ABOUT A SOLUTION TO THE PUBLIC TRANSPORTATION NETWORK PROBLEM IN THE CITY WITH SEVERAL OPERATING CENTERS

Pham Xuan Hinh(*)

Hanoi Metropolitan University

Abstract: *The problem of the city's public transportation network with several operating centers is an integer programming problem, if the number of variables is large enough, it is an NP-hard problem. It is not possible to find an exact solution to this problem in practice. In this paper, we propose a solution to the problem of the city's public transportation network with several operating centers by "simplifying" the problem step by step, after each step the problem becomes easier to solve. Finally, bring the problem to the group of optimal problems with solutions in polynomial time. Then, it can be applied to solve the problem of the public transportation network in Hanoi as well as other cities.*

Keywords: *Traffic schedule, public transportation, Hanoi.*

Received 22 December 2021

Revised and accepted for publication 26 January 2022

(*) Email: pxhinh@daihocthudo.edu.vn

1. INTRODUCTION

The goal of a city public transportation network is to meet the passenger transportation requirements set forth by the urban transportation authority, based on a survey of actual travel demand. This requirement is usually expressed in the form of a set of itineraries connecting basic intersections in the city. In the process of making these itineraries, the vehicles often have to make some other itineraries that are not included in the requirements. This is the unprofitable cost, and one of the important goals in the transportation industry is to reduce these costs, based on the rational arrangement of schedules and the allocation of routes to the operating centers. In fact, the bus network of a city is formed and developed through many stages, besides the development of the city itself. Although it may be designed "optimal" from the beginning, in the process of operating, with the expansion of the urban area and the continuous growth of the forces participating in traffic, the network will cannot keep the original "optimal" structure, due to the need to add new itineraries. Then, a requirement is raised that the schedule needs to be restructured in order to continue to have the optimality at a reasonable level. This paper is intended to propose a solution for finding such solutions. The model of urban traffic with many operating centers we have introduced and initially studied (see [5]). In this article, we will give the solution and after that, it can be applied to the public transportation network of Hanoi city as well as other cities.

2. CONTENT

2.1. Mathematical models

2.1.1. Several concepts and symbols

An *operating center* or *bus station* (depot) is a gathering place for vehicles, from which the vehicles depart to carry out the specified itineraries, and after finishing the day's itineraries, the vehicles must return to center. In the operating center, there is usually a garage for repairing and maintenance routine. The operating center (later often abbreviated as *the center* when there is no possibility of confusion) is denoted by d, and each d has *departure point* is d^{+}, *entry point* is d^{-}.

The set of all service centers for the city bus network is denoted by D, and the set of all departure and arrival points of these centers is denoted respectively by.

$$D^{+} = \ d^{+} / d \in D \ , \quad D^{-} = \ d^{-} / d \in D \ .$$

As mentioned, based on the survey results of the population's travel needs in reality, city traffic management agencies define service requirements for the public transportation network in the form of a set of *passenger itineraries* required to be made within the city (later also known as *mandatory itineraries*). Each such itinerary, denoted by t, has *a starting point* t^{-} (first stop) with *a departure time of* , s_t and a *final stop* with an t^{+} *arrival time* of e_t. We denote T the set of all required itineraries, and T^{-} (T^{+}, respectively) the set of all starting points (end stops) of the itineraries $t \in T$. So,

- *Berth arc* connecting *berth point* d^{-} with the final stop t^{+} of a certain mandatory itinerary. Similar to the above, we have the concept *of berth itinerary* and accompanied by the assumption of the existence of suitable berth itineraries for each required itinerary.

- *The linked arc* connects the end point of one itinerary $p \in T$ (ie p^{+}) with the starting point of another itinerary $q \in T$ (ie q^{-}). On this road, one needs to make a "transitional itinerary" if one wants to make the itinerary q after the itinerary has been made p, by the car that made this itinerary. This is possible when the two-stroke pair $p, q \in T$ is *compatible*. Specifically, we denote $\Delta_{p,q} \geq 0$ the time required to go from p^{+} to q^{-}. If $\Delta_{p,q} \leq s_q - e_p$ then we say that two itineraries p and q are *compatible* , and then we can create a *connected itinerary* (between the two) whose starting point is p^{+} (with a starting time of e_p) and an ending point. to be q^{-} (with an end time of s_q). For the convenience of later arguments, the associated itinerary between the two itineraries is compatible p and q is always considered to exist, even if $p^+ = p^-$ (at a possible cost of zero). The quantity $s_q - e_p$ considered to be *the transition timeout* between two itineraries p and q. When this time is large enough, compatibility is likely to occur. However, in practice, when this quantity is too large, the connection will be inefficient (because the waiting time to connect is too long, which means too much idle time for the vehicle and the driver). In order to avoid having to consider "compatible" itinerary pairs for which it is not practical to join them, it is common to consider only those pairs whose transition timeout does not exceed some pre-determined limit (depending on the destination in real situations, one can choose this limit between 40 and 120 minutes). If this limit is exceeded, one assumes $\Delta_{p,q} = \infty$ and considers the two itineraries to be unconnectable (compatibility condition not satisfied).

