KOMUNIKÁCIE
COMMUNICATIONS

Dear reader,

You have got in your hands this volume of the university scientific letters which is the first time completely devoted to informatics and its applications to a broad spectrum of scientific and professional branches.

Nowadays informatics penetrates almost every human activity and this issue tries to reflect those phenomena. Inside this volume we attempt to submit papers written by authors not only from the Faculty of Management Science and Informatics and other faculties of the University of Žilina, but we appealed to professionals from other cooperating universities to contribute to the topic of this volume.

In the frame of this issue there are found works focused on pure problems of informatics as distributed database and SQL or parallel algorithm, but most of the works are devoted to applications of informatics to transport. This part contains both surveys concerning intelligent transportation systems and such special topics as various sorts of routing and scheduling problems. An attention is also paid to economic problems and distribution logistics.

I would like to express my opinion that this volume would attract attention of professionals from relevant scientific branches and ignite their interest in some future cooperation.

Jaroslav Janáček

Karol Matiaško *

ALLOCATION FRAGMENTS OF THE DISTRIBUTED DATABASE

The paper describes the distribution fragments of the database under a mathematical model with criterial function involving the influence of the Transaction and Concurrency proccesing in Database systems. The model could solve variants for replication of the fragments setting constraints in the model. This approach is prepared for revision of actual distribution using real values of the database (cardinality of tables, referential integrity, important requests etc).

1. Introduction

The design of a distributed database system involves making decisions on the placement of data and programs across the sites of a computer network. In distributed database systems the main problem of distribution is the data distribution.

The Database Allocation Problem (DAP) model dates back to the mid-1970s to the work of Eswaran (1974) [Eswaran75], Levin and Morgan (1975) [Levin75], and others. One of the best is described precisely in [Ozsu91]. DAP has been studied in many specialized settings. In 1975 Eswaran [Eswaran75] proved the simple file allocation model as NP-complete. All known solutions of the allocation were solved with heuristic algorithms.

2. Mathematical model

Our model is based on the work of Valduriez and Ozsu [Ozsu 91] and teamwork of Jaroslav Pokorný from Charles University [Pokorný92] with enlarged results of the research project in our university.

For an allocation model we need to know: database information, site information, network information and set of constraints. Each of them defines the set of parameters for the allocation model. The cost unit will be a/the time unit.

Database information

We need to know:

The set of fragments, [Matiasko02]
The size of each fragment,
The selectivity of each fragment,
The read access,
The update access,
The read polarization,
The update polarization.

The size of fragment.

The size of the fragment F_j is given by

$$size(F_j) = card\ (F_j) * length(F_j)$$

where

$length(F_j)$ is the length in bytes of one tuple of fragment F_j,
$card\ (F_j)$ is the cardinality of the fragment F_j and it is number of tuples in the fragment.

The selectivity of the fragment

The selectivity of the fragment F_j is given by $seli(F_j)$ where it is number of tuples of F_j that need to be accessed in order to precede q_i.

Read access

Read access f^r_{ij} is the number read access (frequenting of requests) that the query q_i makes to a fragment F_j during its execution[Matma99a, Matma99b].

Update access

Update access f^w_{ij} is the number update access (frequenting of requests) that the query q_i makes to a fragment F_j during its execution.

Polarization read access

Polarization read access r_{ij} is the localization the fragments in the query

where

$r_{ij} = 1$ if the query q_i reads from the fragment F_j
$r_{ij} = 0$ if the query q_i does not read from the fragment F_j

Polarization update access

Polarization update access u_{ij} is the localization the fragments in the update query

where

$u_{ij} = 1$ if the query q_i updates the fragment F_j
$u_{ij} = 0$ if the query q_i does not update the fragment F_j

* **Karol Matiaško**
Department of Informatics, Faculty of Management Science and Informatics, University of Žilina, E-mail: karol.matiasko@fri.utc.sk

Site information

For each site of the computer network in Slovakia we need to know:

- set of the clients computers C_{jk} and the set of the queries q_i running on the these clients' computers,
- storage capacity,
- processing capacity.

