Throughput Maximization for Result Multicasting by Admitting Delay-aware Tasks in MEC Networks for High-speed Railways: Some Supplemental Materials

Road map: In section 1, we review the related work. In section 2, we list the key notations of the paper. In section 3, we give the analysis of handovers during the wireless transmission of each train. In section 4, we formulate the problem we defined as integer linear programming (ILP). In section 5, we list the details of `Procedure` 1. In section 6, we provide the analysis of the devised procedures and algorithms in terms of performance guarantee and time complexity. In section 7, we list a table of the communication parameters used in the simulations.

1 Related Work

As a key scenario of the 5G applications, edge-enabled networks for high-speed railways (HSRs) have recently attracted attention.

At first, edge caching and content delivery are investigated in the HSR scenarios. For example, In order to push popular services on high-speed trains (HSTs), Li et al. [1] presented a cache-based scheme with converged wireless broadcasting and cellular networks. They suggest caching and pushing the most popular services onto the train's vehicle relay station ahead of the trains' departure time. A dynamic programming approach is then proposed to maximize the network capacity in a constrained amount of pushing time. Xiong et al. [2] considered that there is little study on users' involvement in data traffic offloading. Thus they proposed a novel scheme to motivate user terminals in the HSRs scenarios to collaborate and cache wireless services to the routing relay (IRR) side. Two game-based auction strategies are respectively developed to maximize the social incomes and minimize the terminals' waiting time. However, the mentioned studies do not consider the delay and resource conditions caused by the data processing.

After that, several studies [3–8] explored the data offloading in the MEC environments for the HSRs. For example, Chen et al. [4] considered an edge computing-aided real-time fault detection framework for traction systems of the HSRs. Such a framework can be easily implemented without the knowledge of precise mathematical models, and there is no need to redesign the control structure. Liu et al. [5] introduced the edge computing model of fault detection and diagnosis for traction control systems for the HSRs, where they aim to minimize the execution cost of task offloading. Zhang et al. [6] examines the dynamic resource allocation and computation offloading with energy considerations in the HSR networks with the dynamic time division duplex. By jointly considering the various constraints, a non-convex optimization problem is constructed to reduce the amount of energy consumed. Then a bi-level optimization scheme with high computational complexity and a suboptimal approach with low computational complexity are proposed to solve the problem. Li et al. [8] took into account a mmWave-based train-ground communication system for the HSRs. Constrained by the local device and onboard mobile relays energy consumption, the problem of minimizing the average task processing latency for all users is formulated. Then a game-based joint scheme is proposed to spilt the data and allocate the sub-channel. In order to address users' task offloading and scheduling issues in high mobility scenarios, Li et al. [7] proposed a genetic algorithm-based scheme for the mobility-aware predictive computation offloading and task scheduling. Specifically, the computation offloading and task scheduling problem are formulated as a combinational optimization problem and a users' speed prediction module is put forward to assist the offloading decision.

All the aforementioned studies did not consider the applications of multicasting in the HSR scenario, such as video conferencing, multimedia pushing, multiplayer gaming, etc. Additionally, the HST is a specific type of vehicle, and there have been numerous studies on multicasting in the Internet of Vehicles (IoV) [9–15]. For example, Based on a road segmentation technique, Bousbaa et al. [9] proposed a distributed algorithm to address the challenges of multicast tree management among vehicles in urban environments. Roger et al. [10] proposed a low-latency multicast scheme to decrease the latency of Vehicle-to-Anything (V2X) communications for autonomous driving applications. By jointly considering coded multicast and edge caching, Bao et al. [11] investigated minimizing data traffic and the redundant problem in vehicular ad hoc networks (VANETs). Kadhim et al. [13] presented an energy-efficient multicast routing protocol by introducing software-defined networks and fog computing into vehicular networks. Hui et al. [14] presented a game-based cooperative content delivery system in which base stations collaborate with RSUs to service a group of vehicles utilizing multicast technology. Keshavamurthy et al. [12] analyzed cloud-based sidelink resource allocation problem for multicast group transmissions in the case of co-operative automated driving (CAD), and a graph-based solution framework is proposed to form clusters and assign inter-cluster resource block pool. Furthermore, they [15] also explored the multicast group-based vehicle-to-vehicle (V2V) communications for CAD scenarios by allocating the sidelink resource, constrained by reliability requirements and half-duplex limitation. The mentioned studies of multicasting in IoV can not be directly applied to the HSR scenarios, since there exists a safe distance [16] of tens of kilometers between neighboring trains, whereas the multicast grouping schemes for IoV are based on clustering the adjacent vehicles.

2 Key Notations

We list the key notations of our paper in Table 1.

3 Handover analysis

Similar to our previous study [17], four handover cases are discussed as follows.

Case 1: Train h finishes downloading the data of computation results before h leaves the coverage area of BS b, and $z(RE_h) < u_{max}$, i.e., $z(RE_h) \leq u_b(l(h), l(b)+\gamma)$. In this case, no handover occurs, and $\mathcal{N}_h^{\downarrow}$ equals 0.

Case 2: Train h cannot finish downloading the data of computation results before h leaves the coverage area of BS b, and $z(RE_h) \leq u_{max}$, i.e., $u_b(l(h), l(b)+\gamma) < z(RE_h) \leq u_{max}$. The rest data of computation results will be transmitted to the adjacent BS of BS b. In this case, the handover occurs only once. Thus $\mathcal{N}_h^{\downarrow}$ equals 1.