- The arc connecting *the berth point* of the operating center d with *its departure point*, i.e. d^-, d^+ , used to return the vehicle to the departure point after completing a schedule and returning to the station, is often called is *the reverse arc* . Itineraries on the retrograde arc will be called *internal itineraries* .

Each itinerary is assigned a *weight* (or *cost*) depending on the distance, travel time, vehicle type,... The weight of *the linked itinerary* can also include waiting time of the vehicle, driver's breaking time. The weight of the departure itinerary often has an additional cost of using the vehicle, while the weight of the internal itinerary is usually given as 0. Based on the weights of the itineraries, the cost of the whole schedule is calculated.

2.1.2. Graph of public transportation network

For each operating center $d \in D$ we define the following sets:

- $A_d^{bb} = \ t^-, t^+ \ \big|\ t \in T_d$ is the collection of all *roads to be implemented* (set out in the plan of the traffic management agency), which can be serviced by the center d.

- $A_d^{xuat} = \ d^+, t^- \ \big|\ t \in T_d$ is the set of all *departure arcs* (from the center d). The set of possible itineraries on these arcs (corresponding to different times) is called the set of departures (of the center d) and is denoted by A_d^{ht-x}.

- $A_d^{nhap} = \ t^+, d^- \ \big|\ t \in T_d$ is the set of all *berth arcs* (to the center d). The set of possible itineraries on these arcs (corresponding to different times) is called the set of landing itineraries (of the center d) and is denoted by A_d^{ht-n}.

- $A_d^{lk} = \ p^+, q^- \ \big|\ p, q \in T_d,\ e_p + \Delta_{p,q} \le s_q$ is the set of all *transition arcs* between compatible pairs of itineraries in the set T_d. The set *of linked itineraries* (which can be performed on these arcs) will be denoted by A_d^{ht-lk}. Thus, the set of *no-load trips* that the hub d can make is $A_d^{ht-kt} := A_d^{ht-x} \cup A_d^{ht-n} \cup A_d^{ht-lk} \cup \ d_0$.

Combined with the set of mandatory itineraries, T_d we have a *set of itineraries* denoted by A_d^{ht}. Thanks to linked itineraries, the set A_d^{ht} is "locally contiguous" in the sense that each itinerary p has at least one q successive itinerary both geographically and in time, in particular $p^- = q^+$ and $e_p = s_q$. Such a pair of itineraries is said to be *adjacent* , and then we can say it p is *the predecessor* of q or q a *successor* of p. The set of all pairs of contiguous itineraries in A_d^{ht} will be denoted by K_d (a subset of $A_d^{ht} \times A_d^{ht}$). For each itinerary $p \in A_d^{ht}$, we denote $K_d(p)$ the set of adjacent paths of p.

To visualize the traffic network structure, we can construct its graph based on the sets of roads defined above. First, we draw a graph of the network managed by the center d, which is *a directed graph* $D_d = \ V_d, A_d$, with

$V_d = \ d^+, d^- \ \cup T_d^- \cup T_d^+$ is *the set of vertices* , $A_d = A_d^{bb} \cup A_d^{xuat} \cup A_d^{nhap} \cup A_d^{lk} \cup \ d^-, d^+$ is *the set of arcs* of the graph.

The graph of the entire network will be the union of all the above graphs, more specifically $D = \ V, A$,

where $V = D^+ \cup D^- \cup T^+ \cup T^-$ is the set of *vertices* and $A = \dot{\bigcup}_{d \in D} A_d$ the set of *arcs* . Here $\dot{\bigcup}$ is the notation of *the disjoint* union, according to which if there is an arc belonging to many sets, when "merged" it will form many separate arcs (like adding an index indicating the set containing it).

