An optimal algorithm for automated truck freight transportation via lane reservation strategy — Source link

Yunfei Fang, Feng Chu, Said Mammar, Ada Che

Institutions: Northwestern Polytechnical University

Published on: 01 Jan 2013 - Transportation Research Part C-emerging Technologies (Pergamon)

Topics: Flow network and Reservation

An optimal algorithm for automated truck freight transportation via lane reservation strategy

Yunfei Fang [a,b], Feng Chu [a], Saïd Mammar [a,*], Ada Che [c]

[a] *Laboratoire d'Informatique, Biologie Intégrative et Systèmes Complexes (IBISC), EA 4526, Université d'Evry Val d'Essonne, 40 rue du Pelvoux, 91020 Evry Cedex, France*
[b] *Institut Charles Delaunay – Laboratoire d'Optimisation des Systèmes Industriels (ICD-LOSI), UMR CNRS 6279, Université de Technologie de Troyes, BP 2060, 10010 Troyes Cedex, France*
[c] *School of Management, Northwestern Polytechnical University, 127 Youyi West Road, Xi'an, Shaanxi 710072, PR China*

This paper investigates an automated truck transportation problem via lane reservation strategy. The focus of the problem is to design lane reservation based paths for time-efficient transportation. The lane reservation strategy requires to select some existing general-purpose lanes from a transportation network and convert them to automated truck lanes in order to ensure the time-guaranteed transportation. However, such conversion may cause traffic impact such as increase of travel time on adjacent lanes due to the disallowing use of the automated truck lanes by the general-purpose vehicles. Thus, the problem aims at optimally designing the time-efficient truck paths while minimizing the impact on the overall network performance. The considered problem is formulated as an integer linear program and is demonstrated NP-hard. To solve it, an optimal algorithm based on the cut-and-solve method is proposed. Numerical computational results of randomly generated instances show the efficiency of the proposed algorithm compared with a referenced software package CPLEX 12.1.

1. Introduction

Transportation plays a significant role in our daily life, especially for a sustainable development of economy. Consequently, the efficiency of goods movement has received much attention and some logistic problems, such as vehicle routing problem and routing-scheduling problem, have been intensively studied by researchers during the last few decades (Laporte, 1992; Laporte, 2009; Koskosidis et al., 1992; Lee et al., 2006). However, increasing traffic congestion causes a lot of problems in freight transportation, such as higher and unpredictable travel time, fuel waste, low transport efficiency, and safety/environment issues. These problems prevent traditional freight transportation from a reliable, economic, efficient, and safe transport. As a result, stakeholders are looking for appropriate solutions for these problems. With recently technical innovation in Advanced Driver Assistance System (ADAS), including longitudinal (spacing and speed) and lateral (steering) control, road and obstacles perception, vehicle-vehicle and vehicle-infrastructure communication, and other related technologies, automated driving is being brought within reach (Shladover, 2010). Because of these features, the future automated trucks can offer a range of advantages, such as decreased drivers' stress, improved safety, reduced fuel consumption, and increased drivers' productivity, etc. Thus, the introduction of automated driving for trucks will be a promising solution to improve the efficient and economic viability of freight transportation (Kunze et al., 2011; Tsugawa et al., 2011).

* Corresponding author. Tel.: +33 (0)01 69 47 06 06.
E-mail addresses: yunfei.fang@iup.univ-evry.fr (Y. Fang), feng.chu@iup.univ-evry.fr (F. Chu), said.mammar@iup.univ-evry.fr (S. Mammar), ache@nwpu.edu.cn (A. Che).

One of the major concerns of successfully utilizing automated trucks in freight transportation is the safety issue. Unlike manually driven vehicles, automated trucks themselves must be able to detect all possible dangers and respond to them promptly and correctly. It is thus preferable to limit the driving environment of automated trucks for the safety. Therefore, dedicated truck lane is one of the smarter options for automated truck freight transportation for reasons as follows. Firstly, it is known that different types of vehicles have different operating characteristic, e.g., acceleration and braking capabilities. If automated trucks are allowed only on the dedicated truck lane, the deviation of the individual truck speed from the average speed is small. Thus the overall truck flow on the truck lane is smooth, which may lead to a potential decrease of accident (Tsao and Botha, 2002). Secondly, it is also said that the dedicated truck lanes limit the driving environment for automated trucks, which can make the technical challenge for automated driving tractable while achieving acceptable safety (Shladover, 2010). Thirdly, the dedicated truck lanes can meet the high time-efficient service required by future freight transportation. Since the truck lanes are used for freight transportation, excluding a large number of general-purpose vehicles, they can keep trucks from getting stuck in traffic and the travel time of the journey can be predictable. Hence, the dedicated truck lanes play a key role in the success of automated truck freight transportation.

Constructing new dedicated truck lanes require large cost and new land resource, and it is not always feasible nowadays. Hence appropriate and efficient utilization of the existing infrastructure becomes important. One opportunity is the concept of lane reservation strategy, which is to select and to convert some existing general-purpose lanes to special lanes, e.g., dedicated truck lanes. However, this strategy of lane conversion will reduce the number of general-purpose lanes, and make the remaining general-purpose lanes more congested. Traffic impact such as increase of travel time will be caused on the remaining general-purpose lanes. Princeton and Cohens (2011) reported that after one general-purpose lane of A1 motorway in Paris Region is reserved only for buses and taxis during the morning peak hours, the travel time on the remaining general-purpose lanes increases almost 53% for a period of three months observation in 2009. On the other hand, the impact of reserving a lane with busy traffic is different from that of reserving a lane with less traffic. It should be carefully considered selecting which existing lanes from the network so as to minimize the impact.

To the best of our knowledge, there are few studies about problems of minimizing the impact of reserved lanes by optimally selecting existing lanes from the network. Wu et al. (2009) studied an optimal routing design problem called lane reservation problem with time-constrained transportation (LRPTCT), which is motivated by performing time-guaranteed transportation tasks for large sport events. In the LRPTCT, some existing general-purpose lanes are optimally selected from the network and reserved for the task paths so that the reserved lanes can provide less travel time and tasks can be completed within the prescribed travel duration. The objective of the problem is to minimize the impact of reserved lanes on the overall network performance. In this paper, we study a routing design problem for automated truck freight transportation, which is called automated truck problem (ATP). The problem is to optimally design paths for automated trucks via lane reservation strategy. Both of the LRPTCT and ATP are intended to optimally design time-guaranteed paths via lane reservation strategy whereas the total impact of reserved lanes is minimized. However, there are significant differences between them. Firstly, the motivations of the two problems are different. The LRPTCT is intended for large sport events in urban network, lanes are necessarily reserved only when the tasks cannot be completed within the prescribed travel duration. Hence, the selected path is not totally reserved, i.e., some lanes in the path are reserved and the other lanes are general-purposed lanes. In the ATP, the path is designed for automated truck freight transportation. As stated above, it will be safer to separate automated trucks from general-purpose vehicles because of different operating characteristics of them. Therefore, the automated trucks are supposed to travel in an exclusive automated truck path, i.e., all the lanes in the truck path are reserved. Secondly, the mathematical model of the LRPTCT is a little different from the ATP one. But, the complexity of the LRPTCT is not demonstrated in the Wu et al. (2009) whereas the ATP is demonstrated to be NP-hard. Then, a heuristic algorithm was developed for the LRPTCT to obtain near-optimal solutions whereas an optimal algorithm based on the cut-and-solve method is developed for the ATP. At last, the problem assumptions in the ATP is more general as the source and prescribed travel duration are different for each task, whereas in the LRPTCT there is only one source and one prescribed travel duration for all the tasks. In the remainder of the paper, we use "reserved lane" instead of "automated truck lane" and "non-reserved lane" instead of "general-purpose lane" for the reason of easy description.

