Proceedings of the 8th International Conference on Applied Informatics
Eger, Hungary, January 27–30, 2010. Vol. 2. pp. 247–254.

Efficient Estimation of Loads in Service Networks*

László Pál[a], Tibor Csendes[b]

[a]Faculty of Business and Humanities, Sapientia, Hungarian University of Transylvania, Miercurea-Ciuc, Romania

[b]Institute of Informatics, University of Szeged, Hungary

Abstract

Nowadays, the estimation of the load of service networks becomes more and more important. Service networks comprise here postal, transportational and communication networks. The cost of the measurement depends both on the place in the network, and on the amount to be measured. For example, for postal services the automatization provides an efficient facility for counting the letters. Obviously, there are other, cheaper and less precise measurement methods. Hence, there are different ways to estimate the daily loads of the deliverers. In such a network the nodes mark the operations of a flow process, the edges represent the flow directions.

The problem is how to plan the traffic measurement in the network with minimal cost, if we know the costs of possible measurements at the nodes. In case of a given output node, we are looking for those nodes, which influence the traffic of this output node. We want to ensure a set precision for the output node values in terms of uncertainty intervals. Our aim is to achieve the result with the smallest measurement cost. The network's evaluation is calculated by interval arithmetic. In the paper we consider a solution algorithm for the presented problem and we test it on randomly generated networks.

Keywords: service networks, transportation networks, traffic flow, measurement planning, interval arithmetic

MSC: 65K15, 90B06, 90C30

*This work was supported by the grant TÁMOP-4.2.2/08/1/2008-0008 of the Hungarian National Development Agency.

1. Introduction

The estimation of the load of service networks becomes a more and more important task nowadays. In the past period, substantial research has been completed to develop flow prediction models, which forecast the future of traffic flows (see e.g. [2], [3], [4], [8]). Service networks comprise here postal, transportational, and communication networks. Further service network analysis details can be found in [1]. Like telecommunications, transportation sector, postal service is a network industry.

The postal network has an obvious hierarchical structure. That is, mail entering the network tends to travel through facilities serving larger and larger geographic areas (outward sorting and transportation), until its final destination is contained in a facility's service area. Then it begins its inward journey, when this process is reversed: mail is transported and sorted at facilities serving smaller and smaller geographic areas until it reaches the letter carrier's route. The postal transportation chain can be represented by a directed graph (see Figure 1). In such a network, the nodes mark the operations (collection, sorting, and delivery) of a flow process, while the edges represent the flow directions. Those nodes which contain just outgoing edges, represent the collection places of the network, while the nodes which have only input edges, represent the distribution (or delivery) places.

The non-negative values in the nodes and along the edges represent the traffic data of some product (e.g. the number of letters). The values in collector nodes represent the collected quantities, while the numbers in the distribution nodes denote the delivery amounts. In the case of the rest of the nodes, the crossing product quantity can be read from the circle. After describing the basic elements of the postal network, we describe the considered problem in the next section.

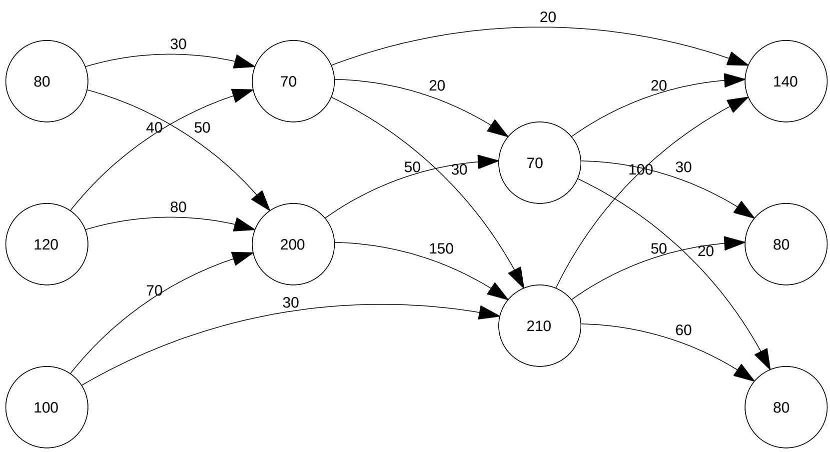


Figure 1: An example of postal network with the basic model elements

2. The problem description

With the mechanization of the post's logistic system, it is possible to estimate easily the daily loads of the deliverers. This automatization provides an efficient facility for counting a large part of the letters.

