PERGAMON

Transportation Research Part A 35 (2001) 913–928

TRANSPORTATION RESEARCH PART A

www.elsevier.com/locate/tra

Creating bus timetables with maximal synchronization

A. Ceder [a,*], B. Golany [b], O. Tal [b]

[a] *Faculty of Civil Engineering, Technion-Israel Institute of Technology, Haifa 32000, Israel*
[b] *Faculty of Industrial Engineering and Management, Technion-Israel Institute of Technology, Haifa 32000, Israel*

Received 26 May 1999; accepted 1 July 2000

Abstract

This paper addresses the problem of generating a timetable for a given network of buses so as to maximize their synchronization. It attempts to maximize the number of simultaneous bus arrivals at the connection (transfer) nodes of the network. Transit schedulers, taking into account the satisfaction and convenience of the system's users, appreciate the importance of creating a timetable with maximal synchronization, which enables the transfer of passengers from one route to another with minimum waiting time at the transfer nodes. In this paper, the problem is formulated as a mixed integer linear programming problem, and a heuristic algorithm is developed to solve the problem in polynomial time. The efficiency of this algorithm, compared to optimal solutions, is illustrated through a series of examples.

Keywords: Public transportation planning; Scheduling; Synchronization; Timetables

1. Introduction and the scope

One of the major roles of transit schedulers, Ceder and Wilson (1986), is to create timetables for the bus routes of a given bus network. According to Ceder (1986), there are three levels of decision problems that have to be addressed prior to the actual scheduling when constructing such timetables:

1. selecting the type of headway (even or uneven headways);
2. selecting a method for setting the frequencies (maximum load or load profile);
3. selecting one or more objective functions according to which the scheduling will be implemented (for example, minimizing operator cost while providing adequate service, minimizing operator and user cost through weighing factors).

* Corresponding author. Tel.: +972-4-8331923; fax: +972-4-8335104.
E-mail address: ceder@tx.technion.ac.il (A. Ceder).

0965-8564/01/$ - see front matter
PII: S0965-8564(00)00032-X

Usually a global approach from the user's perspective considers the minimization of travel and waiting (and possibly walking) times. It is a transit network design approach, which accounted for origin–destination (O–D) data. This paper, however, assumes an existing transit network of routes with certain passenger demand by time of day and focusing on maximal synchronization. This maximal synchronization is a rather important objective from both the operator's and the user's perspectives, involving as it does creating timetables that will maximize the number of simultaneous arrivals of buses at the connection (transfer) nodes. Ceder and Wilson (1986) in a study of transit route design at the network level emphasized the importance of eliminating a large number of transfer points, due to their adverse effect on the user. No doubt, there is a trade-off between this elimination and the efficiency of the bus route network from the operating cost perspective. In order to allow for an adequate bus level-of-service, the schedulers face the synchronization task to ensure maximal smooth transfers involving switching passengers from one route to another without waiting time. This task is extensive, minimizing also the waiting time for those passengers who require connections. By doing so, the scheduler creates a more attractive transit system that generates the opportunity for increasing the number of riders. This is presented schematically in Fig. 1.

Actually, synchronization is the most difficult task of transit schedulers, and is currently addressed intuitively. The scheduler, in fact, attempts to create the departure times in the timetable while complying with (i) the required frequency; (ii) efficient assignment of trips to a single bus chain, and (iii) synchronization of certain arrivals. This paper presents an effective mathematical procedure for maximal synchronization to be employed as a useful tool for the scheduler in the process of creating timetables.

The problem addressed in this paper has not been dealt with extensively in the literature. Voss (1992) formulated the problem of minimizing the waiting time of passengers at the transfer nodes as a quadratic assignment problem (QAP) as it is explained by Lawler (1963) and Hillier and Connors (1966). His study refers to the cases where each bus route, i, is jointed by a set, $n(i)$, of possible departure times. The problem was modified to a case where different routes partly use the same tracks, implying that security distances must be observed. Desilet and Rousseau (1992) describe a different model, which selects a starting time for each route from a set of possible starting times, T. The objective function is to minimize the total penalty associated with transfers from line i to line j, for each i, j. The penalty function, which can be calculated in various ways, takes into account the random nature of traveling times.