The unit cost of storing data at site S_k will be CM_k.
The costs of processing one unit of work at site S_k will be CP_k
The work unit should be identical with read and update access.

Network information

For the network we need to specify the communication cost. c_{ij} denotes the communication cost between site S_i and S_j. This cost depends on the protocol overhead, distances between sites, channel capacities, etc.

For each query q_i it is necessary to solve the simple decomposition operation.

Decision variables

The decision variable is xij, and it is binary.

$x_{ij} = 1$ if the fragment F_i is stored at site S_j
$x_{ij} = 0$ if the fragment F_i is not stored at site S_j

Objective function

$$minimize\ N = \sum_{\forall q_i \in Q_i} ND_i + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} NM_{jk},$$

or

$$N = \sum_{\forall q_i \in Q_i} ND_i \quad \text{if the memory costs are not important}$$

where
ND_i is the query processing cost of application q_i
NM_{jk} is the fragment storing cost of fragment F_j on the site S_k

The storage costs are given by

$$NM_{jk} = CM_k * size(F_j) * x_{jk}$$

and the two summations give us the total storage costs at all sites for all fragments of the computer network.

The query processing costs are given by

$$ND_i = NDB_i + NT_i$$

where
NDB_i is database-processing cost for the application q_i
NT_i is transmission cost for the application q_i

The processing costs are given by

$$NDB_i = NRW_i + NIC_i$$

where
NRW_i is the access cost for the query q_i to fragment F_j
NIC_i is the integrity and concurrency enforcement cost for the query q_i to fragment F_j

The access costs are given by

$$NRW_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} (u_{ij} * f_{ij}^w + r_{ij} * f_{ij}^r) * x_{jk} * CP_{jk}$$

The summation gives us the total number of update and read accesses for all fragments referenced by the query q_i. Multiplication by CP_k gives us the cost of this access at site S_k.

The NI cost and NC cost can be specified much like the processing component and depend on the actual computer, operating system, database system and the set of queries performed on the actual site of the computer network.

$$NIC_i = (KNI_i + KNI_i) * NRW_i$$

KNI_i is the integrity enforcement coefficient for the query q_i to fragment F_j
KNC_i is the concurrency coefficient for the query q_i to fragment F_j

$0 <= KNI_i <= 1$
$0 <= KNC_i <= 1$

The transmission cost

The transmission costs are different for read and for update access. If the update request exists, it is necessary to make it on all sites where replicas are situated. For read access we need read only one of the copies.

The transmission cost for the query q_i is given by

$$NT_i = NTW_i + NTR_i$$

The update component NTW_i of the transmission is

$$NTW_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} (f_{ij}^w * u_{ij} * x_{jk} * w_{z(i),k}(F_j)) + \\ + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} (f_{ij}^w * u_{ij} * x_{jk} * w_{k,z(i)}(F_j))$$

where the first term is sending the update message to the originating site i of q_i, to all the fragment replicas that need to be updated. The second term is for the confirmation. [Matgr98]

The value $w_{i,k}$ is the value of the transmission time for sending the request or answer message from the origin site of the query q_i to the site S_k.
For $w_{z(i),k}$ we suppose $w_{z(i),k}(F_j) = length(F_j)/V_{z(i),k}$
$z(i)$ is the assignment the origin of the query q_i

The retrieve component NTR_i of the transmission is

$$NTR_i = \sum_{\forall F_j \in F} f_{ij}^r \min_{\forall S_k \in S} (r_{ij} * x_{jk} * w_{z(i),k}(F_j)) + \\ + ((r_{ij} * x_{jk} * (sel_i)(F_j)/fsize(F_j)))) * 1/V_{z(i),k}$$

where the first part represents the cost of transmitting the read request to those sites which have copies of fragments that need to be accessed. The second one gives transmission cost for the result of the request.