Case 3: Data volume of computation results $z(RE_h) > u_{max}$, so computation results can be split into $\lfloor z(RE_h)/u_{max} \rfloor$ portions. Different portions need to be transmitted to different BSs, and the number of the BSs thus is $\lfloor z(RE_h)/u_{max} \rfloor$. If train h can download volume $z(RE_h) - \lfloor z(RE_h)/u_{max} \rfloor \cdot u_{max}$ of data of computation results from BS b, before train h leaves the coverage area of BS b, i.e., $z(RE_h) - \lfloor z(RE_h)/u_{max} \rfloor \cdot u_{max} \leq u_b\left(l(h), l(b)+\gamma\right)$. Then $\mathcal{N}_h^{\downarrow}$ equals $\lfloor z(RE_h)/u_{max} \rfloor$.

Case 4: Data volume of computation results $z(RE_h) > u_{max}$, and train h cannot download $z(RE_h) - \lfloor z(RE_h)/u_{max} \rfloor \cdot u_{max}$ of data of computation results from BS b, before train h leaves the coverage area of BS b, i.e., $z(RE_h) - \lfloor z(RE_h)/u_{max} \rfloor \cdot u_{max} > u_b\left(l(h), l(b)+\gamma\right)$. Then $\mathcal{N}_h^{\downarrow}$ equals $\lfloor z(RE_h)/u_{max} \rfloor$.

In summary, $\mathcal{N}_h^{\downarrow}$ can be expressed by

$$\mathcal{N}_h^{\downarrow} = \begin{cases} \lfloor z(RE_h)/u_{max} \rfloor & \text{Case 1 or Case 3,} \\ \lceil z(RE_h)/u_{max} \rceil & \text{Case 2 or Case 4.} \end{cases} \tag{1}$$

4 An ILP Formulation

For the defined problem, we start with the classic ILP formulation. For easy of description, we transform undirected graph $G = (V, E)$ into directed graph $G_d = (V_d, E_d)$. Specifically, For each node $v \in V$, add node v into V_d, for each link $e \in E$, directed edge $\langle u, v \rangle$ and directed edge $\langle v, u \rangle$ are added into E_d, where $u \in V$ and $v \in V$. The weights of edge $\langle u, v \rangle$ and edge $\langle v, u \rangle$ are equal to the weight of link $e \in E$, i.e., $V_d = \{v | \ v \in V\}$, $E_d = \{\langle u, v \rangle \mid u, v \in V\}$. For brevity, we define a sign function $sgn(x)$ as follows

$$\text{sgn}(x) = \begin{cases} 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases} \tag{2}$$

The ILP includes the following decision variables.

$\mathbf{n}_{i,h}^v$ is a decision variable of value 1 if cloudlet task $k_{i,h}$ is computed in cloudlet $v_c \in V_d$ and value 0 otherwise. $\mathbf{n}^v$ is a decision variable of value 1 if cloudlet c_v is the computing cloudlet and value 0 otherwise.

$\mathbf{x}_{i,h}$ is 1 if task $k_{i,h}$ is admitted or 0 otherwise.

$\mathbf{y}_{i,h,h'}$ is 1 if result $r_{i,h,h'}$ is multicasted or 0 otherwise.

$\mathbf{m}_v$ is a decision variable that has value 1 if node v carries the traffic of any task or result; Otherwise, the value is 0. $\mathbf{m}_{\langle u,v \rangle}$ is a decision variable that has value 1 is edge $\langle u, v \rangle$ carries the traffic of any task or result; Otherwise, the value is 0.

According to the problem we defined, the objective function of the ILP is:

$$maximize \quad \sum_{h=1}^{|H|} \sum_{i=1}^{|K_h|} \sum_{h' \in DH_{i,h}} \mathbf{y}_{i,h,h'} \tag{3}$$

Constraints (4), (5) and (6) ensure that (i) each task is computed in the same cloudlet; (ii) there exists only one computing cloudlet in G for computing task set $K_H(t)$; (iii) node $v \in V$ can compute a task if and only if it is a cloudlet,

$$\sum_{v \in V_c} \mathbf{n}_{i,h}^v = \mathbf{x}_{i,h}, \ \ i \in K_h, \ \ h \in H \tag{4}$$

$$\sum_{v \in V_c} \mathbf{n}^v = \text{sgn}\left(\sum_{v \in V_c} \mathbf{n}_{i,h}^v \right), \ \ i \in K_h, \ \ h \in H \tag{5}$$