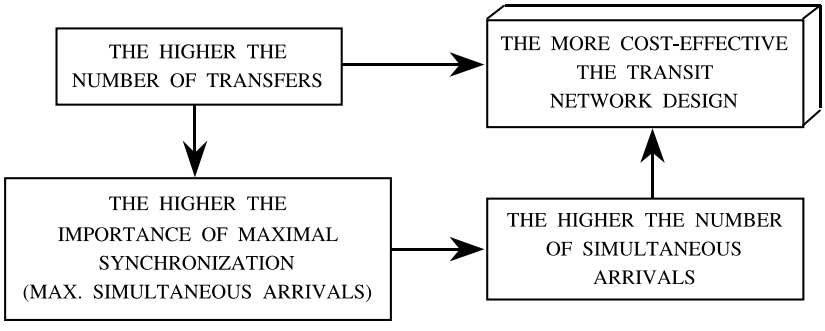

Fig. 1. The relative importance of synchronization.

Daganzo (1990) presents the problem of the coordination of a network comprising only one node, at which inbound and outbound routes intersect. In addition, Lee and Schonfeld (1991) are attempting to synchronize one bus route with a rail line while assuming stochastic conditions. Their conclusion is that there is no justification for synchronization for situations characterized by highly variable arrival times. Finally, Chin and Schonfeld (1998) are trying to optimize the overall cost while integrating the schedule of a rail line and its feeding buses, and also showing the complexity of their problem.

These few articles are not looking at the synchronization problems through the scheduler's practical perspective. The purpose of this paper, though treating the scheduler's problem in a mathematical fashion, is to establish a useful scheduler's tool for synchronization. That is to provide recommendations on how to change the transit timetable while assuring (for a certain sets of routes) maximal synchronization.

This paper presents a different model, which enables transit schedulers to set restrictions on the headways for each route, to introduce different frequencies for every route, and to apply other constraints. The objective function is to maximize the number of simultaneous bus arrivals in the network. The departure times are set in such a way as to achieve this goal. It is worth mentioning that the model presented is also capable, with a small extension, to assign different weights on each different simultaneous arrival (two different lines or time periods). That is, if the scheduler wishes to provide different importance levels for each synchronization situation, he may then introduce the weights, and the objective function will change to maximize the sum of all weights.

Section 2 of this paper provides two mathematical formulations of the problem – a nonlinear programming and a mixed integer linear programming. Section 3 presents a heuristic algorithm that solves the problem. Examples in which the heuristic algorithm is compared to optimal solutions are presented in Section 4, and conclusions Section 6.

2. Formulating the model

The given bus network is presented by a directed graph, $G = \{A, \bar{N}\}$, where A is a set of arcs representing the traveling path of the bus routes; $\bar{N}$ is a set of transfer nodes in the network. The problem data are the following: T is the planning horizon (the departure times of the buses can be set in the interval [0, T] which is a discrete interval); M the number of bus routes in the network; N the number of transfer nodes in the network; Hmin_k the minimum headway (operator's requirements) between two adjacent bus departures in route k $(1 \leqslant k \leqslant M)$; Hmax_k the maximal headway (policy headway) permitted between two adjacent bus departures in route k $(1 \leqslant k \leqslant M)$; F_k the number of departures to be scheduled for route k during the interval [0, T] $(1 \leqslant k \leqslant M)$; T_{kj} is the traveling time from the starting point of route k to node j $(1 \leqslant k \leqslant M,$ $1 \leqslant j \leqslant N)$ (traveling times are considered deterministic, and can be referred to as the mean traveling times).

The following should be noted: (a) this paper assumes throughout that the first departure in each route k must take place in the interval [0, Hmax_k]; (b) the problem is impractical unless the following constraints hold, for each k:

$$\mathrm{Hmax}_k \geqslant \mathrm{Hmin}_k, \tag{1}$$

$$T \geqslant (F_k - 1) \cdot \mathrm{Hmin}_k, \tag{2}$$

$$T < F_k \cdot \mathrm{Hmax}_k, \tag{3}$$

(c) the case where a route k does not pass through a node j is represented by $T_{kj} = -1$.

The decision variables are the following:

(a) X_{ik} represents the departure time of the ith bus in route k $(1 \leqslant i \leqslant F_k)$;

(b) Z_{ikjqn} is a binary variable that yields the value 1 if the ith bus in route k meets the jth bus of route q at node n; otherwise, it yields the value 0.