The contributions of this paper are, first of all, to propose a mathematical (quantitative) method to study an automated truck freight transportation problem via lane reservation strategy. The problem is demonstrated NP-hard based the proposed model. Another main contribution is that a cut-and-solve based optimal algorithm, for which some new techniques of piercing cuts are generated, is developed for the considered problem. Computational results show the efficiency of our proposed algorithm compared with the commercial Optimizer Solver of CPLEX 12.1.

The remainder of the paper is organized as follows. Section 2 describes the problem and formulates it as an integer linear programming model. Then the problem is demonstrated NP-hard. Section 3 presents the solution approach which is based on the cut-and-solve method. Section 4 reports computational results of numerical tests. Section 5 draws some conclusions and discusses future work.

2. Problem description

A transportation network can be represented by a directed graph $G = (V,A)$, where V is the set of nodes and A is the set of directed arcs. The nodes and arcs can be viewed as road intersections and road links in transportation network, respectively.

Give a set of tasks and source–destination (SD) pairs, each task corresponds to one SD pair. The considered problem aims at selecting lanes to be reserved and designing an exclusively reserved path for each task and its SD, so that each task can be completed within the prescribed travel duration. However, converting existing general-purpose lanes to reserved lanes reduces the number of general-purpose lanes. Thus, traffic impact such as increase of travel time on adjacent lane may be caused because the reserved lanes cannot be used by general-purpose vehicles. The objective of the problem is to minimize the traffic impact of all reserved lanes on the overall network performance.

In order to well describe the considered problem, some assumptions are made as follows. (1), there are at least two lanes in each road link allowing one lane to be reserved. Note that the impact of reserving the one and only one lane in certain road is too severe for the network to bear. (2), there is at most one reserved lane in each directed road link. Because the flow of a single task (number of vehicles/hour) is relatively low compared to the capacity of the reserved lanes, hence the reserved lanes can be shared by all tasks. (3), there is one and only one path for each task from its source to destination. Because one purpose of our study is to design paths for fully automated trucks or fleets of automated trucks with the first one driven by human with the others automatically follow it. It is natural that all the automated trucks of each task travel together in only one path for the sake of safety and cost. In closing, the lanes in the same road link are supposed to be identical, as well as the corresponding parameters.

Fig. 1 is an simple example of the considered problem. Nodes 1 and 6, 4 and 3 are two SD pairs, with the prescribed travel duration 5 and 6, respectively. The optimal paths with minimal impact 7 are (1, 2, 5, 6) and (4, 1, 2, 3), respectively. The reserved lane in arc (1, 2) is shared by both tasks. To formulate the problem, some notations are given in Table 1. With these assumptions and notations, ATP can be formulated as the following integer linear program IP_0.

$$(IP_0): \quad \min \sum_{(i,j)\in A} c_{ij} z_{ij} \tag{1}$$

$$s.t. \sum_{j:(s_k,j)\in A} x_{ks_kj} = 1, \quad \forall k \in K, s_k \in S \tag{2}$$

$$\sum_{i:(i,d_k)\in A} x_{kid_k} = 1, \quad \forall k \in K, d_k \in D \tag{3}$$

$$\sum_{j:(j,i)\in A} x_{kji} = \sum_{j:(i,j)\in A} x_{kij}, \quad \forall k \in K, \forall i \neq s_k, d_k \tag{4}$$

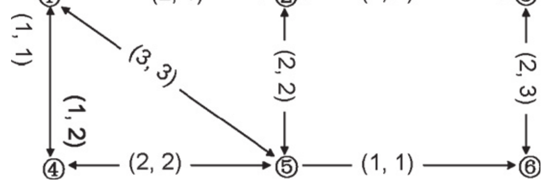
$$\sum_{(i,j)\in A} \tau_{ij} x_{kij} \leqslant p_k, \quad \forall k \in K \tag{5}$$

$$x_{kij} \leqslant z_{ij}, \quad \forall (i,j) \in A, \forall k \in K \tag{6}$$

$$x_{kij} \in \{0,1\}, \quad \forall (i,j) \in A, \forall k \in K \tag{7}$$

$$z_{ij} \in \{0,1\}, \quad \forall (i,j) \in A \tag{8}$$

The objective function (1) is to minimize the impact of all reserved lanes. Constraint (2) (resp. (3)) represents that there is one and only one arc for task k leaving its source s_k (resp. arriving at its destination d_k). Constraint (4) is the flow conservation constraint of task k for the nodes which are neither s_k nor d_k. Thus constraints (2)–(4) together ensure that there is one and only one path for each task from its source to destination. Constraint (5) is the travel duration constraint. The left hand item is total travel duration of the task from its source to destination, i.e., the sum of travel time along each lane of its path. The right hand item is the prescribed travel duration within which task must be completed. Thus, (5) means that the total travel duration for task k from its source to destination should not exceed the prescribed travel duration. The prescribed travel duration specifies the requirement of time-efficient transportation. It is dependent on the total travel distance and/or the transported goods. For example, if perishable goods (e.g., sea food) are transported from a port to some inland customers, the total duration of the transportation should be within several hours. Otherwise, the goods would be not fresh or becoming worse. Constraint (6) guarantees that there is a reserved lane in (i,j) in the path of task k if and only if this lane is reserved. Constraints (7) and (8) are binary constraints on the decision variables.



Two SD pairs are ①-⑥ and ④-③, whit the given travel time 5 and 6, respectively.
The numbers in the parenthesis are (travel time in a reserved lane, impact).

Fig. 1. Example of ATP.

Table 1
Notations of the problem.

Sets and parameters	
K:	set of tasks, $k \in K$
S:	set of source nodes of tasks, $s_k \in S \subseteq V$, $\forall k \in K$
D:	set of destination nodes of tasks, $d_k \in D \subseteq V$, $\forall k \in K$
p_k:	prescribed travel duration to complete task k, $\forall k \in K$
τ_{ij}:	travel time in a reserved lane in (i,j), $\forall (i,j) \in A$
c_{ij}:	traffic impact of a reserved lane in (i,j), $\forall (i,j) \in A$
Decision variables	
x_{kij}:	$x_{kij} = 1$, if task k uses a reserved lane in (i,j); and otherwise $x_{kij} = 0$, $\forall (i,j) \in A$, $\forall k \in K$
z_{ij}:	$z_{ij} = 1$, if there is a reserved lane in (i,j); and otherwise $z_{ij} = 0$, $\forall (i,j) \in A$

2.1. Complexity of ATP

If all the tasks start from the same source node and the prescribed travel duration to complete each task is large enough, then the reduced ATP corresponds to the Steiner tree problem in a directed graph, which is known to be NP-hard (Karp, 1972), hence ATP is NP-hard.