V_{ij} is the transmission velocity from the site S_i to the site S_j

For w_{ik} we suppose $w_{ik}(F_j) = length(F_j)/V_{ik}$

Constraints

The response time constraint

Let the set $T = \{T_i^Q\}$ of the maximum response time of q_i, $q_i \square$ Q exist, then

$$NDBi \leq T_i^Q, \quad \forall q_i \in Q$$

execution time of q_i is less equal than maximum response time of q_i

The storage constraint

If $M = \{m_k\}$., $Sk \square S$ is the set of the storage capacity at each site S_k then

$$\sum_{\forall F_j \in F} size(F_j) * x_{jk} \leq m_k, \quad \forall S_k \in S$$

3. Experiments

For the verification of the model we used Greedy Heuristic [Albandoz94], [Francis 89], [Matiaško98] with orientation to the next experiments:

1. Basic variant – suboptimal solution with location fragments without replication
2. Centralized variant – suboptimal solution with centralized variant, when all fragments are localized on the same node
3. Nonfragmented variant – suboptimal solution without fragmentation,
4. Modificated variant – suboptimal solution with changing ratio destructive and nondestructive operation for the basic variant

A data model and data of information system of our university were used for the experiments with allocation. For computation as a data sample, data of 20 real applications from the information system of our university were used, which was working on five database relations and fragments allocation to five nodes of the university network. Two of these were used on the remote campuses in Prievidza and Ružomberok, and the others were used within the campus in Žilina.

The sets of fragments $F = \{F_i\}$ were defined, where particular fragments corresponding with relations or fragments of relations under the following data model:

- Relation *Student* is horizontally fragmented by study town to
  F_1 is relation StudentZA
  F_2 is relation StudentPD
  F_3 is relation StudentRB
- Relation *Person* is horizontally fragmented by derived fragmentation by joining relation Student, with a study town to
  F_4 is relation PersonZA
  F_5 is relation PersonPD
  F_6 is relation PersonRB
- Relation *Education* is horizontally fragmented by derived fragmentation by joining the relation Student, with a study town to
  F_7 is relation EducationZA
  F_8 is relation EducationPD
  F_9 is relation EducationRB
- Relation *Course* is fragment Crepresents static part of database.

Applications:

As a set of application $A = \{a_i\}$ we prepared 10 of the most typical selections and 10 of the most typical destructing operations from our university information system which created an experimental base for verification functionality of allocation for various counted variants.

a_1 – selection form $F_{1*} * F_{4*} * F_7 * F_{10}$
a_2 – selection form $F_{2*} * F_5 * F_8 * F_{10}$
a_3 – selection form F_3, $* F_6 * F_9 * F_{10}$
$a_4 = a_1 \otimes a_2 \otimes a_3$,
a_5 – selection form $F_1 \otimes F_2 \otimes F_3$
a_6 – selection form $F_4 \otimes F_5 \otimes F_6$
a_7 – selection form $F_3 \otimes F_7 \otimes F_9$
a_8 – selection form $F_1 * F_4 \otimes F_{2*} * F_5 \otimes F_{3*} * F_6$
a_9 – selection form $F_7 * F_{10} \otimes F_8 * F_{10} \otimes F_9 * F_{10}$
a_{10} – selection form F_{10}
a_{11} – a_{20} update in the fragments F_1 to F_{10}

where $\otimes$ is operation UNION

The values of monitored features were measured during a normal running of the information system. These features represented frequentations of nondestructive operations, selection of particular fragments, response times between a workplace of the network, size of relations of particular fragments and duration of elementary operations.

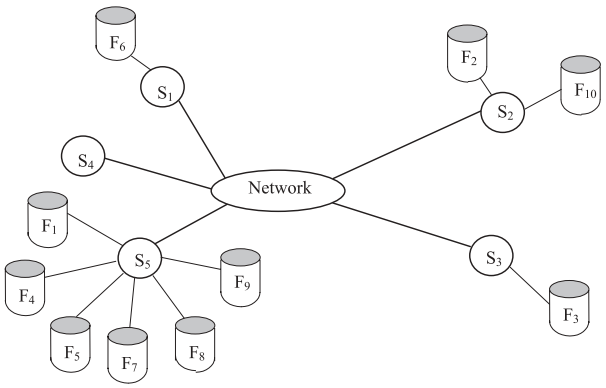


Fig. 1 Allocation of the fragments with one-level replications

First experiment presents the basic variant. The main goal is searching the suboptimal solution of the one level fragmentation. One-level fragmentation means that each fragment will be used only one time. The best allocation of the fragments is illustrated in Fig. 1.