Let

$$A_{kq} = \{n : 1 \leqslant n \leqslant N,\ T_{kn} \geqslant 0,\ T_{qn} \geqslant 0\}.$$

The initial formulation of the problem is as follows:

$$\max \sum_{k=1}^{M-1} \sum_{i=1}^{F_k} \sum_{q=k+1}^{M} \sum_{j=1}^{F_q} \sum_{n \varepsilon A_{kq}} Z_{ikjqn}$$

s.t.

$$X_{1k} \leqslant \mathrm{Hmax}_k, \quad 1 \leqslant k \leqslant M, \tag{4}$$

$$X_{F_k k} \leqslant T, \quad 1 \leqslant k \leqslant M, \tag{5}$$

$$\mathrm{Hmin}_k \leqslant X_{(i+1)k} - X_{ik} \leqslant \mathrm{Hmax}_k, \quad 1 \leqslant k \leqslant M,\ 1 \leqslant i \leqslant F_k - 1, \tag{6}$$

$$Z_{ikjqn} = \max[1 - |(X_{ik} + T_{kn}) - (X_{jq} + T_{qn})|, 0]. \tag{7}$$

Constraint (4) ensures that the first departure time will not be beyond the maximal headway from the start of the time horizon, while constraint (5) ensures that the last departure is within the planning horizon. Constraint (6) indicates the headway limits, and constraint (7) defines the binary variable of the objective function.

This model can be simplified by defining a variable, Y_{kq}, representing the overall number of simultaneous arrivals of buses in route k with buses in route q. The model is changed as follows:

$$\max \sum_{k-1}^{M-1} \sum_{q=k+1}^{M} Y_{kq}$$

s.t.

$$Y_{kq} = \sum_{n \varepsilon A_{kq}} \sum_{i=1}^{F_k} \sum_{j=1}^{F_q} \max[1 - |(X_{ik} + T_{kn}) - (X_{jq} + T_{qn})|, 0]. \tag{8}$$

Constraints (4)–(7) remain unchanged.

The last formulation represents a nonlinear programming problem. It can be reformulated as a mixed integer linear programming problem, which can be solved (up to certain sizes), by several software packages. The nonlinear constraint is (8). Let D_{nijkq} denote a binary variable (defined over the same domain as Z_{ikjqn}), and B denote a large number $(B = T + \max_{i,j} T_{ij})$.

Constraint (8) is exchanged with the following constraints:

$$B \cdot D_{nijkq} \geqslant X_{ik} + T_{kn} - (X_{jq} + T_{qn}), \tag{9}$$

$$B \cdot D_{nijkq} \geqslant X_{jq} + T_{qn} - (X_{ik} + T_{kn}), \tag{10}$$

$$Y_{kq} < \sum_{n \varepsilon A_{kq}} \sum_{i=1}^{F_k} \sum_{j=1}^{F_q} \left(1 - D_{nijkq}\right). \tag{11}$$

If $X_{ik} + T_{kn} = X_{jq} + T_{qn}$, then there is a simultaneous arrival of the ith bus in route k with the jth bus of route q at node n. The variable D_{nijkq} can yield the value 0, and Y_{kq} is increased by one, according to (11).

If $X_{ik} + T_{kn} \neq X_{jq} + T_{qn}$, then the arrivals do not coincide, and D_{nijkq} must yield the value in order to satisfy constraints (9) and (10). The number of simultaneous arrivals between buses of routes k and q (Y_{kq}) is not increased in (11).

An upper bound on the number of possible simultaneous arrivals in a given bus network is

$$Z^* = \sum_{k=1}^{M-1} \sum_{q=k+1}^{M} \sum_{n \varepsilon A k q} \min(F_k, F_q).$$

The number of integer variables in an MIP problem is generally a good index of its complexity. The variable D_{nijkq} represents the simultaneous arrival of the ith bus of route k and the jth bus of route q at node n. This means an integer variable for every combination of two buses on different routes that intersect at node n. Let F be $\max(F_k)$; the number of integer variables in the worst case is $0(NM^2F^2)$, which is a very large number. However, in a more realistic setting, this number is $0(M^2F^2)$, where N can be replaced by the average number of nodes shared in common by any two routes (normally 1 or 2).

3. Heuristic algorithm

In the previous section, the problem was formulated as a large MIP problem. Running small network examples (5 routes, 5 nodes), using any mixed integer LP programming software required hours and even days of running time! This motivated the development of a heuristic algorithm that would solve such problems in a reasonable time. This section will present a heuristic algorithm and provides some remarks about a possible procedure that can be added to it. The algorithm is implemented in Turbo-Pascal (1989), and many examples were checked and compared to the optimal solutions obtained by using an LP programming software. The algorithm is based on the selection of nodes within the network. In each step, the next node is selected, providing that in this node, not all the departure times have yet been determined. Once the departure time is resolved, all its corresponding arrival times are set too.

3.1. Algorithm components

Definition 1. A node is defined as "possible" if the following holds:

(a) There is at least one bus route that passes through the node, and not all the departure times for that route are set;

(b) It is possible to create more simultaneous arrivals at the node.

Definition 2. A node is defined as "new" if no arrival times have been set for it.