3. Solution method

In this section, an optimal algorithm based on the cut-and-solve method is proposed for ATP. The cut-and-solve method is an iterative search strategy, which is introduced for asymmetry traveling salesman problem (ATSP) (Climer and Zhang, 2006). It can be simply explained as follows. At the n-th iteration ($n \geqslant 1$), a *piercing cut* (P_n) is generated and it separates the solution space of the *current problem* (CP_n) into two subspaces (for the first iteration, CP_1 is defined as the original problem). The smaller subspace, called sparse space, corresponds to a *sparse problem* (SP_n) and the larger one, called remaining space, corresponds to a *remaining problem* (RP_n). As sparse space is relatively small, SP_n can be solved to optimality easily. In addition, the sparse space is a subspace of the solution space of the original problem, hence the optimal value of SP_n, denoted as UB_n, is an upper bound of the original problem. The *current best upper bound*, denoted as $UB_{\min}$, is updated as the minimum of UB_n and $UB_{\min}$. After solving SP_n, the sparse space is removed away and the solution space of CP_n is reduced. Because remaining space is large, it is difficult to solve RP_n optimally. So, we relax the integer constraints on decision variables (i.e., the variables are changed to real) and solve the linear relaxation problem of RP_n. An *associated lower bound* of RP_n, denoted as LB_n, is found. If $UB_{\min}$ is smaller than or equal to LB_n, then $UB_{\min}$ is also smaller than or equal to the optimal value of RP_n because LB_n is a lower bound on the optimal value of RP_n. Hence $UB_{\min}$ is the global optimal value and the iteration is terminated. Otherwise, if $UB_{\min}$ is greater than LB_n, the new current problem CP_{n+1} is defined as RP_n and a new iteration repeats. The principle of the cut-and-solve method is presented in Fig. 2. More details of the cut-and-solve method can be found in (Climer and Zhang, 2006).

ATP is a combinatorial optimization problem. For solving such problem, branch-and-bound and branch-and-cut are two conventional methods used to obtain optimal solutions. The implementation of the two conventional methods in depth-first or best-first manner results in the risk of either a large fruitless search of "wrong" subtrees with no optimal solutions or vast memory requirement of storing all active nodes for identifying the best current node. The cut-and-solve method overcomes the problem of making wrong choices in searching "wrong" subtrees, while keeping memory requirement negligible. The reasons are explained as follows. Different from the two conventional methods, the cut-and-solve method generates a search path instead of a search tree. Because the sparse problem is solved to optimality and no branching from it is needed. Hence, there is only one child for each node of the path and no "wrong" subtrees in which the search may get lost. On the other hand, only the current best upper bound and the remaining problem need to be saved in the cut-and-solve method, the memory requirement is negligible. Moreover, the search space is becoming smaller and smaller after each iteration in the cut-and-solve method, while it is reduced only when some conditions are satisfied in the two conventional methods. More details of the cut-and-solve method can be found in (Climer and Zhang, 2006).

It was declared that the implementation of the cut-and-solve method outperforms state-of-the-art solvers for some difficult real-world problem classes of the ATSP (Climer and Zhang, 2006). Later Yang et al. (2012) applied the cut-and-solve method combined with Fenchel cutting plane method to the single source capacitated facility location problem and improved the results of some benchmark instances in the literature. Due to the favorable properties of the cut-and-solve method, it is chosen as our solution approach. In the following, a pre-processing is implemented to reduced the search space of the original problem firstly. Then some new techniques of generating piercing cuts are developed for the studied problem. Finally an optimal algorithm based on the cut-and-solve method is presented.

3.1. Pre-processing

In this subsection, a pre-processing for IP_0 is implemented and then a new integer program IP_1 is obtained. As a consequence of this pre-processing, the search space of IP_0 is reduced. And this pre-processing is demonstrated to be helpful to the acceleration of the proposed algorithm in the next section.

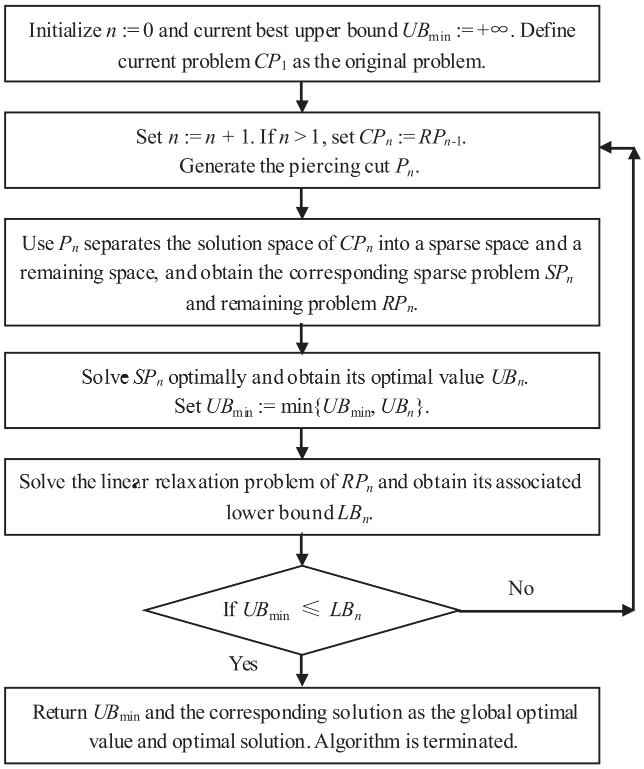

Initialize $n := 0$ and current best upper bound $UB_{\min} := +\infty$. Define current problem CP_1 as the original problem.
Set $n := n + 1$. If $n > 1$, set $CP_n := RP_{n-1}$. Generate the piercing cut P_n.
Use P_n separates the solution space of CP_n into a sparse space and a remaining space, and obtain the corresponding sparse problem SP_n and remaining problem RP_n.
Solve SP_n optimally and obtain its optimal value UB_n. Set $UB_{\min} := \min\{UB_{\min}, UB_n\}$.
Solve the linear relaxation problem of RP_n and obtain its associated lower bound LB_n.
If $UB_{\min} \leqslant LB_n$
No
Yes
Return $UB_{\min}$ and the corresponding solution as the global optimal value and optimal solution. Algorithm is terminated.

Fig. 2. Principle of the cut-and-solve method.

Let $dis(i,j)$ denote the shortest travel time from i to j in an entirely reserved path. This can be achieved by the Dijkstra's shortest path algorithm. Suppose that (s_k,j) is an arc outgoing from the source node s_k of task k. Set E_1 is defined as follows.

$$E_1 = \{x_{ks_kj} | \quad \tau_{s_kj} + dis(j, d_k) > p_k, \forall k \in K, (s_k, j) \in A\} \tag{9}$$

By the definition of E_1, the sum of the travel time in a reserved lane in (s_k,j) and the shortest travel time from j to d_k in an entirely reserved path is greater than p_k, which means that if (s_k,j) is in the path of task k, whatever other parts of its path are, the total travel time for task k from s_k to d_k is greater than p_k, i.e., the travel time constraint is violated. Hence in this case, (s_k,j) is not in the path of task k in an optimal solution. Similarly, E_2 is defined as follows.

$$E_2 = \{x_{kid_k} | \quad dis(s_k, i) + \tau_{id_k} > p_k, \forall k \in K, (i, d_k) \in A\} \tag{10}$$

where (i, d_k) is an arc incoming into the destination node d_k of task k. E_2 indicates that if (i, d_k) is in the path of task k, the total travel time for task k from s_k to d_k is greater than the prescribed travel duration p_k. Then such (i, d_k) is not in the path of task k in an optimal solution.

In the pre-processing step, sets E_1 and E_2 are defined by (9) and (10), respectively. Then all the variable in E_1 and E_2 are set to zero. Apparently, the search space of IP_0 is reduced after this pre-processing, but no feasible solutions of IP_0 are removed. Hence in the following steps of the proposed algorithm, a new integer program IP_1 is solved instead of IP_0. It is defined as follows.

$$(IP_1): \quad \min \sum_{(i,j)\in A} c_{ij} z_{ij}$$

$$s.t. \text{ constraints } (2)-(8)$$

$$x_{ks_kj} = 0, \quad \forall x_{ks_kj} \in E_1 \tag{11}$$

$$x_{kid_k} = 0, \quad \forall x_{kid_k} \in E_2 \tag{12}$$

3.2. New techniques of generating piercing cut for ATP

The efficiency of the cut-and-solve method is critically dependent on the piercing cut. As explained previously, the sparse space removed by the piercing cut should be sparse enough such that the sparse problem can be solved relatively easily. Meanwhile, the sparse space should be large enough to contain at least one feasible solution of the original problem, otherwise the current best upper bound cannot be improved. In this part, new techniques of generating piercing cut are developed for the considered problem.