Table 1: key notations

Notations	Definition	Notations	Definition
$B \subset V$	the set of uniformly deployed homogeneous BSs with routing capabilities	B_H	the BS groups serving train set H
$B_h \in B_H$	the BS group serving train h	$B_h^{\downarrow}$	the group of destination BSs for transmitting result set RE_h to train h
C_H	the admission cost of task set $K_H(t)$	$C_{i,h}$	the admission cost of task $k_{i,h}$
C_v	the computing capacity of cloudlet $v_c \in V_c$	$\mathcal{C}_b^{\downarrow}(x)$	the channel capacity of the downlink of BS b when the train location is x
$c(e)$	the cost of consuming one unit of bandwidth at link $e \in E$	$c(v)$	the cost of consuming one unit of storage capacity at router $v \in V$
$c(v_K)$	the cost of using computing resources in computing cloudlet v_K	DH_h	the set of the destination trains of task set K_h
$DH_{i,h} \subseteq DH_h$	the set of the destination trains to which the computation results need to be sent	$D_h \subseteq B$	the set of destination BSs to which result set R_h is delivered
d_e	the delay on link $e \in E$ for transmitting a unit of data traffic	d_v	the delay on router $v \in V$ for transmitting a unit of data traffic
d_{req}	the identical end-to-end delay requirement of each task	$d_{i,h}^{com}$	the delay that task $k_{i,h}$ computed in computing cloudlet v_K
$d_{i,h}^{rou}$	the routing delay of task $k_{i,h}$ in T_K	$d_{i,h}^{net}$	the network delay of task $k_{i,h}$ and its results
d_H^{net}	the network delay of task set $K_H(t)$	d_H^{total}	the total delay experienced by multicast task set $K_H(t)$ in G
δ	the distance between each BS and the tracks	F_v	the number of CPU cycles per second of each container of cloudlet $v_c \in V_c$
$f_{i,h}$	the demanded CPU cycles for computing task $k_{i,h}$	$G = (V, E)$	the MEC network with a set V of routers and a set E of wired links
$\mathcal{G}_b(x)$	the channel gain between BS b and a train when the train location is x	H	the set of trains in the system, each train is denoted by $h \in H$
$K_H(t)$	the task set from train set H at moment t	$K_h \subseteq K_H(t)$	the task set from train h at moment t
$K_H^v(t) \subseteq K_H(t)$	the set of tasks routing on tree T_v	$k_{i,h}$	the ith task from train h, where $k_{i,h} \in K_h$
$l(\cdot)$	the function to obtain the location of a train / BS	$\mathcal{N}_h^{\downarrow}$	the number of handovers for train h in downlinks
p_h	the simple path connects a certain BS $b \in B_h$ with $v_K \in T_K$	$p_{h,h'} = p_h \cup p_{h'}$	the path passing through root vertex v_T and connecting group B_h and group $B_{h'}$ in T_v
R_H	the set of computation results of $K_H(t)$	$R_h \subseteq R_H$	the computation result set associated with task set K_h
$RE_h \subseteq R_H$	the set of computation results received by train h	$R_H^v \subseteq R_H$	the set of results routing on tree T_v
$r_{i,h,h'} \in R_h$	a copy of result received by each destination train $h' \in DH_{i,h}$	γ	each BS's wireless coverage radius along the track
T_K	the optimal multicast tree which connects computing cloudletv_K with each $B_h \in B_H$	T_v	a general group Steiner tree rooted at any cloudlet $v_T \in V_c$
$u_b(l(\cdot), l(*))$	the data volume transmitted by BS b when train moves from location "$\cdot$" to "$*$"	$\mathcal{U}(b_h^j)$	the data volume that each BS $b_h^j \in B_h^{\downarrow}$ transmits to train h
u_{max}	the maximum volume of data that a train receives from a BS	$V_c \subset V$	the routers with attached cloudlets
v_K	the computing cloudlet for processing task set $K_H(t)$	ν_h	the velocity of train h
$z(\cdot)$	the function to obtain the data volume of a task or result or task set or result set	$\rho_{i,h} \in \mathbb{R}^+$	the ratio between the volumes of task $k_{i,h}$ and result $r_{i,h,h'}$

$$\mathbf{n}^v_{i,h} = \mathbf{n}^v = 0, \ \ \forall v \in V_d \setminus V_c \tag{6}$$

Constraint (7) enforces the capacity constraint for each cloudlet $v_c \in V$.

$$\sum_{h=1}^{|H|}\sum_{i=1}^{|K_h|} \mathbf{x}_{i,h} \le C_v, \ \ \forall v \in V_d, \ \mathbf{n}^v = 1 \tag{7}$$

Constraint (8) is standard linear programming of a GST [18], and all the data traffic of the tasks and results will be routed on the obtained GST.

$$\sum_{\langle u,v\rangle \in \delta(V')} \mathbf{m}_{\langle u,v\rangle} \ge 1, \ \ \forall V' \subseteq V, \text{ such that } v_K \in V' \text{ and } Q \cap B_h = \emptyset \text{ for some } h \in H \tag{8}$$

Constraint (9) enforces that the total cost for routing the tasks and the results on the obtained GST will not exceed the budget.

$$\sum_{h=1}^{|H|}\sum_{i=1}^{|K_h|}\left(z(k_{i,h})\mathbf{x}_{i,h} \cdot \left(\sum_{\langle u,v\rangle \in E} c\left(\langle u,v\rangle\right) \cdot \mathbf{m}_{u,v} + \sum_{v\in V} c(v)\cdot \mathbf{m}_v \right) + \sum_{h' \in DH_{i,h}} z(r_{i,h,h'})\mathbf{y}_{i,h,h'} \cdot \sum_{\langle u,v\rangle \in E} c\left(\langle u,v\rangle\right)\cdot \mathbf{m}_{u,v} \right) \le \beta, \ \ \forall u,v \in V \tag{9}$$

Constraint (10) enforces that the total delay experienced by an admitted task and its corresponding results on the obtained GST can meet the delay requirement of the task.

$$\left(\sum_{v\neq u} \mathbf{m}_{\langle u,v\rangle} \cdot d_e + \sum_{v\in V} \mathbf{m}_v \cdot d_e \right) \cdot \mathbf{x}_{i,h} + \max_{h' \in DH_{i,h}} \left\{ \left(\sum_{v\neq u} \mathbf{m}_{\langle u,v\rangle} \cdot d_e + \sum_{v\in V} \mathbf{m}_v \cdot d_e \right) \cdot \mathbf{y}_{i,h,h'} \right\} + \sum_{v \in V_c} \mathbf{n}^v \cdot d^{com}_{i,h} \le d_{req}, \ \ i \in K_h, \ h \in H \tag{10}$$