The algorithm uses several procedures as described in flow chart form in Fig. 2.

Step 1: Initialization: check whether the problem is feasible, and create the data structures. Mark all nodes as possible;

Step 2: Select the next node, NO, from the possible nodes;

Step 3: If NO is new, perform procedure FIRST, otherwise perform procedure MIDDLE;

Step 4: If there is any possible node, go to Step 2; if there are any more routes, perform procedure CHOOSE and go to Step 2, otherwise stop.

Step 1 contains a check of whether the problem is feasible (constraints (1)–(3)), and two data structures are built:

(a) A structure called route for each bus route i, which includes Hmin_i, Hmax_i, F_i, the number of nodes the route passes through, and the departure times that have already been set;

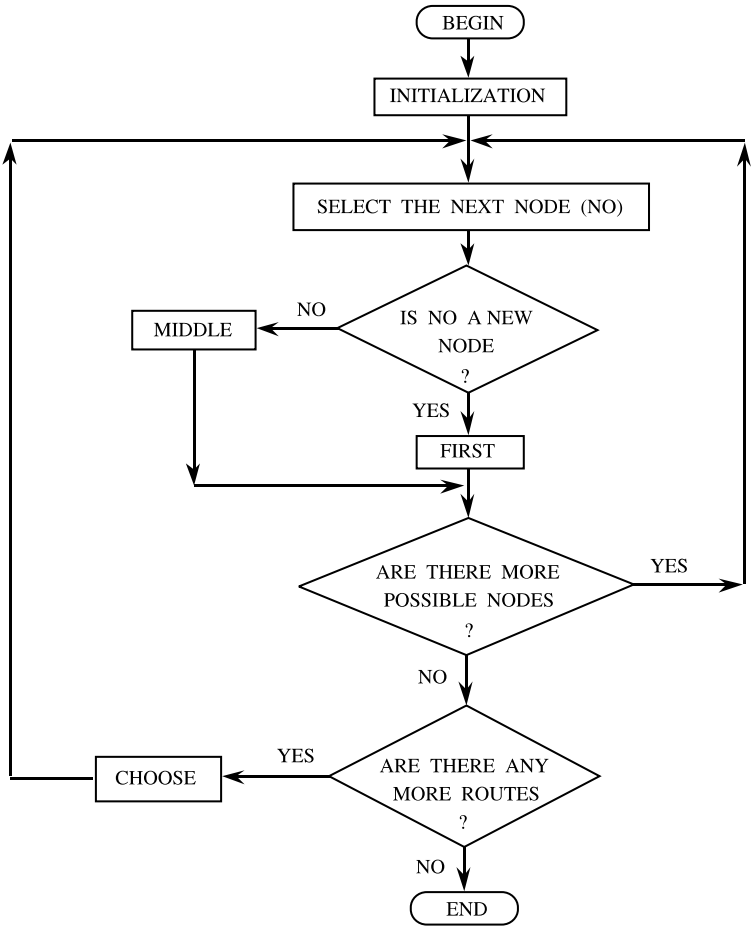

Fig. 2. A flow-chart of basic algorithm.

(b) A structure called node for each node n, which includes the number of routes passing through the node; the route with the maximum traveling time to the node, and the number of simultaneous arrivals at the node at each time point in interval $[0, T + \max_{i,j} T_{ij}]$.

All the nodes are marked possible.

In Step 2, the next node, NO, is selected from among the possible nodes. Node N must satisfy the following:

(a) The number of different bus arrival times at the node is maximal; in such a node, there is a greater probability that another bus departure can be set so that it will arrive at NO by any one of the (already set) arrival times;

(b) Among the nodes satisfying (a), NO is that through which a maximal number of routes pass; in such a node, there is a greater potential for simultaneous arrivals;

(c) Among the nodes that satisfy (b), NO minimizes the maximum travel time of all routes from their origin to the node (after the departure times of buses are set in order to meet at NO, there is still a potential for simultaneous arrivals at distant nodes).

A distinction is made in Step 3 between a new and another node. For a new node procedure, FIRST attempts to set the departure times of buses that pass through it, such that the buses will arrive at the node at the earliest time possible, and simultaneous arrivals will continue to be created at the node according to the Hmin, Hmax of the routes. For example, buses on routes 2 and 3 that arrive simultaneously at a certain node at time t_0: if the next departure time for each of these routes can be set at a fixed difference, d, from the last departure time of the route (for $i = 1, 2, 3$ $\max \text{Hmin}_i \leqslant d \leqslant \min \text{Hmax}_i$), then there will be additional simultaneous arrivals at the node at time $t_0 + d$. Procedure FIRST finds the minimal d possible. If parameter d cannot be set (a situation where $\text{Hmin}_i > \text{Hmax}_j$), then the next departure times of buses on these routes will not be resolved at this step.