3.2.1. Definition of piercing cut, sparse problem and remaining problem

In the pre-processing step, some variables are fixed to zero. As a result, the search space is reduced. This inspires us an idea. If some variables have fixed values in sparse space, then its corresponding optimal solution can be found relatively easily. Let Φ_n $(n \geqslant 1)$ denote a set of some variables. Since all the variables are binary, the sum of the variables in Φ_n is equal to zero, or greater than or equal to one. If SP_n is satisfied with the first condition, then each variable in Φ_n has value of zero. SP_n can be solved easily. Now we define the piercing cut as the sum of the variables in set Φ_n is greater than or equal to one. Using this piercing cut, the current solution space is separated into a sparse space (with the constraint that the sum of the variables in Φ_n is equal to zero) and a remaining space (with the constraint that the sum of the variables in Φ_n is greater than or equal to one).

Now the question is how to obtain the variable set Φ_n. A tool called *reduced cost* is used to define it (Climer and Zhang, 2006). A variable's reduced cost, a lower bound on the increase of the value of the objective function if the value of the variable is increased by one unit (Climer and Zhang, 2006). For example, if a variable has a reduced cost of 10, the value of the objective function will increase at least by 10 if this variable increases one unit for the minimization problem. Φ_n is defined as a set of all decision variables whose reduced cost is greater than a given positive number (Climer and Zhang, 2006). However, unlike ATSP in which all the decision variables belong to the same level, in ATP there are two levels of decision variables: lane reservation variables z_{ij} and task path variables x_{kij}. The way used to define set Φ_n should be carefully determined according to the considered problem. In ATP, the change of value of one lane reservation variable may totally affect task paths because tasks travel in the reserved lanes only. For example, by formula (6) if $z_{ij} = 0$, then $x_{kij} = 0$ for all k, no tasks use (i,j). In addition, the objective of ATP is related to the lane reservation variables only. For these reasons, instead of defining Φ_n by considering both z_{ij} and x_{kij} in ATP, only z_{ij} is used to define it. Let $\psi(z_{ij})$ denote the reduced cost of z_{ij}. Then Φ_n $(n \geqslant 1)$ is defined as follows.

$$\Phi_n = \{z_{ij} | \psi(z_{ij}) > h_n, \forall (i,j) \in A\} \tag{13}$$

where h_n is a given positive number. The reduced cost of z_{ij} can be obtained by solving the linear relaxation problem of CP_n. The choice of the value of h_n is dependent on the distribution of reduced cost of z_{ij}, e.g., $0.1 \times \max\{\psi(z_{ij}) | \forall (i,j) \in A\}$. If the corresponding sparse space does not contain any feasible solutions of the original problem, we can try a larger h_n. Once Φ_n is obtained, the piercing cut P_n $(n \geqslant 1)$ is defined as follows.

$$(P_n): \quad \sum_{z_{ij} \in \Phi_n} z_{ij} \geqslant 1 \tag{14}$$

With this piercing cut, the current solution space is separated into two subspaces. And the optimal solution in the small one, which corresponds to SP_n, can be found relatively easily. By the cut-and-solve method, SP_n can be obtained by adding CP_n with the constraint of the sum of variables in Φ_n is equal to zero, while RP_n can be obtained by adding CP_n with the constraint of the sum of variables in Φ_n is greater than or equal to one. For $n \geqslant 2$, CP_n is defined as RP_{n-1}, then SP_n and RP_n can be defined as follows.

$$(SP_n): \quad \min \sum_{(i,j) \in A} c_{ij} z_{ij}$$

$$s.t. \text{ constraints } (2)\text{–}(8), (11) \text{ and } (12)$$

$$\sum_{z_{ij} \in \Phi_t} z_{ij} \geqslant 1, \quad t = 1, 2, \ldots, n-1 \tag{15}$$

$$\sum_{z_{ij} \in \Phi_n} z_{ij} = 0 \tag{16}$$

$$(RP_n): \quad \min \sum_{(i,j) \in A} c_{ij} z_{ij}$$

$$s.t. \text{ constraints } (2)\text{–}(8), (11), (12), (14) \text{ and } (15)$$

It is not difficult to see that CP_n $(n \geqslant 2)$ is represented as the objective function with the constraints (2)–(8), (11), (12) and (15). Notice that CP_1 is defined as IP_1, in which there is no (15). Hence by the principle of the cut-and-solve method, for $\forall n \geqslant 1$ we add (16) or (14) to CP_n, then obtain SP_n or RP_n.

3.2.2. Enlargement of set Φ_n

Via a preliminary computation, we find that for some problem instances the sparse problem is not easy to solve, which means that the corresponding sparse space may be not sparse enough. To obtain a smaller sparse space, we can simply choose a smaller h_n. However, sometimes a large proportion of z_{ij} have reduced cost of zero, the previous way of defining Φ_n is no longer suitable. According to the cut-and-solve method, the global optimal solution is provided by the sparse problem. The variables in Φ_n have values of zero in SP_n, hence we hope that the variables in Φ_n are not basic variables in the global optimal solution. By analyzing the preliminary computational results, we find that the variables $z_{s_k j}$ or $z_{i d_k}$ with small values in the optimal solution of the linear relaxation problem of CP_n are unlikely basic variables in the global optimal solution. For example, $z_{s_kj_1}$, $z_{s_kj_2}$ and $z_{s_kj_3}$ are all the arcs outgoing from the source s_k and have values of 0.5, 0.3 and 0.2, respectively. $z_{s_kj_2}$ and $z_{s_kj_3}$ have small values 0.2 and 0.3, and they are unlikely basic variables in the global optimal solution. However this is not always true. In the following, set Φ_n is enlarged in a way which is based on the above observation. The computational results of this implementation is presented in the next section.

Let $\varphi(z_{ij})$ denote the value of z_{ij} in the optimal solution of the linear relaxation problem of CP_n. And define $b_{kn} = \max\{\varphi(z_{s_kj})|(s_k,j) \in A\}, \forall k \in K$ and $b'_{kn} = \max\{\varphi(z_{id_k})|(i,d_k) \in A\}, \forall k \in K$. For $n \geqslant 1$, two sets Ω_n and Ω'_n are defined as follows.

$$\Omega_n = \{z_{s_kj}|\varphi(z_{s_kj}) < b_{kn}, \forall k \in K, (s_k,j) \in A\} \tag{17}$$

$$\Omega'_n = \{z_{id_k}|\varphi(z_{id_k}) < b'_{kn}, \forall k \in K, (i,d_k) \in A\} \tag{18}$$

Then set Φ'_n $(n \geqslant 1)$ is defined as follows.

$$\Phi'_n = \Phi_n \cup \Omega_n \cup \Omega'_n \tag{19}$$

By this definition, set Φ'_n is obtained by adding some z_{s_kj} and z_{id_k} to Φ_n. The larger Φ'_n is, the smaller sparse space is. On the other hand, (s_k,j) and (i,d_k) are arcs outgoing from the source s_k and incoming into the destination d_k of task k, respectively. As a result, there are less arcs which can be used by task k to depart from s_k or arrive at d_k. In this sense, SP_n may be solved easily. In addition, Φ'_n will be used in our proposed algorithm in the following.