Inspired by the network flow model [19] for the multicasting routing in the directed Steiner tree, we propose our flow model (Constraints (11)-(19)) to capture the traffic changes in the directed GST. We call that a node *consumes* a task/result if the node takes out one task/result from the passing data flow. Clearly, such a node can be computing cloudlet or a BS in the BS group, while the other nodes do not consume any task or result. Specifically, let $R_{u,v}$ and $S_{u,v}$ be the aggregate number of results and tasks going from vertex u to v, respectively. Constraint (11) restricts the range of $R_{u,v}$ and $S_{u,v}$

$$S_{u,v}, R_{u,v} \ge 0, \ \ \forall u,v \in V_d \tag{11}$$

Constraint (12) enforces that the number of the results routing on the path of the obtained GST is less than the number of the results multicasted by the root.

$$\sum_{u\neq v} \mathbf{m}_{\langle v,u\rangle} \cdot R_{v,u} \le \sum_{h=1}^{|H|}\sum_{i=1}^{|R_h|}\sum_{h' \in DH_{i,h}} \mathbf{y}_{i,h,h'}, \ \ \forall u,v \in V_d, \ \mathbf{n}^v = 1 \tag{12}$$

Constraint (13) enforces that no result can return to the root.

$$\sum_{u \in V, u\neq v} R_{u,v} = 0, \ \ \mathbf{n}^v = 1 \tag{13}$$

Constraints (14) and (15) handle four cases of the results consumed by the groups, where node v will not consume any result if (i) $v \notin B_H$ or (ii) v is in a certain BS group and v connects other nodes of the same group; otherwise, node v will consume $\sum_{h=1}^{|H|}\sum_{i=1}^{|R_h|} \mathbf{y}_{i,h,h'}$ amount of results if (iii) v is in a certain BS group and v connects a node $u \notin B_H$ or (iv) v is in a certain BS group and v connects a node $u \in B_H$ belonging to another BS group.

$$\sum_{v\neq u} \mathbf{m}_{\langle u,v\rangle} \cdot R_{u,v} - \sum_{v \neq w} \mathbf{m}_{\langle v,w\rangle} \cdot R_{v,w} = 0, \ \ \begin{cases} \text{Case 1:} \forall \mathbf{n}^v, \mathbf{n}^u, \mathbf{n}^w \neq 1, \ \forall v \notin B_H \\ \text{Case 2:} \forall \mathbf{n}^v, \mathbf{n}^u, \mathbf{n}^w \neq 1, \ v \in B_{h'}, \ \forall u,w \in B_{h'}, \ h' \in DH_h \end{cases} \tag{14}$$

$$\sum_{v\neq u} \mathbf{m}_{\langle u,v\rangle} \cdot R_{u,v} - \sum_{v \neq w} \mathbf{m}_{\langle v,w\rangle} \cdot R_{v,w} = \sum_{h=1}^{|H|}\sum_{i=1}^{|R_h|} \mathbf{y}_{i,h,h'}, \ \ \begin{cases} \text{Case 3:} \forall \mathbf{n}^v, \mathbf{n}^u, \mathbf{n}^w \neq 1, v \in B_{h'}, \ h' \in DH_h, \ \exists u \notin B_H \\ \text{Case 4:} \forall \mathbf{n}^v, \mathbf{n}^u, \mathbf{n}^w \neq 1, v \in B_{h'}, \ \forall u \in B_{h''}, \ h', h'' \in DH_h \end{cases} \tag{15}$$

Constraint (16) ensures that no unprocessed task can leave the root of the GST.

$$\sum_{u\neq v} S_{v,u} = 0, \ \ \forall u,v \in V_d, \ \mathbf{n}^v = 1 \tag{16}$$

Constraint (17) and constraint (18) ensure that any node will not consume the task except the node is the root.

$$\sum_{v \neq u} \mathbf{m}_{\langle u,v \rangle} \cdot S_{u,v} - \sum_{v \neq w} \mathbf{m}_{\langle v,w \rangle} \cdot S_{v,w} = 0, \ \forall u, v, w \in V_d, \ \forall \mathbf{n}^v, \ \mathbf{n}^u, \mathbf{n}^w \neq 1 \tag{17}$$

$$\sum_{v \neq u} \mathbf{m}_{\langle u,v \rangle} \cdot S_{u,v} - \sum_{v \neq w} \mathbf{m}_{\langle v,w \rangle} \cdot S_{v,w} = \sum_{h=1}^{|H|} \sum_{i=1}^{|K_h|} \mathbf{x}_{i,h}, \ \forall u, v, w \in V_d, \ \mathbf{n}^v = 1, \ \forall \mathbf{n}^u, \mathbf{n}^w \neq 1 \tag{18}$$

Constraint (19) ensures that only when an edge between vertex u and vertex v is included in the GST is it possible that $S_{u,v} > 0$.

$$\left(\sum_{h=1}^{|H|} \sum_{i=1}^{|K_h|} \mathbf{x}_{i,h} \right) \cdot \sum_{\langle u,v \rangle \in E} \mathbf{m}_{\langle u,v \rangle} \geq S_{u,v}, \ \forall u, v \in V_d \tag{19}$$

Constraints (20), (21), (22), (23), (24) restrict the ranges of decision variables to 0 and 1.

$$\mathbf{n}^v_{i,h}, \mathbf{n}^v \in \{0, 1\}, \ \forall v \in V_c, \ \forall i \in K_h, \ \forall h \in H \tag{20}$$

$$\mathbf{x}_{i,h} \in \{0, 1\}, \ \forall i \in K_h, \ \forall h \in H \tag{21}$$

$$\mathbf{y}_{i,h,h'} \in \{0, 1\}, \ \forall i \in K_h, \ \forall h \in H, \ h' \in DH_{i,h} \tag{22}$$

$$\mathbf{m}_{\langle u,v \rangle} \in \{0, 1\}, \ \forall u, v \in V_d \tag{23}$$

$$\mathbf{m}_v \in \{0, 1\}, \ \forall v \in V_d \tag{24}$$

Constraints (25) and (26) describe the priority relationships between the tasks and their results, i.e., the number of the results is greater than those of the corresponding tasks, and a result can be multicasted if and only if its corresponding task has been admitted.