For a node that is not new, there is an attempt to set the departure times of buses on routes passing through it, such that the buses will arrive at the node at the earliest time among all the arrival times already set in that node. If no more simultaneous arrivals are available at the node, the node is marked "not possible". This procedure is called MIDDLE.

Step 4 contains a test of whether there are any more possible nodes. If not, there may be routes on which not all the departure times of the buses were set. In such cases, the route which passes through the maximum number of nodes is chosen, and its next departure time is set by using the difference Hmin_i from the last departure. In such a manner, the algorithm sets additional bus arrivals for the maximum number of nodes possible. All the nodes through which route i passes are marked possible, and the algorithm returns to step 2. This procedure is called CHOOSE. The complexity of the algorithm is, in the worst case, $0(NTFM^2)$.

A possible variation of this algorithm is to run the algorithm N times, each time choosing a different node to be the first handled (the first time the node is chosen in the basic algorithm, it is selected according to the principles explained in Step 2; there are examples where this selection is unsuccessful). The complexity of this variation is, in the worst case, $0(TFN^2M^2)$.

3.2. A possible additional procedure

It is certain that the results could be improved by allowing the timetable obtained by the algorithm to be "shifted". That is, for each node and for each two time points: t_1, t_2 $(t_1 < t_2)$ at

which there are bus arrivals at the node, there is an attempt to "shift" all the bus departure times arriving at the node at t_1 so that they will arrive at time t_2. If this succeeds, the timetable is changed accordingly, and the number of simultaneous arrivals increases. It should be noted that in order to shift a single bus departure time, the following must be checked:

(a) After the "shifting", the constraints on Hmin and Hmax must still hold. Otherwise, there should be shifting of additional departure times on the route;

(b) As a result of shifting the departure time of bus i, its arrival time for all the nodes passed through is changed. Therefore, the departure time of each bus that arrives simultaneously with bus i at any node must also be shifted. For each such shifting, there is again a need to check the constraints on Hmin and Hmax, and so on. This procedure is recursive, and an attempt to shift a single departure time may cause the shifting of all the departure times of the network. The complexity of the algorithm with this shifting procedure is, in the worst case, $0(M^2N^3F^2T^3)$.

The complete explanation of the shifting procedure appears in a research report by Tal et al. (1991).

4. Examples

This section describes three examples of bus networks. The first two examples are presented in detail for the sake of clarity. All the three examples are same optimal and heuristic solutions.

4.1. Detailed examples

Fig. 3 presents a simple network example combining two transfer points with two routes. The numbers on the arcs are travel times (say, in minutes). What follows is a demonstration of the heuristic algorithm.

Step 1: Structure (a)

i	Hmin_i	Hmax_i	F_i	No. of nodes
I	5	15	4	2
II	8	20	3	2

Structure (b):

Node n	No. of routes	Route with max T_{in}
1	2	II
2	2	II

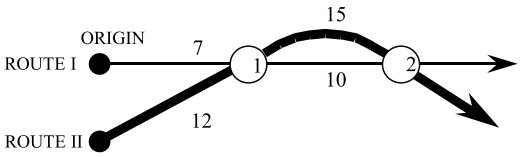

Fig. 3. The basic network for example 1.

Step 2: Select NO with maximum arrival time (1st criterion), maximum routes crossing (2nd), and min–max travel time from origin to NO (3rd). Thus, number of departure (arrival) times ($=0$) for both nodes.

- Number of crossing routes ($=2$) for both nodes.
- Maximum travel time for node 2: maximum $(17, 27) = 27$ and for node 1: maximum $(7, 12) = 12$.
- Min $(12, 27) = 12$.
- Selected NO is, therefore, node 1.

Step 3: Setting earliest time possible for node 1, and continuing in this node the synchronization based on: $\max(5, 8) \leqslant d \leqslant \min(15, 20) \rightarrow d_{\text{minimum}} = 8$. This is the procedure FIRST which results in

Departure time, route I	Departure time, route II	Meeting time
5	0	12
13	8	20
21	16	28

where meeting refers to as simultaneous arrival.

The number of departures of route II reaches their $F_2 = 3$.

Step 4: Since node 2 is still untreated – go to step 2 and select it.

Step 3: Since node 2 is not new (see Definition 2), procedure MIDDLE is applied. $F_1 = 4 > 3$ (currently created); hence, one more departure time of route I is set based on the already created departure times of route II (at 0, 8, 16), such that the synchronization is made at node 2. The results are the additional departure time for route I (at 26).