3.2.3. Reduction of sparse problem and remaining problem

In this part, reduction of the sparse problem and remaining problem are made. Firstly, we define the new sparse problem SP'_n $(n \geqslant 2)$ and new remaining problem RP'_n $(n \geqslant 2)$.

$$(SP'_n): \quad \min \sum_{(i,j)\in A} c_{ij}z_{ij}$$
$$s.t. \text{ constraints } (2)-(8), (11), (12) \text{ and } (16)$$
$$\sum_{z_{ij}\in\Phi_{n-1}\setminus\Phi_n} z_{ij} \geqslant 1 \tag{20}$$

$$(RP'_n): \quad \min \sum_{(i,j)\in A} c_{ij}z_{ij}$$
$$s.t. \text{ constraints } (2)-(8), (11), (12) \text{ and } (14)$$

Then we have the following theorem.

Theorem 1. *For $n \geqslant 2$, if*

$$\Phi_1 \supseteq \cdots \supseteq \Phi_{n-1} \supseteq \Phi_n \tag{21}$$

is true, then

(a) *SP'_n is equal to SP_n,*
(b) *RP'_n is equal to RP_n.*

Proof. The constraints are the same in SP'_n and SP_n except that (20) and (15) are different. In the following we prove the equivalence of (20) and (15). For $\forall t < n$, we have $\Phi_t \supseteq \Phi_{n-1} \supseteq \Phi_{n-1}\setminus\Phi_n$ by (21). Then $\Phi_t = (\Phi_t\setminus(\Phi_{n-1}\setminus\Phi_n)) \cup (\Phi_{n-1}\setminus\Phi_n)$ and $(\Phi_t\setminus(\Phi_{n-1}\setminus\Phi_n)) \cap (\Phi_{n-1}\setminus\Phi_n) = \emptyset$. Thus $\forall t < n$, we have

$$\sum_{z_{ij}\in\Phi_t} z_{ij} = \sum_{z_{ij}\in(\Phi_t\setminus(\Phi_{n-1}\setminus\Phi_n))} z_{ij} + \sum_{z_{ij}\in(\Phi_{n-1}\setminus\Phi_n)} z_{ij} \geqslant \sum_{z_{ij}\in(\Phi_{n-1}\setminus\Phi_n)} z_{ij} \geqslant 1$$

The last inequality holds by (20). This indicates (15) is true. Hence (15) can be deduced from (20). On the other hand, we have (15) in SP_n, or $\sum_{z_{ij}\in\Phi_{n-1}} z_{ij} \geqslant 1$. By (21), we obtain $\Phi_{n-1} = (\Phi_{n-1}\setminus\Phi_n) \cup \Phi_n$ and $(\Phi_{n-1}\setminus\Phi_n) \cap \Phi_n = \emptyset$. Hence

$$\sum_{z_{ij}\in\Phi_{n-1}} z_{ij} = \sum_{z_{ij}\in(\Phi_{n-1}\setminus\Phi_n)} z_{ij} + \sum_{z_{ij}\in\Phi_n} z_{ij} = \sum_{z_{ij}\in(\Phi_{n-1}\setminus\Phi_n)} z_{ij}$$

The last equation is obtained because $\sum_{z_{ij}\in\Phi_n} z_{ij} = 0$ holds by (16). Then we obtain $\sum_{z_{ij}\in(\Phi_{n-1}\setminus\Phi_n)} z_{ij} \geqslant 1$ by (15). Hence (20) can be deduced from (15). Therefore SP'_n is equal to SP_n.

The proof of (b) is similar. The difference is that there is no (15) in RP'_n. We need to prove that (15) is redundant in RP'_n. $\forall t < n$, by (21), we have $\Phi_t = (\Phi_t\setminus\Phi_n) \cup \Phi_n$ and $(\Phi_t\setminus\Phi_n) \cap \Phi_n = \emptyset$. Hence

$$\sum_{z_{ij}\in\Phi_t} z_{ij} = \sum_{z_{ij}\in(\Phi_t\setminus\Phi_n)} z_{ij} + \sum_{z_{ij}\in\Phi_n} z_{ij} \geqslant \sum_{z_{ij}\in\Phi_n} z_{ij} \geqslant 1$$

The last inequality is obtained by (14). Hence we have $\sum_{z_{ij}\in\Phi_t} z_{ij} \geqslant 1, \forall t < n$, which indicates that (15) is redundant in RP'_n. Therefore RP'_n is equal to RP_n. $\square$

The $n-1$ equalities in (15) are reduced to only one equality in (20) in SP'_n and totally removed from RP'_n. Both SP'_n and RP'_n have less constraints than SP_n and RP_n, respectively. Finally the set Φ''_n $(n \geqslant 1)$ used in the proposed algorithm is defined as follows.

$$\Phi''_n = \{z_{ij} | z_{ij} \in \Phi'_n \cap \Phi''_{n-1}, \forall (i,j) \in A\} \tag{22}$$

where Φ'_n is defined by (19) and Φ''_0 is defined as Φ'_1. It is not difficult to see that $\Phi''_1 \supseteq \cdots \supseteq \Phi''_{n-1} \supseteq \Phi''_n$ is satisfied. The piercing cut P''_n $(n \geqslant 1)$ is defined as follows

$$(P''_n): \quad \sum_{z_{ij}\in\Phi''_n} z_{ij} \geqslant 1 \tag{23}$$

SP''_n $(n \geqslant 2)$ and RP''_n $(n \geqslant 1)$ are defined as follows.

$$(SP''_n): \quad \min \sum_{(i,j)\in A} c_{ij} z_{ij}$$

$$s.t. \text{ constraints } (2)\text{—}(8), (11) \text{ and } (12)$$

$$\sum_{z_{ij}\in\Phi''_{n-1}\setminus\Phi''_n} z_{ij} \geqslant 1 \tag{24}$$

$$\sum_{z_{ij}\in\Phi''_n} z_{ij} = 0 \tag{25}$$

$$(RP''_n): \quad \min \sum_{(i,j)\in A} c_{ij} z_{ij}$$

$$s.t. \text{ constraints } (2)\text{—}(8), (11), (12) \text{ and } (23)$$

Notice that CP''_1 is still defined as IP_1 and CP''_n $(n > 1)$ is defined as RP''_{n-1}. The difference between SP''_1 and SP''_n $(n \geqslant 2)$ is that there is no (24) in SP''_1.

3.3. A cut-and-solve based optimal algorithm

The overall algorithm, called Algorithm 1, is presented as follows. The optimality of the algorithm is guaranteed by Theorem 2.