$$\mathbf{x}_{i,h} \geq \mathbf{y}_{i,h,h'}, \ \forall i \in K_h, \ \forall h, h' \in H \tag{25}$$

$$\mathbf{x}_{i,h} \leq \sum_{h' \in DH_h} \mathbf{y}_{i,h,h'} \leq \mathbf{x}_{i,h} \cdot |R_{i,h}|, \ \forall i \in K_h, \ \forall h, h' \in H \tag{26}$$

5 The Details of Procedure 1

We list the details of `Procedure` 1 as follows.

Procedure 1: Adjusting the Group Steiner Tree with Delay Constraints

Input: A network $G=(V,E)$, group Steiner tree T_v rooted at v_T in G, auxiliary graph $G_v=(V_v,E_v)$ with respect to G and v_T, delay constraint $\widetilde{d_H^{net}}$, BS group $B_H=\{B_1,...,B_h,...,B_{|H|}\}$, Task set $K_H(t)$, Result set R_H.

Output: (i) Adjusted tree T_v, such that $d(T_v)\le\widetilde{d_H^{net}}$; (ii) Task set $K_H^v(t)$ and result set R_H^v

$Q_{delay}, p_{h,h'}^{dclc} \leftarrow \emptyset$; /*$Q_{delay}$ is a max-heap*/
loop $\leftarrow 1$;
$K_H^v(t)\leftarrow K_H(t), R_H^v\leftarrow R_H$;
foreach $h\in H$ **do**
  Find path p_h in T_v, and $h'\leftarrow h$;
  while $h'\le|H|$ **do**
    $h'\leftarrow h'+1$, find path $p_{h'}$ in T_v and construct path $p_{h,h'}\leftarrow p_h\cup p_{h'}$, $P_H\leftarrow P_H\cup\{p_{h,h'}\}$, $Insert\ (Q_{delay},p_{h,h'})$; /*Construct set P_H and sort the paths in P_H in decreasing order of their delay*/
    Find task set K_p and calculate routing delay $d_{i,h}^p$ for each task $k_{i,h}\in K_p$, sort task set K_p in increasing order by the delay of each task;
  end
end
while *loop* $\le|P_H|$ **do**
  $p_{h,h'}\leftarrow ExtractMax\ (Q_{delay})$;
  if $d(p_{h,h'})\le\widetilde{d_H^{net}}$ **OR** *loop* $>|P_H|$ **then**
    The adjustment is finished, **return** $T_v, K_H^v(t), R_H^v$;
  else if $d(p_{h,h'})>\widetilde{d_H^{net}}$ **then**
    Find the least cost path $p_{h,h'}^{dclc}$ between dummy vertices x'_h and $x''_{h'}$ with delay constraint $\widetilde{d_H^{net}}$ by using Juttner's algorithm [20] in G_v;
    if $p_{h,h'}^{dclc}$ *exists* **then**
      Path $p_{h,h'}^{adj}=p_h^{adj}\cup p_{h'}^{adj}$ in G, which connects group B_h and $B_{h'}$, is derived from $p_{h,h'}^{dclc}$ in G_v;
      Examine the feasibility of p_h^{adj} and $p_{h'}^{adj}$ by respectively invoking `Procedure` 2;
    end
    if p_h^{adj} *and* $p_{h'}^{adj}$ *are both feasible* **then**
      Replace path $p_{h,h'}$ in T_v with $p_{h,h'}^{adj}$, $J_h\leftarrow J_h\cup\{p_{h'}^{adj}\}$, $J_{h'}\leftarrow J_{h'}\cup\{p_h^{adj}\}$, *loop* $\leftarrow$ *loop* $+1$;
    end
    if *(i)* $p_{h,h'}^{dclc}$ *does not exist* **OR** *(ii)* p_h^{adj} *or* $p_{h'}^{adj}$ *is not feasible* **then**
      For each task $k_{i,h}\in K_p$, whose delay $d_{i,h}^p$ is greater than $\widetilde{d_H^{net}}$, REJECT result $r_{i,h,h'}$ of this task, which corresponds to $d_{i,h}^p$, $R_H^v\leftarrow R_H^v\backslash\{r_{i,h,h'}\}$. If the rejected result $r_{i,h,h'}$ is the only result of task $k_{i,h}$, then REJECT task $k_{i,h}$, $K_H^v(t)\leftarrow K_H^v(t)\backslash\{k_{i,h}\}$. *loop* $\leftarrow$ *loop* $+1$;
    end
  end
  if *loop* $\le|P_H|$ **then**
    If $p_h\neq p_h^{adj}$, then remove p_h from Q_{delay}, $Insert\ (Q_{delay},p_h^{adj})$;
    If $p_{h'}\neq p_{h'}^{adj}$, then remove p'_h from Q_{delay}, $Insert\ (Q_{delay},p_{h'}^{adj})$;
  end
end

6 Algorithm Analysis

Corollary 1. *If a GST can be successfully constructed or adjusted, then the delay requirements of the tasks routing on this GST can be met.*

Theorem 1. `Procedure` 3 *has a relative performance guarantee of* $1/2$.