Step 4: No more routes, and the algorithm ends with the final results:

Departure time		Meeting time at node 1	Meeting time at node 2	Total no. meetings
Route I	Route II			
5	0	12		4
13	8	20		
21	16	28		
26			43	

In this simple example, the optimal (via an LP software) and heuristic procedures coincide.

Fig. 4 presents a more complex example, combining four nodes with four routes.

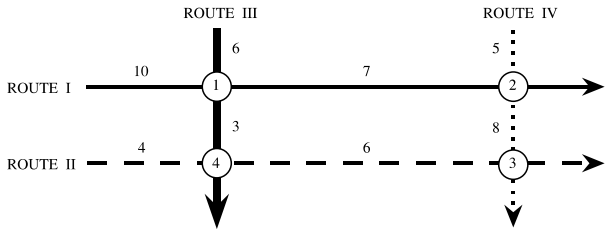

Fig. 4. The basic network for example 2.

The data for example 2 are as follows:

i	Hmin_i	Hmax_i	F_i	T
Route I	8	15	2	30
Route II	10	15	3	
Route III	10	15	3	
Route IV	14	20	2	

Step 1: In structure (a), the number of nodes each route passes through is two.
Structure (b):

Node n	No. routes	Route with maximum T_{in}
1	2	I
2	2	I
3	2	IV
4	2	III

Step 2: Select NO

- Number of departure (arrival) times ($=0$) for all nodes.
- Number of crossing routes ($=2$) for all nodes.
- Maximum travel times for the nodes are 10, 17, 13, 9, respectively.
- The minimum is 9, so NO $=4$.

Step 3: Procedure FIRST. The first meeting time possible at node 4 is nine. The parameter d is 10 (max (10, 10) $\leqslant d \leqslant$ min (15, 15)).
Procedure FIRST results:

Departure time, route II	Departure time, route III	Meeting time
5	0	9
15	10	19
25	20	29

Step 4: There are more possible nodes – go to Step 2.
Step 2: Select NO.
In the previous steps, the algorithm sets three arrival times for nodes 1 and 3. The number of routes passing through these nodes is the same; therefore, NO $=1$, by the min–max criteria.
Step 3: Procedure MIDDLE is performed.
The bus arrival times (route III) at node 1, as set by procedure FIRST, are 6, 16, 26. Procedure MIDDLE sets the departure time of route I buses to arrive at node 1, and meet route III buses as early as possible. Procedure MIDDLE results in the following:

Departure time, route I	Meeting time, node 1
6	16
16	26

Step 4: There are more possible nodes – go to Step 2.

Step 2: Select NO.

- The number of bus arrivals at node 2 is two (route I buses).
- The number of bus arrivals at node 3 is three (route II buses).
- Hence, NO = 3.

Step 3: Procedure MIDDLE is performed.

This procedure sets the bus departure time of route IV buses to meet the route II buses at node 3. The arrival times of node 3 are 15, 25, 35. The procedure yields two more meetings by the following:

Departure time, route IV	Meeting time, node 3
2	15
22	35

Step 4: No more nodes or routes. Stop!

The algorithm ends with this timetable:

Departure time			
Route I	II	III	IV
6	5	0	2
16	15	10	22
	25	20	

The total number of meetings is seven (which is the optimal solution for this network), as follows:

Node	Time of meeting
4	9
4	19
4	29
1	16
1	26
3	15
3	35

Fig. 5 presents the third example with a case of $\mathrm{Hmin}_i > \mathrm{Hmax}_j$ for a 2-route 4-node network.

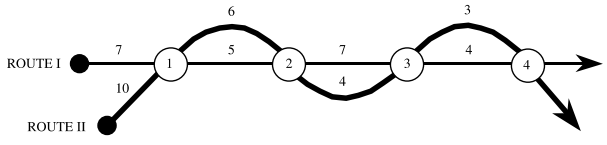

Fig. 5. The basic network for example 3.

The data for example 3 are as follows

i	Hmin_i	Hmax_i	F_i	T
Route I	6	10	4	30
Route II	3	5	6	

with $\text{Hmin}_1 > \text{Hmax}_2$.

Step 1: In structure (a), the number of nodes each route passes through is four. Structure (b)

Node n	No. routes	Route with max T_{in}
1	2	II
2	2	II
3	2	II
4	2	I, II

Step 2: Select NC:

- Number of departure (arrival) times ($=0$) for all nodes.
- Number of crossing routes ($=2$) for all nodes.
- Maximum travel times for the nodes are 10, 16, 20, 23, respectively.
- The minimum is 10, so NO $=1$.