The objective function for this variant has the value of 878202. This result shows that most fragments are allocated to the workplaces, which provides minimal cost considering transmission speed in the network.

We prepared an intuitive allocation, which related with the method BestFeed [Ceri84]. In this variant every fragment is situated to that workplace, under its maximal query frequency. If we suppose no destructive operation, the objective function enhances to the value 783035 and another fragment allocation – Fig. 2.

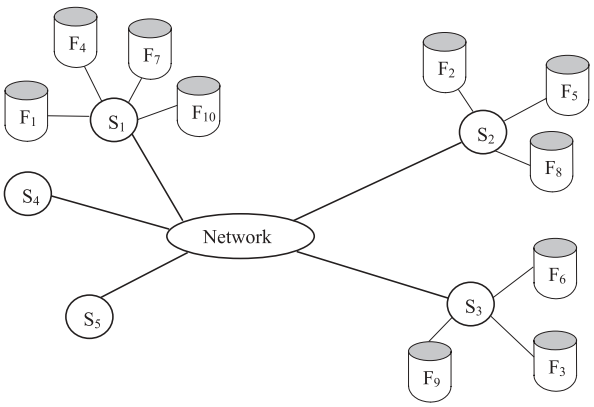


Fig. 2 Intuitive Fragments Allocation

From the point of view of destructive operation (DELETE, INSERT, UPDATE), then optimal allocation is another – Tab. 1, and objective function has the value of 362417. It is important and interesting to watch the influence of destructive operations to behavior of the whole system.

Allocation fragment only for the destructive operations Tab. 1

362417	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
S1	0	0	0	0	0	1	0	0	0	0
S2	0	1	0	0	0	0	0	0	0	1
S3	0	0	1	0	0	0	0	0	0	0
S4	0	0	0	0	0	0	0	0	0	0
S5	1	0	0	1	1	0	1	1	1	0

During experiments the *centralized variant* was made. Experiments with all allocated fragment are always on the same node. For each node we get one variant of the solution. The results are in the Tab. 2.

According to the results the centralized variant would be the best as allocated fragments on the node S_4 with objective function value 953792.

Table of the costs with real and percentage deviation form optimum (N – cost, DN – difference cost of optimal value, % – difference cost of optimal value) Tab. 2

	N	DN	%
Variant1	878202	0	0
Variant2	2077116	1198914	57
Variant3	1754237	876035	49
Variant4	2624590	1746388	66
Variant5	953792	75590	7
Variant6	1026311	148109	14

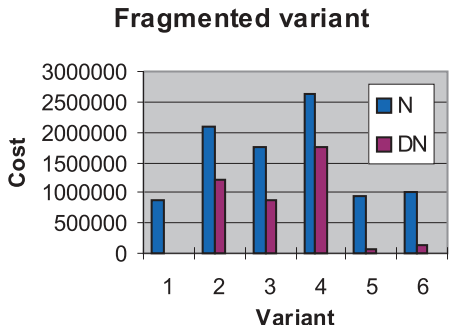


Fig. 3 Graph of costs for every variant of the solution.

When we treat *the nonfragmented variant*, in which the fragments F_1, F_2, F_3 collect one fragment, allocated always on the one node, and by the same way fragments F_4, F_5, F_6 and fragments F_7, F_8, F_9 then the cost for distribution has the value of the objective function 1000908 – (Tab.3).

The result for the nonfragmented variant Tab.3

	N	DN	%
Nonfragmented variant	1000908	122706	12ą

When we compare the result, which we get for the fragmental variant, it is different from the optimal value by 12 percent.