Proof. First, let $q_{\max}$ be a feasible solution of the ILP, where we admit the task with the most results and the results are also multicasted, i.e.,

$$q_{\max}=\max_{i,h}\sum_{h'\in DH_{i,h}}\mathbf{y}_{i,h,h'}.$$

Then, denote p^{LP} as the optimal solution of linear relaxation of **P1**, where p^{LP} is an upper bound of the optimal solution. Furthermore, denote by p^G the objective function value obtained by `Procedure` 3. p^G takes the best solution among p^{LP}

and $q_{\max}$, and p^G can be expressed as follows

$$p^G = \max\left\{p^{LP}, q_{\max}\right\}.$$

Considering the optimal solution of **P1**, which is denoted as p^*, it can be seen that the following inequality for p^* holds:

$$p^* < p^{LP} + q_{\max}.$$

Let $\rho = \frac{p^G}{p^*}$ be the approximation ratio and we can get with

$$\rho = \frac{p^G}{p^*} = \frac{\max\left\{p^{LP}, q_{\max}\right\}}{p^*} > \frac{\max\left\{p^{LP}, q_{\max}\right\}}{p^{LP} + q_{\max}}.$$

Clearly, $p^{LP} + q_{\max} \leq 2 \cdot \max\left\{p^{LP}, q_{\max}\right\}$, thus $\rho > \frac{\max\left\{p^{LP}, q_{\max}\right\}}{2 \cdot \max\left\{p^{LP}, q_{\max}\right\}} = \frac{1}{2}$.

$\square$

Lemma 1. *Given a network $G = (V, E)$, a group Steiner tree T_v rooted at v_T in G, an auxiliary graph $G_v = (V_v, E_v)$ with respect to G and v_T, delay constraint $\widetilde{d_H^{net}}$, BS group $B_H = \{B_1, ..., B_h, ..., B_{|H|}\}$, Task set $K_H(t)$, and Result set R_H, there is a procedure* `Procedure 1`*, which takes $O\left(|H|^2 \cdot \left(|E|^2 log^4|E| + |K_H(t)||H|(|V| + |E|)\right)\right)$ time in the worst case for (i) adjusting tree T_v, such that $d(T_v) \leq \widetilde{d_H^{net}}$, (ii) obtaining task set $K_H^v(t)$ and result set R_H^v that routing on tree T_v.*

Proof. First, it does the initialization in $O(1)$ time (Lines 1-3).

Then it processes every $h \in H$ as follows (Lines 4-10). It conducts a for loop with $O(|H|)$ iterations (Line 4). In each iteration, it finds path p_h in $O(|V| + |E|)$ time (Line 5). After that, it constructs path set P_H using a while loop with $|H|$ iterations (Lines 6-9) as follows. It finds path $p_{h'}$ in $O(|V| + |E|)$ time and constructs path $p_{h,h'}$ and merges $p_{h,h'}$ into set P_H in $O(1)$ time. It tasks $O\left(log\left(|H|^2 + |H|\right)\right)$ time to insert $p_{h,h'}$ into max-heap Q_{delay}, the reason is that the number of elements in Q_{delay} equals the value of $|P_H|$, i.e., $\frac{(|H|+1)\cdot|H|}{2}$. Thus it tasks $O\left(|V| + |E| + log\left(|H|^2 + |H|\right)\right)$ time to do the operations of Line 7. Then, in Line 8, it finds task set K_p in $O(|K_H(t)|)$ time. The calculation of routing delay $d_{i,h}^p$ for each task $k_{i,h} \in K_p$ takes $O((|V| + |E|))$ time, thus the total calculations of routing delays for task set K_p take $O\left((|V| + |E|) \cdot |K_H(t)|\right)$ time. After that, the sorting operations tasks $O(|K_H(t)| \cdot log|K_H(t)|)$ time. Thus, it tasks $O\left((|V| + |E| + log|K_H(t)|) \cdot |K_H(t)|\right)$ to do the operations of Line 8.

Next, it processes the paths in P_H with a while loop with $|P_H|$ iterations (Line 11) as follows (Lines 12-31). In each iteration, It pops out path $p_{h,h'}$ from max-heap Q (Line 12) in $O\left(log\left(|H|^2 + |H|\right)\right)$ time, where $p_{h,h'}$ has the maximum delay among all the paths in P_H. Specially, it tasks $O(K_H(t))$ time to do the calculations of $d(p_{h,h'})$ since we have calculated all the delays of tasks that pass through path $p_{h,h'}$ in Line 8. Thus, the operations of Lines 13-14 take $O(K_H(t))$ time. In the **best** case, `Procedure 1` will stop at Line 14. Otherwise, it will do the operations of Lines 15-27. It tasks $O(K_H(t))$ time to do the operations of Line 15. Then, it takes $O(|E|^2 \cdot log^4|E|)$ to find delay-constrained-least-cost (DCLC) path $p_{h,h'}^{dclc}$ by using Juttner's algorithm [20] (Line 16). After that, it takes $O(|V| + |E|)$ time to obtain the adjusted path $p_{h,h'}^{adj}$ (Lines 17-18), and the feasibility examination for the adjusted path takes $O\left(|H|^2 \cdot (|V| + |E|) \cdot |K_H(t)|\right)$ time (Line 19), which has been proved in Lemma 2. After the examination, the path replacement takes $O(|V| + |E|)$ time and the merging operation of J_h and $J_{h'}$ takes $O(1)$ time (Line 22). However, if the above cases can not be met, then `Procedure 1` is in the **worst** case (Line 24), and it processes the adjusted paths as follows (Lines 25). The operation takes $O(|K_H(t)|)$ time for searching a task whose delay is greater than $\widetilde{d_H^{net}}$. Then, the corresponding result which causes the greater delay are rejected. Then, it takes $O\left(log\left(|H|^2 + |H|\right)\right)$ time to remove elements from heap Q_{delay} and inserting new elements into Q_{delay} (Lines 29-30).

