorithm 3: Hyperpath Recovery in $\mathscr{H}^1$

```
Input: The capacity table $M$, $\xi(v_t)$, $t^*$
Output: $\mathscr{V}_{\Pi_t}$, $T_{v_i}$ for each $v_i \in \mathscr{V}_{\Pi_t}$, $BS_{\Pi_t}(v_i, t)$ for
        each $v_i \in \mathscr{V}_{\Pi_t}$ and $t \in T_{v_i}$
1  Initialization: $Q_1 = \{v_t\}$, $\mathscr{V}_{\Pi_t} = \phi$, $T_{v_t} = \{t^*\}$,
   $T^1_{v_t} = \{t^*\}$
2  while $Q_1 \neq \phi$ do
3      $Q_2 = \phi$;
4      for each $v_i \in Q_1$ do
5          if $v_i \notin \mathscr{V}_{\Pi_t}$ then
6              $\mathscr{V}_{\Pi_t} = \mathscr{V}_{\Pi_t} \cup \{v_i\}$
7          for each $t \in T^1_{v_i}$ do
8              $T^1_{v_i} = T^1_{v_i} \backslash \{t\}$;
9              if $Pe(v_i, t)$ exists then
10                 $BS_{\Pi_t}(v_i, t) = Pe(v_i, t)$;
11                 $u = Pv(Pe(v_i, t), t)$;
12                 $Q_2 = Q_2 \cup T(Pe(v_i, t))$;
13                 $t_u = t - d(Pe(v_i, t))$; $T^1_u = T^1_u \cup \{t_u\}$;
14                 $T_u = T_u \cup \{t_u\}$;
15                 for each $v_k \in T(Pe(v_i, t))\backslash\{u\}$ do
16                     $t_{v_k} = \underset{t' \leq t_u}{argmax}\{\xi(v_k, t')\}$;
17                     $T^1_{v_k} = T^1_{v_k} \cup \{t_{v_k}\}$;
18                     $T_{v_k} = T_{v_k} \cup \{t_{v_k}\}$;
19     $Q_1 = Q_2$;
```

References

[1] G. Ausiello, G. F. Italiano, L. Laura, U. Nanni, F. Sarracco, "Structure theorems for optimum hyperpaths in directed hypergraphs," *Combinatorial Optimization*, Vol. 7422, pp. 1-14, 2012.

[2] G. Ausiello, G. F. Italiano, U. Nanni, "Hypergraph traversal revisited: cost measures and dynamic algorithms," *Mathematical Foundations of Computer Science*, Vol. 1450, pp. 1-16, 1998.

[3] G. Ausiello, L. Laura, "Directed hypergraphs: introduction and fundamental algorithms-A survey," *Theor. Comput. Sci.*, Vol. 658, No. PB, pp. 293-306, 2017.

[4] C. Berge, "Graphs and Hypergraphs," 1973.

[5] K. Bérczi, E. R. Bérczi-Kovács, "Directed hypergraphs and Horn minimization," *Information Processing Letters*, Vol. 128, No. C, pp. 32-37, 2017.

[6] O. Berman, G. Y. Handler, "Optimal minimax path of a single service unit on a network to nonservice destinations," *Transportation Science*, Vol. 21, No. 7, pp. 115-122, 1987.

[7] J. A. Bondy, U. S. R. Murty, "Graph Theory With Applications," pp. 171-185, 1976.

[8] A. Bretto, "Hypergraph theory: an introduction," 2013.

[9] R. Bisla, G. Singh, "Maximum capacity path problem in MANET'S," *Weekly Science*, Vol. 1, No. 1, 2013.

[10] Xi. Cai, D. Sha, C. K. Wong, "Time-varying network optimization," pp. 135-149, 2007.

[11] J. Che, X. Tong, L. Yu, "A dynamic bidirectional heuristic trust path search algorithm," *CAAI Transactions on Intelligence Technology*, Vol. 7, No. 3, pp. 340-353, 2022.

[12] H. N. Gabow, "Scaling algorithms for network problems," *J. Comput. Syst. Sci.*, Vol 31, No. 2, pp. 148-168, 1985.

[13] G. Gallo, G. Longo, S. Pallottino, S. Nguyen, "Directed hypergraphs and applications," *J. Discrete Applied Mathematics*, Vol. 42, No. 2, pp. 177-201, 1993.

[14] P. Hansen, "Bicriterion path problems," *Lecture notes in Economics and Mathematical Systems*, Vol. 177, pp. 109-127, 1980.

[15] D. Kirchler, L. Liberti, R. W. Calvo, "Efficient computation of shortest paths in time-dependent multi-modal networks," *LACM Journal of Experimental Algorithmics*, Vol. 19, No. 2, pp. 1-29, 2014.

[16] V. Kalbel, M. A.F. Peinhardt, "On the Bottleneck Shortest Path Problem," 2006.

[17] Y. Lu, G. Jossé, T. Emrich, U. Demiryurek, M. Renz, C. Shahabi, M. Schubert, "Scenic routes now: efficiently solving the time-dependent arc orienteering problem," *CIKM'17: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 487-496, 2017.

[18] G. Nannicini, D. Delling, D. Schultes, L. Liberti, "Bidirectional A* search on time-dependent road networks," *NETWORKS*, Vol. 56, No. 2, pp. 240-251, 2012.

[19] M. Pollack, "The maximum capacity through a network," *Operations Research*, Vol. 8, No. 5, pp. 733-736, 1960.

[20] A. P. Punnen, "A linear time algorithm for the maximum capacity path problem," *European Journal of Operational Research*, Vol. 53, No. 3, pp. 402-404, 1991.

[21] D. Pretolani, "A directed hypergraph model for random time dependent shortest paths," *European Journal of Operational Research*, Vol. 123, No. 2, pp. 315-324, 2000.

[22] G. H. Shirdel, H. Rezapour, "Time-varying maximum capacity path problem with zero waiting times and fuzzy capacities," *SpringerPlus*, Vol. 5, No. 1, pp. 981, 2016.

[23] R. H. Sloan, D. Stasi, G. Turán, "Hydras: directed hypergraphs and horn formulas," *Theoretical Computer Science*, Vol. 685, No. PB, pp. 417-428, 2017.

[24] G. H. Shirdel, B. Vaez-Zadeh, "Finding the shortest path for a Hypergraph," *Discret. Math. Algorithms Appl.*, Vol. 14, No. 3, pp. 2150120:1-2150120:17, 2021.

[25] J. Tayyebi, A. Deaconu, "Expanding maximum capacity path under weighted sum-type distances," *AIMS Mathematics*, Vol. 6, No. 4, pp. 3996-4010, 2021.