Our aim is to estimate the load in a node or in several nodes. The basic element of the approximation is the possibility of the measurement in a node. The measurement has its cost, which depends both on the place in the network, and on the amount to be measured. In the postal network we distinguish two types of the nodes: one with automated counting and one with manual counting. The most important difference between them is the cost of the measurement. In the first case, a measurement is obviously cheaper per letter.

The problem considered in this paper is now how to plan the traffic measurement in the network with minimal cost, when we know the cost of measurement in the nodes. In the case of a given output node, we are looking for those nodes, which influence the traffic of that particular output node. We want to ensure a preset precision level for the output node values in terms of uncertainty intervals. The question is, where should we measure in order to reach the given precision for the output interval. Our aim is to achieve the result with the smallest measurement cost. Similar optimization problems in postal networks are formulated in ([6], [7]) considering different objectives like vehicle factors, time limit, frequency, cost, and so on. In the following part, we describe two important techniques which will be used during the problem investigation.

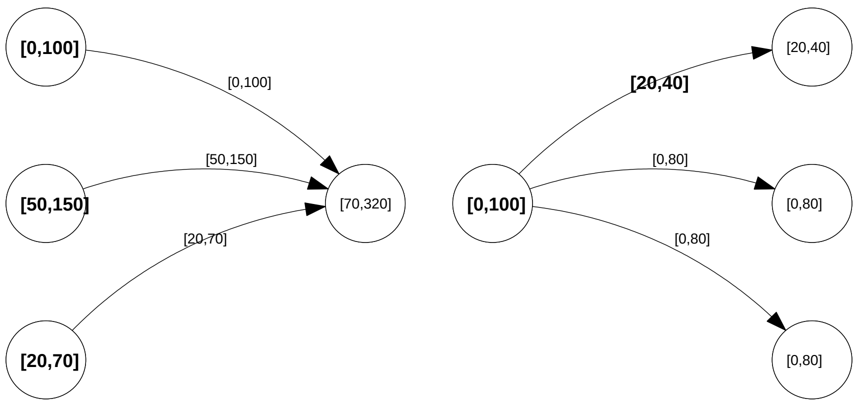
2.1. Network evaluation

An important task of the investigated problem is the network evaluation. Based on the known data in some node, we can evaluate the whole network in order to update the influenced node and edge values. The network evaluation is made with the help of interval calculation [5]. All the data in the nodes and along the edges are represented by intervals. During the network evaluation the basic interval arithmetic operations are used and the calculated values are propagated form the input nodes to the output nodes.

Usually interval arithmetic techniques are used in problems where controlling errors of different kind is important. The evaluation is made by calculating upper and lower bounds on the given values and then developing suitable numerical methods. In the present case, we do not have rounding errors, but we often have uncertain data represented by intervals. We list here the basic interval arithmetic operations for better understanding of the later discussion.

$$
\begin{aligned}
[a, b] + [c, d] &= [a + c, b + d], \\
[a, b] - [c, d] &= [a - d, b - c], \\
[a, b] * \ [c, d] &= [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}], \\
[a, b] \ / \ \ [c, d] &= [a, b] * [1/d, 1/c], \text{if } 0 \notin [c, d].
\end{aligned}
$$

We have included here the multiplication and the division as well, but in the network evaluation only the first two operations are needed. The addition and subtraction operations are illustrated in Figure 2. The corresponding interval arithmetic operations are the following: $[0, 100] + [50, 150] + [20, 70] = [70, 320]$ and $[0, 100] - [20, 40] = [-40, 80]$. In the postal network negative lower bounds does not make sense. That is why we set the 0 as the respective lower bound on the figure. In both cases, the known data in the nodes and those along the edges are marked with bold numbers. The remaining intervals are calculated values.



(a) Addition (b) Subtraction

Figure 2: Example of the basic two network operations

On Figure 3 we can follow a network evaluation starting from the input nodes. We consider the data in the input nodes as known. It can be imagined that these values are based on earlier, recorded data. The network evaluation is completed by using the previously presented rules. Starting from the input nodes, all nodes and edges are evaluated getting wider and wider intervals. In the output nodes we obtain then the widest intervals.

2.2. Measurement in a node

The aim of the measurement in a node, is to reduce the uncertainty in the influenced output nodes. By counting the letters in a node, we obtain the values at the outgoing edges too. Hence, we do not distinguish the node measurement from the respective edge measurement and we consider the edge measurement cost approximately equal to the measurement cost in the given node. The cost of the measurement in a node with manual counting is in our study now equal to the number of the letters, while in the case of automated counting the cost is ten times cheaper per letter.