Each required itinerary t that can be served by several centers (eg (a^-, a^+)) will be represented by multiple parallel arcs on the entire network graph $D = \ V, A$ (with the number of arcs exactly equal to the number of hubs that can be serviced it, ie $|G(t)|$).

2.2. The problem of setting up a traffic schedule in a transportation network with several operating centers

2.2.1. Objective function

The objective function (cost function) is established on the basis of the followings. For each no-load trip $a \in A_d^{ht-kt}$, we give a weight corresponding to it $c_a^d \in \mathbb{R}$, which is the operating cost of the itinerary a when performed by the center's vehicle d. In addition, we add to the weight of each departure itinerary a M sufficiently large number, representing the investment cost per locomotive (the number *M* is usually greater than the operating cost per trip). The costs on the required paths and inverse arcs are fixed, so they can be ignored in the optimization problem, that is, they can be zero.

2.2.2. Mathetical model

Assume there is an acceptable solution. For each $d \in D$ and $i, j \in A_d^{ht}$, a decision variable X_{ij}^d is used to denote that the two itineraries i, j are on the same central deployment schedule d and j are successors of i. Thus, we have $X_{ij}^d = 1$ if and only if $i, j \in K_d$ and these two itineraries are on the same schedule, and we have $X_{ij}^d = 0$ in all other cases. The symbol C_{ij}^d is the cost of making the itinerary j immediately following the itinerary i. With the concept of cost weighting mentioned above, it is possible to see $C_{ij}^d = c_i^d + c_j^d$, but note that then the weight of each itinerary is calculated twice because in the cycle there are always 2 adjacent itineraries to one already. for, so to make it more reasonable, it is recommended to multiply by a factor of 1/2. The goal of minimizing total costs is

$$\min \sum_{d \in D} \sum_{i,j \in K_d} C_{ij}^d X_{ij}^d . \qquad (1)$$

Provided that the variables X_{ij}^d can only take one of two values, 0 or 1, we see that the necessary and sufficient condition for a itinerary to $i \in T$ be executed exactly once would be:

$$\sum_{d \in D, j \in A_d^{ht}} X_{ij}^d = 1, \ \forall i \in T . \qquad (2)$$

We know that when a required route j is executed (within a schedule) there will be exactly one predecessor and one adjacent itineraries executed (within the same schedule). This condition is mathematically expressed by the following statements.

LEMMA 1 . *In terms of constraints (2), the following two conditions are equivalent:*

(*) *Each mandatory itinerary j that is executed will always have exactly one predecessor and a successor to it that is also executed in the same schedule as it;*

(**) $\sum_{k \in A_d^{ht}} X_{kj}^d - \sum_{k \in A_d^{ht}} X_{jk}^d = 0, \ \forall j \in T_d, \forall d \in D$. (3)

Proof . From (*) we have $\sum_{k \in A_d^{ht}} X_{kj}^d = \sum_{k \in A_d^{ht}} X_{jk}^d = 1$, and from here we can deduce (3). We only have to prove the opposite sign. Due to condition (2), for one $j \in T_d$ there are only 2 possibilities:

(i) $\sum_{k \in A_d^{ht}} X_{jk}^d = 0$, (ii) $\sum_{k \in A_d^{ht}} X_{jk}^d = 1$.

Case (i) means that the itinerary is j not executed on the same schedule as any of its adjacent routes (which belong to the center d). From (i) and condition (3) alo we can deduce $\sum_{k \in A_d^{ht}} X_{jk}^d = 0$, and this means that none of the predecessors of j (which are in the service domain of d) can be executed on the same schedule as it. Thus the itinerary itself j cannot be executed (by the center d) because otherwise, at least one of its adjacent itineraries must be executed. (Note that j not being enforced by the center d is not inconsistent with that $j \in T_d$ and must be enforced once, because T_d it is only the set of itineraries that d can serve, not the set of routes that d force must execute.)