Theorem 2. *When*Algorithm 1*is terminated, it can find an optimal solution of the original problem.*

Proof. If Algorithm 1 is terminated, i.e., $UB_{\min}$ is smaller than or equal to LB_n. Then $UB_{\min}$ is also smaller than or equal to the optimal value of RP_n because LB_n is a lower bound on the optimal value of RP_n. This implies that there is no better solution value than $UB_{\min}$ in the solution space of RP_n. Therefore $UB_{\min}$ is the global optimal value and the corresponding solution is the global optimal solution. $\square$

Algorithm 1. An optimal algorithm for solving ATP

```
1:  Implement pre-processing for $IP_0$: define sets $E_1$ and $E_2$ and set all the variables in them to zero, obtain a new
    integer program $IP_1$.
2:  Initialize $n := 0$ and current best upper bound $UB_{\min} := +\infty$. Set current problem $CP''_1 := IP_1$.
3:  Solve the linear relaxation problem of $CP''_1$ and obtain variables' reduced cost.
4:  **repeat**
5:    Set $n := n + 1$. If $n > 1$, set $CP''_n := RP''_{n-1}$. Define set $\Phi''_n$ by (22) and piercing cut $P''_n$ by (23).
6:    Use $P''_n$ to separate the solution space of $CP''_n$ and obtain sparse problem $SP''_n$ and remaining problem $RP''_n$.
7:    Solve $SP''_n$ and obtain its optimal value $UB_n$. Set $UB_{\min} := \min\{UB_{\min}, UB_n\}$.
8:    Solve the linear relaxation problem of $RP''_n$ and obtain its associated lower bound $LB_n$ and variables' reduced cost.
9:  **until**$UB_{\min} \leqslant LB_n$
10: Return $UB_{\min}$ and the corresponding solution as the global optimal value and optimal solution, respectively.
    Algorithm is terminated.
```

4. Computational results

The algorithm was implemented in Visual C++ combined with a software package CPLEX 12.1. The linear relaxation problem of the remaining problem and the sparse problem were solved by the CPLEX LP and MIP solvers in default mode, respectively. Computational experiments were performed on a PC with 3.00 GHz CPU and 4.00 GB RAM. Sixty-two problem sets and five instances for each set were randomly generated in order to evaluate the performance of the proposed algorithm. They were generated in the following way.

The graph $G(V,A)$ was generated based on the network model proposed by Waxman (1988). The nodes of the graph were randomly distributed in a rectangular area $[0,100] \times [0,100]$. The existence of an arc (i,j) was dependent on a probability function $\alpha \exp(-d(i,j)/\beta L)$, where $0 < \alpha, \beta \leqslant 1$, $d(i,j)$ was the Euclidean distance from nodes i to j, and L was the maximum distance between any two nodes. Parameter α was proportional to the number of arcs and a high value of β gave a high ratio of long arcs to short arcs. The following parameters were generated in the method which was based on Wu et al. (2009). The SD pairs were randomly selected from set V. The travel time in lanes τ'_{ij} and τ_{ij} were defined as d_{ij} and $f_{ij}d_{ij}$, respectively, where f_{ij} was randomly generated from [0.5,0.8]. The prescribed travel duration to complete task p_k was generated from $[dis(s_k,d_k),dis'(s_k,d_k)]$, where $dis(s_k,d_k)$ and $dis'(s_k,d_k)$ were the shortest travel time from s_k to d_k in an exclusively reserved path and in an exclusively non-reserved path, respectively.

It is very difficult to evaluate quantitatively the impact of reserved lanes on adjacent lanes. It can be found that the impact has a very close relation with the increase of travel time on adjacent lanes due to the disallowing use of the automated truck lanes by the general-purpose vehicles. Consequently, we evaluate the impact using the increase of the travel time, as is the case in Wu et al. (2009). In Wu et al. (2009), the impact is evaluated as $c_{ij} = \tau'_{ij}/(m_{ij}-1)$, where τ'_{ij} is the travel time in a general-purpose lane before implementing lane reservation and m_{ij} is number of lanes in arc (i,j), respectively. Although the context of the problem in Wu et al. (2009) is not identical to that of our problem, both problems have the same lane reservation idea, i.e., convert some existing general-purpose lanes to reserved lanes for special users only. The cause of the impact comes from the lane reservation strategy. Moreover, the actually statistical result in Princeton and Cohens (2011) showed that the travel time in the general-purpose lanes increases about 53% after one of three lanes is reserved in A1 highway in Paris, which is very close to the result (50%) obtained by the formula proposed by Wu et al. (2009). As stated above, the formula proposed by Wu et al. (2009) is applicable to our problem to some extent. Hence we adopt this formula to estimate the impact. Moreover, we have also conducted numerical experiment for sensitivity analysis of different setting of the impact to evaluate the performance of the algorithm.

The performance of the proposed algorithm was compared with the direct use of CPLEX in terms of the computational time (in CPU seconds) of finding an optimal solution. In addition, the efficiency of the pre-processing in Section 3.1 and the enlargement of set Φ_n in Section 3.2.2 was tested with random instances. For brevity, denote Algorithm 1' as the algorithm implementing all the steps of Algorithm 1 without the pre-processing step and Algorithm 1" as the algorithm implementing all the steps of Algorithm 1 without the enlargement of set Φ_n step, respectively. Let T, T', T'' and T_c denote the average computational time required by Algorithm 1, Algorithm 1', Algorithm 1" and CPLEX, respectively. The computational results are presented in Tables 2–5 and Figs. 3–6.

Table 2 gives the computational time of the three algorithms. We find that T is less than T' and T'' for all eight sets in Table 2. The average values of T', T'' and T are 173.23, 92.90 and 60.99 s, respectively. The minimal, maximal and average values of T'/T are 1.92, 7.82 and 2.84, respectively. The minimal, maximal and average values of T''/T are 1.02, 1.89 and 1.52, respectively. These results show that the pre-processing and the enlargement of Φ_n steps are useful for accelerating the proposed algorithm. Fig. 3a presents the computational time for sets 1–8. It can be seen from Fig. 3a that T increases gradually and T' increases rapidly when the size of problem increases. Fig. 3b presents the results of T'/T and T''/T for sets 1–8. T'/T ranges between 1.92 and 7.82, while T''/T ranges between 1.02 and 1.89. The curve of T'/T is above that of T''/T. This implies that the pre-processing step is more efficient than the enlargement of Φ_n step in accelerating the proposed algorithm.

Table 3 provides the computational results of our algorithm and CPLEX for five types of impact of reserved lanes. The first four types of impact are calculated as $l_r c_{ij} = l_r \tau'_{ij}/(m_{ij}-1), r = 1,2,3,4$, where l_1 = 1.0 and l_2, l_3 and l_4 are randomly generated

Table 2
Computational results of Algorithm 1, Algorithm 1' and Algorithm 1".

Set	$\|V\|$	$\|K\|$	$2\|A\|/\|V\|$	T'	T''	T	T'/T	T''/T
1	60	25	7	19.83	7.18	5.62	3.53	1.28
2	60	30	7	66.63	10.15	8.52	7.82	1.19
3	70	25	7	75.91	13.85	11.62	6.53	1.19
4	70	30	7	125.52	43.46	41.04	3.06	1.06
5	80	25	7	125.71	31.57	30.98	4.06	1.02
6	80	30	7	307.98	133.25	96.70	3.18	1.38
7	90	25	7	247.09	92.59	75.71	3.26	1.22
8	90	30	7	417.18	411.17	217.71	1.92	1.89
Average				173.23	92.90	60.99	2.84	1.52

Table 3
Computational results for various types of impact.