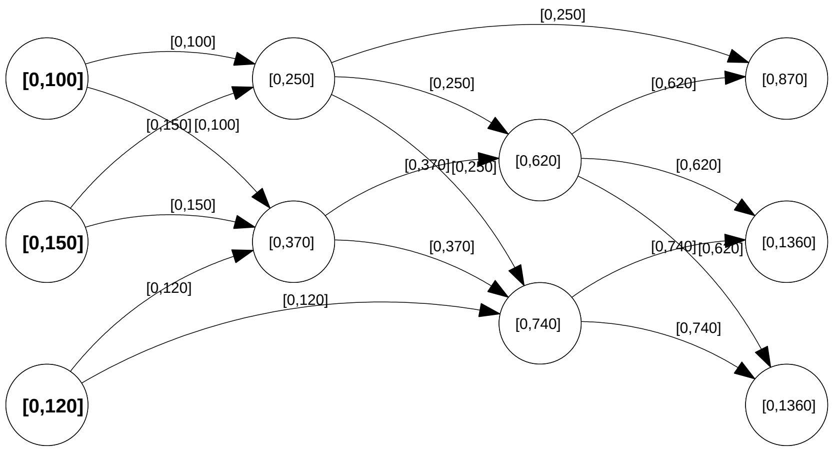


Figure 3: A network evaluation starting form the input nodes

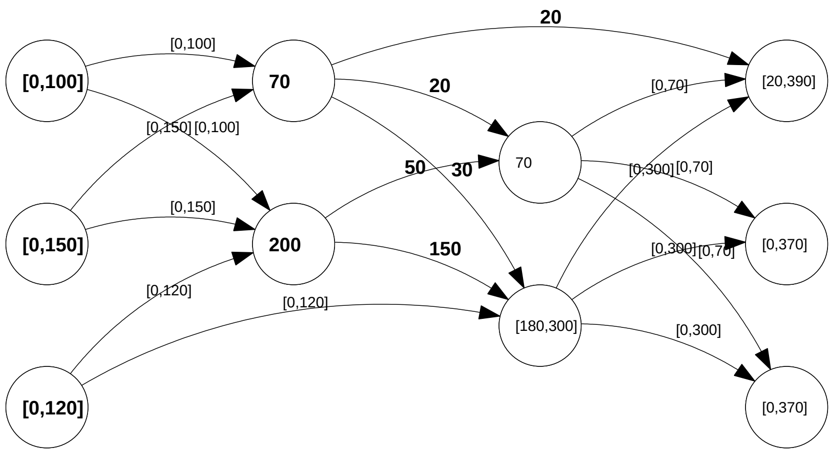


Figure 4: A full network evaluation example calculation using measurements in nodes

After a measurement in a node, we always evaluate the network using the previously presented rules. On Figure 4 we can follow the effects of two node measurements and a network evaluation. As a result, narrower intervals can be observed in the output nodes. Our aim is to achieve the user set precision in the output nodes with the smallest measurement cost. In the next section, we describe a solution algorithm for this problem.

3. Solution approaches

We have developed a greedy algorithm which tries to find those nodes which have the highest influence to the given output node with minimal cost. The main idea of the algorithm is the following: we choose always that node from the network, where the measurement is cheap, and the same time the uncertainty is the largest one temporarily. As it can be expected, this procedure can result in the largest improvement in the precision of the output node value. The main steps are:

Algorithm 1: A greedy algorithm for the determination of the measuring nodes

```
1  function GreedyAlg();
2     tCost = 0;
3     mNodes = {∅};
4     IntializeNetwork;
5     Evaluate(network);
6     w = width(outputnode.interval);
7     while w > a prescribed value do
8        nodelist = getList(outputnode);
9        node = findNode(nodelist);
10       mNodes = mNodes ∪ node;
11       cost = Measure(node);
12       tCost = tCost + cost;
13       Evaluate(network);
14       w = width(outputnode.interval);
15    end
16    return mNodes, tCost
```

At the beginning of the algorithm (lines 2 and 3) we initialize the total cost of the measurements and the measured nodes list. In line 4, the input nodes of the network are initialized with respective intervals, based on earlier recorded data, while in line 5 the network is evaluated starting from these input nodes.