Case (i) means that there exists only one itinerary that k_0 is the successor of to j be executed with it, and then condition (3) entails $\sum_{k \in A_d^{ht}} X_{kj}^d = 1$, i.e. there will be only one itinerary that k_1 is the predecessor of j and is executed with it. The statement has been proven. □

The number of vehicles used at the center is exactly equal to the number of departures made, i.e. equal to $\sum_{j \in A_d^{ht-x}} X_{d_0 j}^d$. Therefore, the constraint on the number of vehicles used at each center is shown as follows: $\lambda_d \le \sum_{j \in A_d^{ht-x}} X_{d_0 j}^d \le \kappa_d$, $\forall d \in D$. (4)

The integer condition 0 or 1 of the decision variable is rewritten as $X_{ij}^d \in \ 0,1\ ,\quad \forall d \in D,\quad \forall i,j \in A_d^{ht}.$ (5)

Thus, for an implementation to be acceptable, conditions (2)-(5) must be satisfied. The problem of finding the optimal solution is the problem of finding an acceptable solution with the smallest (1) cost function, that is, it can be described by (1)-(5). Later, for brevity, we will call it *problem* (MD).

As we can see above, for each possible *solution* , by using the decision variable, we can create a vector $X = (X_{ij}^d)_{d \in D, i, j \in A_d^{ht}}$ satisfying the constraints (2)-(5). The reverse is also true, thanks to the following lemma.

LEMMA 2 . *From the vector of decision variable values satisfying the conditions (2)-(5) we can establish an acceptable implementation plan* (*that is, an acceptable schedule set*).

Proof. The set of itineraries executed in the schedule will be determined through a set of vector coordinates that take the value of 1. This set of itineraries is distributed to the centers based on the index d. From the set of itineraries of each center, we will determine the *schedules* for that center, uniquely by conditions (2), (3) and (5). Indeed, taking a forced itinerary $a \in T$ and, according to condition (2), we find only one center d and one path $j \in A_d^{ht}$ such that $X_{aj}^d = 1$. This also means j that the successor of a and is executed on the same schedule as a. As said, Condition (3) means that there is only one itinerary l is the predecessor of a and is executed on the same schedule as a. Repeat the same process for both l and j you will find your predecessor l and successor's itineraries on the j same schedule $a.$.Keep going with this process until you come across a *berth* and a *departure itinerary*. This means that we have found a schedule containing the stated itinerary a. We denote this schedule as L_a. Continue, take another itinerary b in the volume T (outside the L_a aforementioned schedule), and do the same as above we will find another schedule L_b. Condition (2) shows that these itineraries cannot share the same itinerary. Since the set T is finited, after a finite number of times we will exhaust the set T and that is, find all the schedules. The lemma is proven.

2.2.3. Use another decision variable

Since the set of routes A_d^{ht} is many times larger than the set of required itineraries T_d, the problem size will be much smaller if the variable i, j runs only on the set T_d, rather than on the set A_d^{ht}. The fact that an associated itinerary k is performed after the required itinerary i always entails some required itinerary j that will be performed immediately after the itinerary k. In essence, this can be seen as the imperative that j is performed following the required passage i through the association path k, and thus can be viewed k as born as a consequence of performing two consecutive imperatives i, j (when considering only the set T_d). Since the associated itinerary k is uniquely determined from a pair of compatible itineraries $i, j \in T_d$, we can convent that when we say "two itineraries $i, j \in T_d$ are executed consecutively on the same schedule" to mean the the link between them is also done within the framework of that schedule. As mentioned, the departure (or arrival) itinerary can also be seen as a link between the required and internal itineraries d_0 and is therefore also determined from the pair of a mandatory and an internal itinerary.

The above analysis shows that we have the basis to "ignore" the set of linked itineraries as well as the departure and arrival itineraries, but only put the decision variable on the remaining set of itineraries, ie. set $\tilde{T}_d := T_d \cup \ d_0$. Specifically, for a given implementation and for $i, j \in \tilde{T}_d$, we have $X_{ij}^d = 1$ if and only if the pair of (i, j) routes are compatible and are on the same central schedule d in this implementation. Similar to the pair of contiguous itineraries, for the pair of compatible itineraries (i, j) we also call i the *predecessor* of j and j the *neighbor* of i. According to the convention on the existence of departure and arrival arcs, the internal itinerary is considered the ancestor of all mandatory itineraries and also the successor of each of these. In the new view, excluding internal itineraries, a schedule is a sequence of consecutive mandatory itineraries, which is essentially the same set of mandatory itineraries in a schedule in the sense of before, so we'll sometimes call it a *compact schedule*. Adding an internal itinerary to a collapsed schedule results in a collapsed *cycle*. Thus, in a reduced cycle, every mandatory itinerary has its predecessor and successor.