Thus, the total time complexity of `Procedure 1` in the worst case is $O\left(|H|^2 \cdot \left(|E|^2 log^4|E| + |K_H(t)||H|(|V| + |E|)\right)\right)$.

$\square$

Lemma 2. *Given an adjusted path p_h^{adj}, delay constraint $\widetilde{d_H^{net}}$, historical record set J_h, there exists a procedure* `Procedure 2` *for examining path p_h^{adj} is feasible or not, which takes $O\left(|H| \cdot |K_H(t)| + |H| \cdot |K_H(t)|\right)$ time.*

Proof. First, it conducts a for loop with $O(|J_h|)$ iterations (Line 1). The value of $|J_h|$ equals that of $|H|$. In each iteration, it calculates the routing delay of a path with function $d(\cdot)$ at a cost of $O\left((|V| + |E|) \cdot |K_H(t)|\right)$ (Line 3), the reason is that the calculations here are the same as that of Line 8, `Procedure 1`. Thus, the total time complexity of `Procedure 2` is $O\left(|H| \cdot (|V| + |E|) \cdot |K_H(t)|\right)$. $\square$

Lemma 3. *Given a bipartite graph $G_b = (V_b, U_b, E_b)$, there exists a procedure* `Procedure 3` *for determining the the sets of admitted tasks and multicasted results, which takes $O(|R_H|)$ time.*

Proof. The initialization of variables tasks $O(1)$ time (Lines 1-2). Then, it conducts a for loop with $O(|U_b|)$ iterations. The value of $|U_b|$ equals that of $|R_H|$. In each iteration, the operations task $O(1)$ time (Lines 4-24). Thus, the total time complexity of `Procedure 3` is $O(|R_H|)$. $\square$

Theorem 2. *Given a train set H with the initial locations and velocities of the trains, a MEC network $G = (V, E)$ with a set V of routers, a subset B of BSs, a subset $V_c \subseteq V$ of cloudlets, and a set E of links, task set $K_H(t)$ and their multicast-oriented computation results with delay requirements and resource demands, there is an algorithm* `HeuAlg` *for the network throughput maximization problem, which takes* $O(|V| \cdot (|H|^3|K_H(t)|(|V| + |E|) + |H|^2|E|^2\log^4|E| + |R_H|\log|R_H| + |K_H(t)|\log|K_H(t)|)$ $+|B| + |R_H| \cdot log(1/\epsilon^3) + 1/\epsilon^4)$*, where ϵ is a constant with $\epsilon > 0$.*

Proof. First, it takes $O(1)$ time to initialize the variables (Lines 1-3). Then, it takes $O(|B|)$ time to create the BS groups and the rejection of results and tasks takes $O(|R_H| \cdot log(1/\epsilon^3) + 1/\epsilon^4)$ time by using Lawler's fast approximation algorithm [21] (Line 4), where $\epsilon > 0$ is the accuracy for the 0-1 knapsack problem.

Then, it conducts a for loop with $O(|V|)$ iterations, and it processes each cloudlet $v_c \in V_c$ as follows. In each iteration, it constructs the auxiliary graph in $O(|V| + |E|)$ time and it finds the GST on the auxiliary graph in $O(|H| \cdot (|E| + |V|log|V| + |V| + log|H|))$ time by using Sun's $|H|$-approximation algorithm [22] (Line 6). The operation of inserting the obtained GST into queue Q takes $O(1)$ time (Line 7).

Then, it processes the GSTs in queue Q by using a while loop with $|V|$ iterations (Line 9-24). It pops out the first GST in the queue (Line 10). The calculation of the delay of a GST takes $O\left((|V| + |E|) \cdot |K_H(t)|\right)$ time (Line 11). It employs `Procedure` 1 to adjust the GST in $O\left(|H|^2 \cdot \left(|E|^2 log^4|E| + |K_H(t)||H|(|V| + |E|)\right)\right)$ time (Line 12). Set $K_H(t)$ and R_H can be obtained in $O(1)$ time (Line 14). The sorting operations of Line 16 and Line 17 take $O(|K_H(t)|log|K_H(t)|) + O(|R_H|log|R_H|)$ time. The construction of the bipartite graph tasks $O(|K_H(t)| + |R_H|)$ time (Line 18). Then, it employs `Procedure` 3 to admit the tasks and multicast the results in $O(|R_H|)$ time (Line 19). The rest operations of obtaining the GST with the maximum throughput and returning the feasible task set and result set take $O(1)$ time (Lines 20-23).

Thus, the total time complexity of `HeuAlg` is $O(|V| \cdot (|H|^3|K_H(t)|(|V| + |E|) + |H|^2|E|^2\log^4|E| + |R_H|\log|R_H| + |K_H(t)|\log|K_H(t)|) + |B| + |R_H| \cdot log(1/\epsilon^3) + 1/\epsilon^4)$.

□

7 Wireless Communication Parameters

The wireless communication parameters are listed in Table 2, which are set according to [17, 23–25].

Table 2: Wireless Communication Parameters

Parameters	Value
Length of Train	200m
Distance between BS and Railways	100m
BS Antenna Height	32m
AP Antenna Height	1.5m
Coverage Radius of a BS	1000m
Handover Time	100ms
Number of Subcarriers	1024
Bandwidth of Subcarrier W_b	15kHz
Carrier Frequency f	2Ghz
Symbol Duration τ	1/14 ms
BS Transmit Power P_b	53dBm
Power Density of Background Noise σ	-145dBm/Hz
Path Loss Exponent ϵ	2

References

[1] B. Li, J. Xiong, B. Liu, L. Gui, M. Qiu, and Z. Shi, "Cache-based popular services pushing on high-speed train by using converged broadcasting and cellular networks," *IEEE Transactions on Broadcasting*, vol. 65, no. 3, pp. 577–588, 2018.