The algorithm contains a main iteration cycle, and the steps from line 8 to line 14 will be repeated until a suitable condition will come true. At the beginning of the iteration (line 8) we actualize the available nodes which have influence to the given output node. The *getList* function takes as parameter the output node and returns those nodes which are not measured yet and may influence the output node. In the second step of the iteration cycle (line 9) we select the most promising node using the *findNode* function which takes as parameter the previously returned node list. We choose here the automated node with the widest interval if it is available, otherwise a node with manual counting having the widest interval. The reason behind the widest interval selection is that it yields the largest improvement in the precision of the output node value. The returned node is added to the measured node list in line 10. After we have found the most promising node, we measure it (line 11) by replacing the node's and the outgoing edge's interval data by the real traffic data. The node measurement cost is calculated by dividing the replaced interval midpoint by ten, if it is an automated node, otherwise it is equal to the midpoint of the interval. In line 12, the total cost is updated. The last procedure of the main iteration is the network evaluation (line 13) using the previously described rules. We also recalculate the width of the output node in line 14. The algorithm is stopped when the uncertainty represented by width of the interval that belongs to the output node became smaller than a prescribed value. In line 16, the measured nodes and the total cost of the measurement is returned.

4. Computational test

We have completed a numerical test which is aimed to show the performance of the described algorithm. For testing purposes, we use a network with 23 nodes, from which there are 4 input nodes (collection), 2 output nodes (delivery) and 5 nodes with automatized counting. The measured values in the nodes and along the edges are generated randomly with uniform distribution getting a consistent network. The input node values are generated from the interval $[50, 200]$. In the algorithm we have tested just one output node. For each case of the generated network, we set the uncertainty interval width to 200. The algorithm was tested on 50 such networks generated randomly, and we recorded the cost of measurements and the number of measured nodes (see Table 1).

Test case	AlgCost	NrNodes
1	282.9	6
2	394.9	6
3	452.8	7
4	258.4	6
5	399.1	6
6	272.5	6
7	400.8	7
8	316.9	6
9	266.4	6
10	455.5	6
...	...	...
50	276.4	6
Average	334.1	6.1

Table 1: The measurements costs and the number of measured nodes

In a particular example problem, the input nodes have the following starting intervals: $[0, 101]$, $[0, 124]$, $[0, 123]$, $[0, 61]$. The set precision of the output node was 200. As the result of the first evaluation of the network we get $[0, 13455]$ in the output node. After a few iteration of the algorithm we get the $[149, 312]$ interval in the output node and we stop the algorithm. Usually the found precision is better than the one set at beginning of the algorithm. The number of measured nodes during the algorithm in order to find the set precision are six, with the total measuring cost of 210. Among the six nodes there are four automated nodes and two with manual counting.

The conclusion of the numerical test is that the greedy algorithm we designed was capable to solve the randomly generated problems in a satisfactory way. In the future, we would like to extend the algorithm for more than one output node to be handled, and we also want to modify the network evaluation in order to get narrower intervals in the output nodes. Furthermore, we will try other solution approaches and applying these methods on real life applications.

References

[1] Bell, M.G.H., Iida, Y., Transportation Network Analysis, Wiley, Hoboken, 1997.

[2] Ben-Akiva, M., Bierlaire, M., Burton, D., Koutsopoulos, H.N., Mishalani, R., Network State Estimation and Prediction for Real-Time Traffic Management, *Networks and Spatial Economics*, Vol. 1 (2001), 293–318.

[3] Cremer M., Keller H., A new class of dynamic methods for the identification of origin-destination flows. *Transportation Research*, Vol. 21B (1987), 17–32.

[4] Ding, Y., Mirchandani, P.B., Nobe, S.A., Prediction of network loads based on origin-destination synthesis from observed link volumes, *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1607/1997 (2007), 95–104.

[5] Rump, S.M., *INTLAB – Interval Laboratory.* In: T. Csendes (ed.): Developments in Reliable Computing, Kluwer Academic Publishers, Dordrecht, 1999, 77–104.

[6] Song, Q., Zhang, C., Li, X., Hao, F., Genetic algorithm based modeling and optimization of the borough postal transportation network, *Decision and Control, 2007 46th IEEE Conference*, New Orleans, LA, 2007, 2850–2855.

[7] Syberfeldt, A., Grimm, H., Ng, A., Andersson, M., Simulation-based optimization of a complex mail transportation network, *Proceedings of the 40th Conference on Winter Simulation*, Miami, Florida, 2008, 2625–2631.

[8] Tsekeris, T., Stathopoulos, A., Treating Uncertain Demand Information in Origin-Destination Matrix Estimation with Traffic Counts, *Journal of Transportation Engineering*, Vol. 134 (2008), 327–337.

László Pál
530104 Miercurea-Ciuc, Piaţa Libertăţii Nr.1., Romania
e-mail: `pallaszlo@sapientia.siculorum.ro`

Tibor Csendes
6720 Szeged, Árpád tér 2., Hungary
e-mail: `csendes@inf.u-szeged.hu`