The cost of making a itinerary j following the itinerary i can be considered as the cost of the associated itinerary on the arc connecting the two itineraries i, j and will be denoted by C_{ij}^d. As mentioned above, it can include the cost of the itinerary i (when we want to transfer the cost from the mandatory itinerary to the no-load itinerary). Then it is $C_{d_0 j}^d$ the $i = d_0$ cost of the departure itinerary. As mentioned, it usually includes both the actual cost on the departure supply and the weight of the vehicle used (to regulate the number of vehicles used at the center under consideration: the larger this weight is, the number of the cars used is less). Now, the goal of minimizing the total deployment cost would be

$$\min \sum_{d \in D} \sum_{i,j \in \tilde{T}_d} C_{ij}^{d} X_{ij}^{d} . \qquad (6)$$

Provided that the variables X_{ij}^{d} can only take one of two values, 0 or 1, we see that the necessary and sufficient condition for a itinerary to $i \in T$ be executed exactly once would be:

$$\sum_{d \in D, j \in \tilde{T}_d} X_{ij}^{d} = 1 \, , \; i \in T \, . \qquad (7)$$

As mentioned above, every mandatory itinerary in a certain (reduced) cycle has an adjacent itinerary and the predecessor in this cycle. In terms of constraint (7), the argument as in Proposition 1, we know this is expressed as follows

$$\sum_{k \in \tilde{T}_d} X_{kj}^{d} - \sum_{k \in \tilde{T}_d} X_{jk}^{d} = 0 \, , \; \forall j \in T_d, \forall d \in D \, . \; (8)$$

Note that, the number of vehicles serving at a center must be equal to the number of departures made, which is equal to $\sum_{k \in A_d^{ht-x}} X_{d_0 k}^{d}$, and must also be equal to the number of arrivals made, i.e. equal $\sum_{k \in A_d^{ht-n}} X_{k d_0}^{d}$ to , so this condition (8) is also satisfied at d_0 , i.e. for all $j \in \tilde{T}_d$.

The constraint on the number of vehicles used at each center is shown as follows:

$$\lambda_d \le \sum_{j \in A_d^{ht-x}} X_{d_0 j}^{d} \le \kappa_d \, , \; \forall d \in D \, . \quad (9)$$

In summary, the problem of finding the optimal schedule set with the given constraints can be described by (6) - (9) in combination with the integer condition 0 or 1 of the decision variable, i.e.

$$X_{ij}^{d} \in \;\; 0,1 \;\; , \quad \forall d \in D, \quad \forall i, j \in \tilde{T}_d \, . \qquad (10)$$

Then, for brevity, we will call it ***Problem*** **(MD1).**

LEMMA 3. *For each acceptable implementation, we have a vector of decision variable values* $X = (X_{ij}^{d})_{d \in D, i, j \in \tilde{T}_d}$ *satisfying the conditions* (7)-(10) . *Conversely, from each decision variable value vector* $X = (X_{ij}^{d})_{d \in D, i, j \in \tilde{T}_d}$ *satisfying the conditions* (7)-(10) *we can establish an acceptable implementation* .

Proof. (Similar to the proof of Lemma 2)

Note. From Lemma 2-3 we see that finding the optimal implementation is referred to as problem solving (MD) or (MD1).

Initial solution

As mentioned above, finding an exact solution to the integer programming problem is not feasible in practice. In a situation like this one can think of one of a number of *hypersensitive solutions that* are increasingly proving effective in practice (see [3],... and the docs therein), or find a way *to simplify problem* (based on practical conditions), to hope to get a good enough solution instead of looking for a theoretically exact optimal solution that is computationally infeasible.

Here, we propose a way to "simplify" the problem, making it easier to solve, and at the same time the resulting solution still has a clear practical meaning.

2.3. Remove constraints on vehicle types

The fact that there are many types of vehicles participating in the operating of the same public transportation network is a reality, and even inevitable. However, adding the set $G(t)$ and its associated constraint sets T_d will make the problem much more complicated. We can "untie" this constraint by reducing the problem with many types of vehicles to the problem of only one type of vehicle, according to the solution below.

A simple yet practical solution is to convert. In this way, we temporarily assume that only *one standard class of vehicles is used* (take the most common among usable vehicles). After solving the problem with this assumption, the manager can use the conversion of vehicles to other existing categories, based on their transportation capacity and based on permissible road conditions on a number of vehicles on specific routes. Thus, on wide roads, standard vehicles will be converted for larger vehicles (actually available in original conditions) and on narrow roads they will be converted for smaller vehicles (actually present in the initial conditions), accompanied by an increase or decrease in the driving frequency to suit the volume of passengers to be served. Usually, the "conversion" is not "equivalent" mathematically, so this solution does not preserve the "theoretical optimality" of the solution. However, this solution is realistical, with the stated goal of finding a viable solution at a significantly improved cost.