[2] J. Xiong, H. Xie, B. Liu, B. Li, and L. Gui, "Cooperative caching services on high-speed train by reverse auction," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9437–9449, 2021.

[3] P. Dong, T. Zheng, X. Du, H. Zhang, and M. Guizani, "Svcc-hsr: Providing secure vehicular cloud computing for intelligent high-speed rail," *IEEE Network*, vol. 32, no. 3, pp. 64–71, 2018.

[4] H. Chen, B. Jiang, W. Chen, and Z. Li, "Edge computing-aided framework of fault detection for traction control systems in high-speed trains," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1309–1318, 2020.

[5] Y. Liu, S. Lin, W. Li, and Z. Zhong, "Vehicular edge computing model for fault detection and diagnosis of high-speed train," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2020.

[6] Q. Zhang, H. Zheng, and Z. Zhong, "Energy-aware dynamic computation offloading in high-speed railway networks with d-tdd," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, pp. 1–6, IEEE, 2020.

[7] H. Li, Y. Sun, Y. Zhang, B. Jin, Z. Wang, W. Wu, and C. Fang, "Mobility-aware predictive computation offloading and task scheduling for mobile edge computing networks," in *2021 7th International Conference on Computer and Communications (ICCC)*, pp. 1349–1354, IEEE, 2021.

[8] L. Li, Y. Niu, S. Mao, B. Ai, Z. Zhong, N. Wang, and Y. Chen, "Resource allocation and computation offloading in a millimeter-wave train-ground network (early access)," *IEEE Transactions on Vehicular Technology*, pp. 1–16, 2022.

[9] F. Z. Bousbaa, N. Lagraa, C. A. Kerrache, F. Zhou, M. B. Yagoubi, and R. Hussain, "A distributed time-limited multicast algorithm for vanets using incremental power strategy," *Computer Networks*, vol. 145, pp. 141–155, 2018.

[10] S. Roger, D. Martin-Sacristan, D. Garcia-Roger, J. F. Monserrat, P. Spapis, A. Kousaridas, S. Ayaz, and A. Kaloxylos, "Low-latency layer-2-based multicast scheme for localized v2x communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 2962–2975, 2018.

[11] H. Bao, C. Huang, Z. Tang, Q. Li, Q. Ni, X. Dong, and B. Fu, "Coded multicasting in cache-enabled vehicular ad hoc network," *Computer Networks*, vol. 159, pp. 157–170, 2019.

[12] P. Keshavamurthy, E. Pateromichelakis, D. Dahlhaus, and C. Zhou, "Cloud-enabled radio resource management for co-operative driving vehicular networks," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2019.

[13] A. J. Kadhim and S. A. H. Seno, "Energy-efficient multicast routing protocol based on sdn and fog computing for vehicular networks," *Ad Hoc Networks*, vol. 84, pp. 68–81, 2019.

[14] Y. Hui, Z. Su, and T. H. Luan, "Collaborative content delivery in software-defined heterogeneous vehicular networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 575–587, 2020.

[15] P. Keshavamurthy, E. Pateromichelakis, D. Dahlhaus, and C. Zhou, "Resource scheduling for v2v communications in co-operative automated driving," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2020.

[16] S. Li, L. Yang, and Z. Gao, "Distributed optimal control for multiple high-speed train movement: An alternating direction method of multipliers," *Automatica*, vol. 112, p. 108646, 2020.

[17] J. Xu, Z. Wei, Z. Lyu, L. Shi, and J. Han, "Throughput maximization of offloading tasks in multi-access edge computing networks for high-speed railways," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9525–9539, 2021.

[18] N. Garg, G. Konjevod, and R. Ravi, "A polylogarithmic approximation algorithm for the group steiner tree problem," *Journal of Algorithms*, vol. 37, no. 1, pp. 66–84, 2000.

[19] T. Liu and W. Liao, "Multicast routing in multi-radio multi-channel wireless mesh networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 10, pp. 3031–3039, 2010.

[20] A. Juttner, B. Szviatovski, I. Mécs, and Z. Rajkó, "Lagrange relaxation based method for the qos routing problem," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 859–868, IEEE, 2001.

[21] E. L. Lawler, "Fast approximation algorithms for knapsack problems," *Mathematics of Operations Research*, vol. 4, no. 4, pp. 339–356, 1979.

[22] Y. Sun, X. Xiao, B. Cui, S. Halgamuge, T. Lappas, and J. Luo, "Finding group steiner trees in graphs with both vertex and edge weights," *Proceedings of the VLDB Endowment*, vol. 14, no. 7, pp. 1137–1149, 2021.

[23] L. Tian, J. Li, Y. Huang, J. Shi, and J. Zhou, "Seamless dual-link handover scheme in broadband wireless communication systems for high-speed rail," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 4, pp. 708–718, 2012.

[24] C. Zhang, P. Fan, K. Xiong, and P. Fan, "Optimal power allocation with delay constraint for signal transmission from a moving train to base stations in high-speed railway scenarios," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5775–5788, 2015.

[25] L. Wang, B. Ai, Y. Niu, X. Chen, and P. Hui, "Energy-efficient power control of train–ground mmwave communication for high-speed trains," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7704–7714, 2019.