Step 3: Procedure FIRST.

The first meeting time possible at node 1 is 10. The procedure cannot set the parameter d. Therefore, procedure FIRST results in only the first departure time of each route, as follows:

Departure time, route I	Departure time, route II	Meeting time
3	0	10

Step 4: There are more possible nodes – go to step 2.

Step 2: Number of arrival times is 1, 2, 2, 2, respectively, for nodes 1, 2, 3, 4. NO $=2$ (maximum travel time, 16, is minimum).

Step 3: Procedure MIDDLE. The procedure fails to create meetings at node 2.

Steps 2, 3: NO $=3$. There are no meetings possible. NO $=4$. Procedure MIDDLE sets a new meeting at node 4, at time 26, by setting the second departure time of Route II to 3.

The algorithm continues to perform procedure MIDDLE until it ends with the following (also optimal) results:

Departure time		Meeting time at node 1	Meeting time at node 2	Total no. meetings
Route I	Route II			
3	0	10		6
9	3	16		
15	6	22		
21	9		26	
	12		32	
	15		38	

5. Real-life example

The fourth example was taken from a real-world synchronization problem in Israel. The network described in Fig. 6 comprises a main road that has three major transfer nodes at which passengers can transfer from different routes. There are seven bus routes traveling in each direction, or 14 bus routes altogether. The traveling direction of each route is shown in Fig. 6.

The traveling times of routes 1–7 which travel 'downwards' are shown in Table 1.

A value of −1 in the table indicates that the route does not pass through the node. The traveling times of the opposite routes (8–14) are identical for all routes:

The traveling time to node 3 is 24 min; from node 3 to node 2 – 1 min; and from node 2 to 1 – 1 min.

The time interval is 2 h and 54 min (between 9:00 and 11:54) and $F_i = 12$ for each i. The headway limits, Hmin_i and Hmax_i were set to be 14 and 20 min, respectively, for each i.

The matrix T_{kj} was changed so that unnecessary simultaneous arrivals – namely, bus routes in opposite directions that intersect at one node, or two routes that share the same track and intersect at a common node – will not be taken into account. Similarly, for routes 12, 13, and 14, which have exactly the same track, different departure times were set, albeit with equal headways.

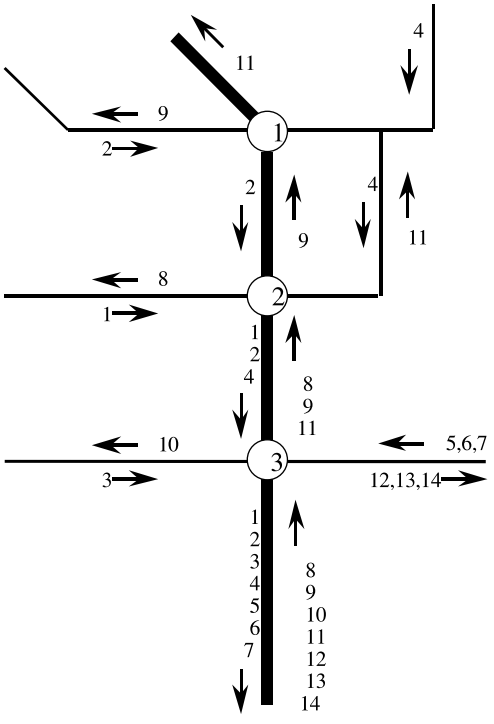

Fig. 6. A real-life synchronization problem.

Table 1
The traveling time (in minutes) of routes 1–7 (for example 4)

Route no.	Node no.		
	1	2	3
1	−1	12	14
2	5	6	8
3	−1	−1	18
4	−1	16	18
5	−1	−1	15
6	−1	−1	19
7	−1	−1	19

Table 2
Departure times of buses 1–7 for example 4

Line 1	9:13	9:27	9:41	9:55	10:09	10:23	10:37
	10:51	11:05	11:19	11:33	11:47		
Line 2	9:19	9:33	9:47	10:01	10:15	10:29	10:43
	10:57	11:11	11:25	11:39	11:53		
Line 3	9:06	9:20	9:34	9:48	10:02	10:16	10:30
	10:44	10:58	11:12	11:26	11:40		
Line 4	9:09	9:23	9:37	9:51	10:05	10:19	10:33
	10:47	11:01	11:15	11:29	11:43		
Line 5	9:09	9:23	9:37	9:51	10:05	10:19	10:33
	10:47	11:01	11:15	11:29	11:43		
Line 6	9:05	9:19	9:33	9:47	10:01	10:15	10:29
	10:43	10:57	11:11	11:25	11:39		
Line 7	9:12	9:26	9:40	9:54	10:08	10:22	10:36
	10:50	11:04	11:18	11:32	11:46		