Thus, the problem solving with mixed vehicle types can be reduced to solving a number of problems with a single vehicle type. The following will consider only this problem.

2.4. Problem with homogeneous conditions on vehicle types

Under the condition of homogeneity of vehicle types, the cost on the itineraries no longer depends on the technical characteristics at the control center, but only depends on the driving distance, or equivalently converted into time. Cars run on itineraries. Thus, it can be seen that the cost C_{ij}^{d} will no longer depend on d but only on the travel time from the end of the itinerary i to the beginning of the itinerary j, and in a way that moves the cost out of the required itineraries. C_{ij} will include the time the car runs on the itinerary i, denoted by h_i. Thus, the pair of two *mandatory itineraries* i, j would be *compatible* if $s_i + h_i \leq s_j$. The set of such pairs of itineraries will be called the set of compatible itinerary pairs, no longer depending on the operator center, and also the set of associated itineraries between compatible pairs of itineraries (of course we still reserves the right to remove "useless compatible" pairs with a sub-condition as commented above). Since the vehicles are the same, the set T_d is also the same (and coincides with T), and the sets $G(t)$ also no longer depend on t (and are always equal to D). Now, for each control center $d \in D$, we have the following sets:

- $A_d^{ht-lk} = A^{ht-lk}, \quad \forall d \in D$. $A_d^{ht-kt} = A_d^{ht-x} \cup A_d^{ht-n} \cup A^{ht-lk} \cup \ d_0$, $A_d^{ht} = T \cup A_d^{ht-kt}$.

The graph of the center has d the $D_d = \ V_d, A_d$ following $V_d^{'} = \ d^+, d^- \ \cup T^- \cup T^+$ sets of *vertices* and *arcs* , $A_d = A_d^{ht} \cup \ d^-, d^+$.

Now Problem (MD) is rewritten as the following problem

$$(\text{MD}\quad \begin{cases} \sum_{d\in D} \sum_{i,j \in A_d^{ht}} C_{ij} X_{ij}^d \to \min \\ \quad \textit{Constraints} \\ \sum_{d\in D, j\in A_d^{ht}} X_{ij}^d = 1, \forall i \in T, \\ \sum_{i\in A_d^{ht}} X_{ij}^d - \sum_{k\in A_d^{ht}} X_{jk}^d = 0, \quad \forall j \in T_d, \forall d \in D. \\ \lambda_d \leq \sum_{j\in A_d^{ht-x}} X_{d_0 j}^d \leq \kappa_d, \quad \forall d \in D, \\ X_{ij}^d \in \ 0,1 \ , \quad \forall i,j \in A_d^{ht}, \quad \forall d \in D, \end{cases}$$

where, with $i, j \in A_d^{ht}$, the decision variable X_{ij}^d indicates whether the itinerary j is adjacent to the route i and is in the same central schedule d.

Similarly, with the notation $\tilde{T}_d = T \cup \ d_0$, we can rewrite Problem (MD1) as follows

(M

$$\begin{cases} \sum_{d \in D} \sum_{i,j \in \tilde{T}_d} C_{ij} X_{ij}^d \quad \to \quad \min . \\ \textit{Constraints} \\ \sum_{d \in D, j \in \tilde{T}_d} X_{ij}^d = 1, \quad i \in T . \\ \sum_{k \in \tilde{T}_d} X_{kj}^d - \sum_{k \in \tilde{T}_d} X_{jk}^d = 0 \qquad , \\ \forall j \in \tilde{T}_d, \forall d \in D . \\ \lambda_d \le \sum_{j \in A_d^{ht-x}} X_{d_0 j}^d \le \kappa_d, \quad \forall d \in D . \end{cases}$$

where, with $i, j \in \tilde{T}_d$, the decision variable X_{ij}^d indicating whether the required pair of itineraries (i, j) is compatible and executed on the same central schedule d.

The above problem will be solved by the decomposition solution in the next section.