Set	Impact	$\|V\|$	$\|K\|$	$2\|A\|/\|V\|$	T	T_c	T/T_c
9	Type 1	60	15	8	6.90	10.17	0.68
10	Type 1	60	20	8	6.01	17.91	0.34
11	Type 1	70	20	8	5.38	15.22	0.35
12	Type 1	70	25	8	20.10	42.87	0.47
13	Type 1	80	20	8	36.97	97.86	0.38
14	Type 1	80	25	8	83.77	336.43	0.25
15	Type 2	60	15	8	6.59	13.72	0.48
16	Type 2	60	20	8	2.83	6.44	0.44
17	Type 2	70	20	8	4.09	14.20	0.29
18	Type 2	70	25	8	15.50	38.55	0.40
19	Type 2	80	20	8	56.84	134.85	0.42
20	Type 2	80	25	8	71.34	279.18	0.26
21	Type 3	60	15	8	5.17	8.18	0.63
22	Type 3	60	20	8	8.30	26.82	0.31
23	Type 3	70	20	8	5.24	15.23	0.34
24	Type 3	70	25	8	21.95	52.13	0.42
25	Type 3	80	20	8	40.47	96.42	0.42
26	Type 3	80	25	8	87.97	319.18	0.28
27	Type 4	60	15	8	9.20	14.12	0.65
28	Type 4	60	20	8	5.18	10.71	0.48
29	Type 4	70	20	8	5.49	19.48	0.28
30	Type 4	70	25	8	12.30	28.23	0.44
31	Type 4	80	20	8	33.13	82.00	0.40
32	Type 4	80	25	8	76.21	293.04	0.26
33	Type 5	60	15	8	5.33	9.93	0.65
34	Type 5	60	20	8	5.23	13.34	0.48
35	Type 5	70	20	8	5.58	16.24	0.28
36	Type 5	70	25	8	21.87	54.05	0.44
37	Type 5	80	20	8	50.88	119.36	0.40
38	Type 5	80	25	8	115.91	495.18	0.26
Average					27.73	89.37	0.31

Table 4
Computational results for various types of average node degree.

Set	$\|V\|$	$\|K\|$	$2\|A\|/\|V\|$	T	T_c	T/T_c
39	100	10	5	1.46	1.03	1.42
40	100	15	5	1.07	3.45	0.31
41	100	20	5	31.83	34.81	0.91
42	100	25	5	33.02	54.98	0.60
43	100	30	5	53.58	68.50	0.78
44	100	10	7	3.23	4.85	0.67
45	100	15	7	84.35	158.30	0.53
46	100	20	7	172.53	402.94	0.43
47	100	25	7	279.91	656.40	0.43
48	100	30	7	267.78	1227.46	0.22
49	100	10	12	21.03	38.71	0.54
50	100	15	12	76.00	567.30	0.13
51	100	20	12	701.71	3218.10	0.22
52	100	25	12	1751.32	5345.93	0.33
53	100	30	12	7189.59	>18000.00	<0.40
Average				711.23	>1985.52	<0.36

from [0.5, 1.0], [1.0, 1.5] and [0.5, 1.5], respectively. Notice that the intervals from which l_2, l_3 and l_4 are generated, the are used to generate small impact, large impact, small and large impact simultaneously, respectively. The fifth type of impact is randomly generated from [0.5, 10]. We observe that our algorithm is faster than CPLEX over all sets in Table 3. The average values of T and T_C are 27.73 and 89.37 s over sets 9–38. The average value of T/T_c is 0.31. The minimal values of T/T_c for each type of impact are 0.25, 0.26, 0.28, 0.26 and 0.26, respectively. The maximal difference among the five minimal T/T_c is 0.03. It can be seen from Fig. 4a that the changing trend of T for each type of impact is almost the same and the differences among

Table 5
Computational results for various sizes of problem.

Set	$\|V\|$	$\|K\|$	$2\|A\|/\|V\|$	T	T_c	T/T_c
54	110	10	7	2.37	4.57	0.52
55	110	15	7	11.45	45.05	0.45
56	120	15	7	45.80	90.49	0.51
57	120	20	7	166.41	417.17	0.40
58	130	20	7	438.96	1075.65	0.41
59	130	25	7	671.37	1689.28	0.40
60	140	25	7	765.84	2014.52	0.38
61	140	30	7	1172.22	3038.63	0.39
62	150	30	7	1543.43	4058.54	0.38
Average				536.31	1381.54	0.42

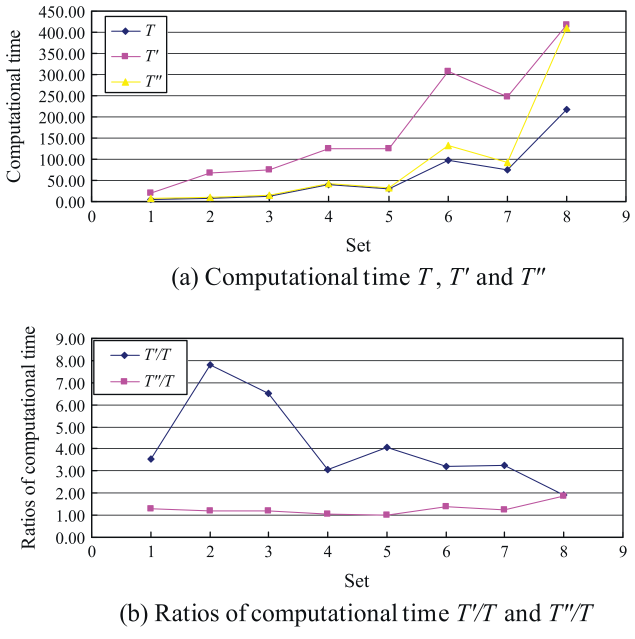
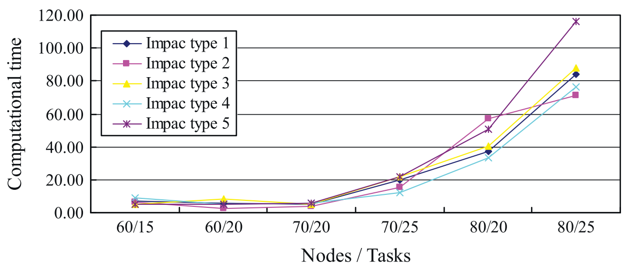


Fig. 3. Computational results of Algorithm 1, Algorithm 1' and Algorithm 1''.

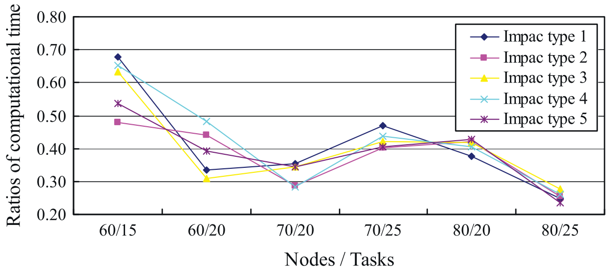
five curves vary slightly for the same number of nodes and tasks. Moreover, we find from Fig. 4b that the five curves of T/T_c also vary slightly for the same number of nodes and tasks. These results show that the performance of the proposed algorithm is stable in terms of impact. Hence, in the following tests the parameter impact is still calculated as $\tau'_{ij}/(m_{ij}-1)$.

Table 4 presents the computational results for $2|A|/|V|$ = 5, 7 and 12, respectively. We call $2|A|/|V|$ as average node degree, which denotes the average number of arcs connected with a node. The higher it is, the denser the network is. It can be seen from Table 4 that T is less than T_c over sets 40–53 and greater than T_c for set 39. CPLEX cannot find an optimal solution within 18000.00 s for set 53, while our algorithm only takes 7189.59 s. Moreover, T_c increase sharply when the average node degree increases. For example, the values of T_c are 54.98, 656.40 and 5345.93 s for sets 42, 47 and 52, respectively. The average value of T/T_c over all sets 39–53 is less than 0.36. It can be seen from Fig. 5a that T_c increases much sharply for sets 50, 51 and 52, while T increases gradually. Fig. 5b presents the results of three values of average node degree. Generally speaking, the values of T/T_c are small for the case of average node degree 12. These results show that: (1), the complexity of the problem increases with average node degree; (2) our proposed algorithm is more effective than CPLEX for a large value of average node degree.

In Table 5, we report the results of various sizes of problem. In Table 5, T is less than T_c over sets 54–62. The average values of T and T_c are 536.31 and 1381.54 seconds, respectively. The minimal, maximal and average values of T/T_c are 0.38, 0.52 and 0.42, respectively. We find from Fig. 6a that T increases gradually with the size of the problem, while T_c increases much more quickly. Fig. 6b presents the ratios of computational time T and T_c. As showed in Fig. 6b, T/T_c varies slightly when the size of problem increases.