Table 3
Departure times of buses 8–14 for example 4

Line 8	9:00	9:14	9:28	9:42	9:56	10:10	10:24
	10:38	10:52	11:06	11:20	11:34		
Line 9	9:00	9:14	9:28	9:42	9:56	10:10	10:24
	10:38	10:52	11:06	11:20	11:34		
Line 10	9:00	9:14	9:28	9:42	9:56	10:10	10:24
	10:38	10:52	11:06	11:20	11:34		
Line 11	9:00	9:14	9:28	9:42	9:56	10:10	10:24
	10:38	10:52	11:06	11:20	11:34		
Line 12	9:03	9:17	9:31	9:45	9:59	10:13	10:27
	10:41	10:55	11:09	11:23	11:37		
Line 13	9:07	9:21	9;35	9:49	10:03	10:17	10:31
	10:45	10:59	11:13	11:27	11:41		
Line 14	9:11	9:25	9:39	9:53	10:07	10:21	10:35
	10:49	11:03	11:17	11:31	11:45		

This example was too ample to be solved on a PC using a mixed integer LP software. The best solution obtained by the heuristic algorithm has 240 bus simultaneous arrivals. The timetables are shown in Tables 2 and 3.

6. Conclusions and extensions

This paper has described the problem of maximal synchronization in creating bus timetables; that is maximizing the number of simultaneous bus arrivals at transfer nodes. The mathematical model described, with deterministic traveling times, was formulated as a mixed integer linear programming problem, which is known to be an NP problem. For large networks, it is not practical to solve the problem by using software for MIP problems, and that was the motivation for developing a heuristic algorithm.

The heuristic algorithm developed is implemented with Turbo-Pascal programs, and achieved very good results compared to the optimal solution. One of the examples described in the paper is a real synchronization problem taken from an Israeli bus company. The method developed has important applications for the transit schedulers who are facing the synchronization problem frequently while creating the timetables. Often the schedulers are ready to "sacrifice" a certain pattern of a timetable (e.g., even headways) in order to connect (synchronize) two or more trips from different routes. These connection possibilities are being provided by this work and this knowledge of connections is also suitable for the on-going development of real-time passenger information systems.

The model described can be extended in various ways; for example, defining weights for each node or for a simultaneous arrival between each two bus routes (describing situations in which there are major nodes, or differences in the importance of bus routes). A relatively easy extension to the problem presented is to assign weights to each simultaneous arrival (weights for different lines, time periods, direction of travel, etc.). The objective function will be then to maximize the sum of all weights. Another option is to define a "simultaneous arrival" in different ways. For example, a simultaneous arrival can also be defined as an arrival of two buses within an interval of t minutes.

References

Borland International, 1989. Turbo Pascal version 5.0.

Ceder, A., 1986. Methods for creating bus timetables. Transportation Research A 21, 59–83.

Ceder, A., Wilson, N.H.M., 1986. Bus network design. Transportation Research B 20, 331–344.

Chin, C., Schonfeld, P., 1998. Joint optimization of a rail transit line and its feeder bus system. Journal of Advanced Transportation 32 (3).

Daganzo, C.F., 1990. On the coordination of inbound and outbound schedules at transportation terminals. In: Proceedings of the 11th International Symposium on Transportation and Traffic Theory, Yokohama, Japan, Elsevier, New York, 1990, pp. 379–390.

Desilet, A., Rousseau, J., 1992. Syncro: a computer-assisted tool for the synchronization of transfer in public transit networks. In: Desrochers, M., Rousseau, J.M. (Eds.), Computer-Aided Transit Scheduling. Springer, Berlin, pp. 153–166.

Hillier, F.S., Connors, M.M., 1966. Quadratic assignment problem algorithms and the location of indivisible facilities. Management Science 13 (1), 42–57.

Lawler, E.L., 1963. The quadratic assignment problem. Management Science 9 (4), 586–599.

Lee, M., Schonfeld, P., 1991. Optimal slack time for timed transferred at transit terminal. Journal of Advanced Transportation 25 (3).

Tal, O., Ceder, A., Golany, B., 1991. Maximal synchronization in planning bus timetables. Research Report 91-169, Transportation Research Institute, Technion-Israel Institute of Technology.

Voss, S., 1992. Network design formulation in schedule synchronization. In: Desrochers, Rousseau (Eds.), Computer-Aided Transit Scheduling. Springer, Berlin, pp. 137–152.