1) Decomposition and iterative algorithms

2) Decomposition solution and initialization plan

We know that unprofitable costs are incurred only on the set of *no-load itineraries*, which are of two types: *associative itineraries* (between required itineraries) and inbound and *outbound itineraries* (which we sometimes call as the *berth-departure itineraries*). We can "simplify" by treating the two costs as less interdependent, and thus can be considered independently. First, we consider the problem of cost minimization on interconnected itineraries, and cost minimization on inbound and outbound itineraries will be discussed later.

While considering the costs on linked itineraries, we temporarily assume that the cost on all entry and exit itineraries is zero. Then, the cost of a travel schedule will not depend on the its departs and berths at any center. That is, we can consider multiple centers as the same one, and so we can use a 1-centre network scheduling algorithm to solve this problem. This problem is much easier than the original problem, which can be solved algorithmically with polynomial time. The result of this process will give us a set of schedules across the network, in which the itineraries are linked together into feasible schedules with minimal "link costs". We call this set of schedules *the initial set of schedules*. With these schedules, the arrival and departure information is meaningless, so we can cut it out and call the rest the *free schedule*. In the next step, we "restore" the information about the actual costs on the input and output arcs and then proceed to optimally allocate the set of free schedules (from the initialization schedule set) to the executive center, according to the algorithm presented in [3]. The output will be a possible alternative, and is called *the initialization plan.*

The initialization method received above is generally not the optimal one, so we can improve it by the following steps.

We consider each operating center with the set of schedules allocated to it (from the initialization scheme) as a *transportation network independent of an operating center*. Since the set of existing schedules (from the initialization plan) is not optimal for this network, we can "optimize" it by "disassemble" these schedules into itineraries independently, extract the set of required itineraries, and then solve the problem of setting up optimal schedules, "covering" these mandatory itineraries, for a network with *one operating center*, according to the presented algorithm. in [4] . The result will be a set of optimal schedules, which is better than the original set of schedules. Since there are all $|D|$operating centers, in this step, we need to solve $|D|$such a problem, and the result will be a set of possible schedules for the entire network. The obtained result is considered an *acceptable solution* .

2.5. The ability to get better through iterative process

2.5.1. Description of the iterative process

In the acceptable solution, the set of schedules of each operating center is generally much different from what was available after the above process. This also means that the set of required itineraries belonging to each operating center has also been changed. That is, if we take out the set of required itineraries included in these schedules, we will have new data for the problem of setting up the optimal schedule set (for each network of each separate center). Thus, with solving this problem, we will get a better solution than the previous one (in general). After solving the above problem (for each operating center), we have a new set of schedules that are better than the accepted solution mentioned. If we cut out the entry and exit routes for each of these schedules, we get a new set of free schedules. After reallocating this set of free schedules to the centers (according to the algorithm in [3]), we will get an even better solution. The above procedure can be repeated many times, and after each iteration the results will be improved. However, as stated at the outset, doing an extra loop only makes sense when the result of the latter is actually significantly better than the previous one (in practical terms). The implementation of multiple iterative process in succession should only be applied when setting up a new network or when carrying out a comprehensive "reform" of the transportation network .

2.5.2. Create a new initialization plan

Obviously, in the above iteration process, the result obtained in the last loop depends a lot on the initialization plan (ie the initialization plan if we set up a new network, or the existing plan if we renovate an existing network). A poorer initial solution can still lead to a better final solution. Therefore, using many different initial alternatives will lead to many different final solutions, and thus we are more likely to choose the best possible alternative. One question is that how to create different initial plan. Our procedure allows to do this quite simply as follows. Let's take any possible solution (possibly the last one), and swap some schedules between operatings centers (so as not to violate constraints, which are not complicated for the operator). in the case of only one type of vehicle). After this swap, we "peeled" out the set of required itineraries from the schedules of each operating center, and then used it as input for the optimal scheduling problem on the center's own network, and this is the new beginning.

2.6. Problem for a traffic network with an operating center

When there is only one operating center, the index d can be eliminated and from Problem (MD*) we have the following problem

$$(S \quad \begin{cases} \min \sum_{i,j \in A^{ht}} C_{ij} X_{ij} \\ \textit{Constraints} \\ \sum_{j \in A^{ht}} X_{ij} = 1, \quad \forall i \in T, \\ \sum_{j \in A^{ht}} X_{ij} - \sum_{k \in A^{ht}} X_{jk} = 0, \quad \forall i \in A^{ht}, \\ \lambda \le \sum_{j \in A^{ht-x}} X_{i_0 j} \le \kappa, \\ X_{ij} \in \; 0,1 \;, \qquad \forall i,j \in A^{ht}, \end{cases}$$

where A^{ht} is the set of all itineraries, A^{ht-x} is the set of departures, is also X_{ij} the decision variable, and takes the value of 1 if and only if the pair of itineraries (i,j) are adjacent and executed on the same schedule.