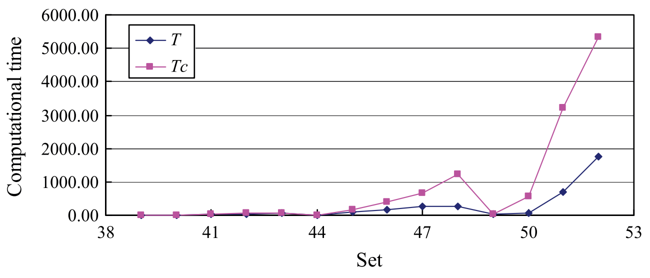


(a) Computational time T for various types of impact

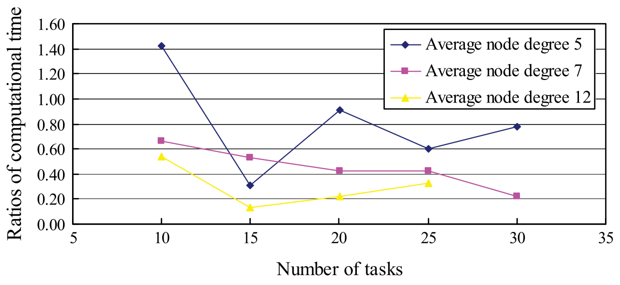


(b) Ratios of computational time T/T_c for various types of impact

Fig. 4. Computational results for various types of impact.



(a) Computational time T and T_c for various types of average node degree



(b) Ratios of computational time T/T_c for various types of average node degree

Fig. 5. Computational results for various types of average node degree.

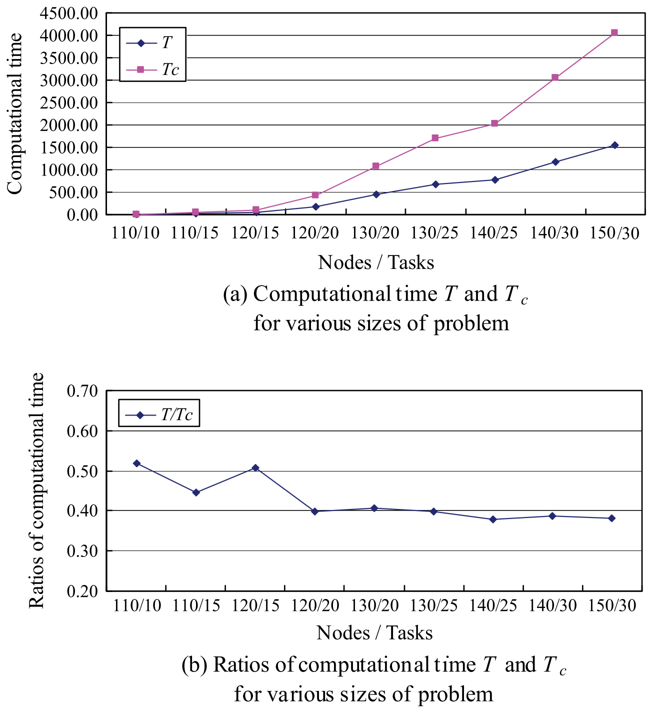


Fig. 6. Computational results for various sizes of problem.

5. Conclusions and future research

In this paper, we studied an automated truck freight transportation concept. The problem is to optimally select some existing general-purpose lanes from network and convert them to automated truck lanes via lane reservation strategy so that the automated trucks can travel fast in the truck lanes and time-efficient transportation paths can be guaranteed. The objective of the problem is to minimize the impact of the truck lanes on the overall network performance. The considered problem was formulated as an integer linear program and was demonstrated as an NP-hard problem. To solve the problem, an optimal algorithm based on the cut-and-solve method was proposed. In our algorithm, some new techniques of generating piercing cut were developed for the cut-and-solve method. Numerical computational results of random problem instances have shown that the proposed algorithm outperformed a referenced software package CPLEX 12.1 in finding an optimal solution.

The automated truck freight transportation via lane reservation strategy considered in this work can serve as a basic problem to provide a framework for further study of other more complex automated network design problems when manual and automated vehicles have to share the space through lane reservation. The results of our study can also be extended to other applications of lane reservation strategy, such as evacuation during emergency and hazardous materials routing. The final goal of our work is to develop a new methodology for the network design for automated truck freight transportation. As a first step of the study, the travel time along a link is assumed to have fixed values in the problem, which is far from the realistic case. To better model practical situation, dynamic factors such as non-constant link travel time function and traffic flow might be introduced into the problem formulation. Moreover, the evaluation of the impact caused by reserved lanes should be further investigated. Some theoretical studies of the problem might be performed in order to enhance problem solution finding.

Acknowledgements

This work was partially supported for the Cai Yuanpei program by the French Ministries of Foreign and European Affairs and the Higher Education and Research and the Chinese Ministry of Education under Grant No. 24021SH. The authors are grateful to the anonymour reviewers for their insightful remarks and suggestions on improving the quality of the paper.

References

Climer, S., Zhang, W., 2006. Cut-and-solve: an iterative search strategy for combinatorial optimization problems. Artificial Intelligence 170 (8–9), 714–738.
Karp, R., 1972. Reducibility Among Combinatorial Problems. Complexity of Computer Computations. Plenum Press, New York.
Koskosidis, Y., Powell, W., Solomon, M., 1992. An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints. Transportation Science 26 (2), 69–85.

Kunze, R., Haberstroh, M., Ramakers, R., Henning, K., Jeschke, S., 2011. Automated truck platoons on motorways – a contribution to the safety on roads. In: Automation, Communication and Cybernetics in Science and Engineering 2009/2010. Springer, Berlin Heidelberg.

Laporte, G., 1992. The vehicle routing problem: an overview of exact and approximate algorithms. European Journal of Operational Research 59 (3), 345–358.

Laporte, G., 2009. Fifty years of vehicle routing. Transportation Science 43 (4), 408–416.

Lee, Y., Jung, J., Lee, K., 2006. Vehicle routing scheduling for cross-docking in the supply chain. Computers & Industrial Engineering 51 (2), 247–256.

Princeton, J., Cohens, S., 2011. Impact of a dedicated lane on the capacity and the level of service of an urban motorway. Procedia Social and Behavioral Sciences 16, 196–206.

Shladover, S., 2010. Truck automation operational concept alternatives. In: Intelligent Vehicles Symposium. IEEE, San Diego, USA, pp. 1072–1077.

Tsao, H., Botha, J., 2002. Definition and evaluation of bus and truck automation operations concepts. Technical report, California Partners for Advanced Transit and Highways (PATH), Institute of Transportation Studies (UCB), UC Berkeley.

Tsugawa, S., Kato, S., Aoki, K., 2011. An automated truck platoon for energy saving. In: 2011 IEEE/RSJ International Conference Intelligent Robots and Systems, San Francisco, CA, USA, pp. 4109–4114.

Waxman, B., 1988. Routing of multipoint connections. IEEE Journal on Selected Areas in Communications 6 (9), 1617–1622.

Wu, Y., Chu, C., Chu, F., Wu, N., 2009. Heuristic for lane reservation problem in time constrained transportation. In: Proceedings of Automation Science and Engineering, Bangalore, India, pp. 543–548.

Yang, Z., Chu, F., Chen, H., 2012. A cut-and-solve based algorithm approach for the single-source capacitated facility location problem. European Journal of Operational Research 221 (3), 521–532.