Similarly, with the notation $\tilde{T} = T \cup \; d_0 \;$, from the problem (MD1*) we have the following problem:

$$(S \quad \begin{cases} \sum_{i,j \in \tilde{T}} C_{ij} X_{ij} \quad \rightarrow \quad \min. \\ \textit{Constraints} \\ \sum_{j \in \tilde{T}} X_{ij} = 1, \quad i \in T. \\ \sum_{k \in \tilde{T}} X_{kj} - \sum_{k \in \tilde{T}} X_{jk} = 0, \quad \forall j \in \tilde{T}. \\ \lambda \le \sum_{j \in A^{ht-x}} X_{d_0 j} \le \kappa \quad . \\ X_{ij} \in \; 0,1 \;, \quad \forall i,j \in \tilde{T}. \end{cases}$$

where X_{ij} is the decision variable and takes the value 1 if and only if the pair of (i,j) trips are compatible and executed on the same schedule.

This problem has been completely solved by algorithms with polynomial time, through converting it to an allocation problem, or a network-minimum flow problem, and then solving it by a Hungarian algorithm or an auction algorithm (Auction Algorithm). Details can be found in works [1], [2] and the references therein.

3. CONCLUSION

In this paper, we have analyzed the practical conditions to come up with a solution to partially handle the complex constraint conditions, thereby converting the original problem to a group of problems with simpler constraints (though still in the NP-hard class). To solve each of these problems, we will use the combined solution of the two processes of iteration and decay. The decomposition aims to convert the original problem into two solvable (in polynomial time) optimal problems. During the iteration, at each step we get an improvement in the objective function. Therefore, it is possible to stop the algorithm at any time to get an acceptable solution that is better than the original starting one. Along with the current strong development of information technology, we can solve the problem of public transportation for Hanoi and other cities.

REFERENCES

1. AHUJA RK, MAGNANTI TL, ORLIN JB (1993), *Network Flows: Theory, Algorithms and Applications* , Prentice-Hall, Inc., Englewood Cliffs, New Jersey.
2. BIANCO L., MINGOZZI A., RICCIARDELLI S. (1994), *A set partitioning approach to the multiple-depot vehicle scheduling problem* , Optimization: Methods and Software, 7 (1994), pp. 163-194.
3. DELL'AMICO M., FISCHETTI M., TOTH P. (1993), “Heuristic algorithms for multiple depot vehicle scheduling problem*”, Management Science*, N1, Vol. 39 (1993), pp. 115-125.
4. FRELING R., WAGELMANS A., PAIXÃO J., “Models and Algorithms for Single-Depot Vehicle Scheduling”, *Transportation Science*, Vol. 35, No. 2, 2001, pp. 165–180.
5. Pham Xuan Hinh (2021), “An approach to the solution of the Hanoi tracffic problem”, *Scientific Journal of Hanoi Metropolitan University*, No. 46, 2021, pp. 5-10.

VỀ MỘT LỜI GIẢI BÀI TOÁN MẠNG GIAO THÔNG CÔNG CỘNG CỦA THÀNH PHỐ VỚI NHIỀU TRUNG TÂM ĐIỀU HÀNH

Tóm tắt: *Bài toán mạng giao thông công cộng của thành phố với nhiều trung tâm điều hành là một bài toán quy hoạch nguyên, nếu số biến đủ lớn thì nó là một bài toán NP- khó. Việc tìm lời giải chính xác cho bài toán này trong thực tế là không khả thi. Trong bài báo này chúng tôi đề xuất một lời giải cho bài toán mạng giao thông công cộng của thành phố với nhiều trung tâm điều hành bằng cách "đơn giản hóa" bài toán theo từng bước, sau mỗi bước bài toán trở nên dễ giải hơn. Cuối cùng đưa bài toán về nhóm các bài toán tối ưu có lời giải trong thời gian đa thức. Từ đó có thể áp dụng để giải quyết bài toán cho mạng giao thông công cộng của thành phố Hà nội và các thành phố khác.*

Từ khoá: *Lịch trình giao thông, giao thông công cộng, Hà Nội.*