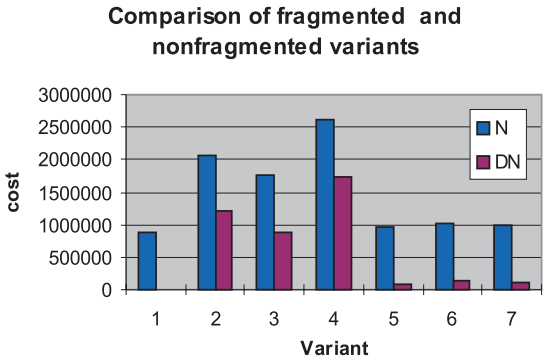


Fig. 4 Comparison of fragmented and nonfragmented variants

Change of the cost when the number of the "select" is constant — Tab. 4

Variant	N1	DN	%	Destructive operation [%]
8	1291932	413730	32	100
9	1235437	357235	28	90
10	1178942	300740	25	80
11	1129705	251503	22	70
12	1065964	187762	17	60
13	1009473	131271	13	50
14	952984	74782	7	40
15	896491	18289	2	30
16	840000	-38202	-5	20
17	792581	-85621	-11	10

Change of the cost when the number of the "update" is constant — Tab. 5

Variant	N2	DN	%	Nondestructive operat. [%]
18	1277275	399073	31	100
19	1206130	327928	27	90
20	1134989	256787	22	80
21	1075923	197721	18	70
22	992704	114502	11	60
23	921562	43360	4	50
24	850418	-27784	-4	40
25	779275	-98927	-13	30
26	708133	-170069	-25	20
27	636991	-241211	-38	10

From the point of view of the *modified variant*, we compared two situations. In the first variant we watched how the value of the objective function is changed (N1) when the number of the selected operation (only SELECT) is constant, and the number of the destructive operation is changed. At the beginning of this experiment the frequencies of all the kinds of operation are the same. On the next variant the number of the destructive operations is reduced by 10percent. The objective function is improved by 30percent of the number of destructive operations. DN is difference of the cost for the variant and optimal.

In another case of this variant we watched the change of the value of the objective function (N2) when the number of the destructive operations is constant and the number of the nondestructive is changed, as in the previous variant, in each step by 10percent. The objective function value is improved by 50 percent of the number of nondestructive operations. DN is the difference between variant costs and optimal costs.

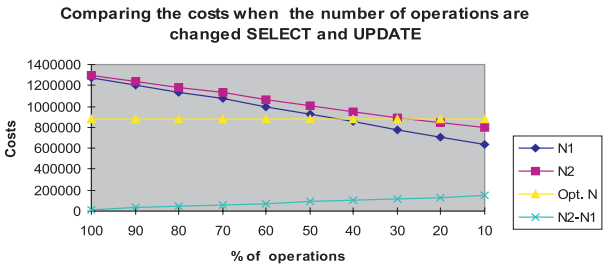


Fig. 5 Graph of the dependence costs and ratio change of the select

4. Conclusion

Development of information technology allows development of information systems effectively and in harmony with organization structure of firms. Therefore, distributed database systems are the tools that are helpful for the development of those systems. But designing the data model for a distributing database system is always challenge from the fragmentation database to the allocation the fragments or all databases, regardless of the available conditions.

References

[1] ALBANDOZ, J. P. e col.: *Lecturas en Teoria de Localization*, Universidad de Sevilla, 1996,
[2] CERI, S., PELAGATTI, G.: *A solution method for the Allocation problem in Distributed Databases,* Process Letters, 10, 1982,
[3] ESWARAN, K., P.: *Placement of records in a file and file allocation in a computer network*, Information Proccessing 1974: pp. 304-307, North Holland Publ. Co., Amsterdam, 1974,
[4] ESWARAN: *The notions of consistency and predicate locks in a Database Systems,* CACM 11,1975,
[5] FRANCIS, R. L., MIRCHANDANI, P.: *Discrete location theory,* Wiley, New York 1989,
[6] LEVIN, K. D. and MORGAN, H. L.: *Optimizing distributed databases- A Framework for research,* Proc. AFIPS NCC 1975, pp. 473-478, AFIPS Press, 1975,
[7] MATIAŠKO, K.: *Distributed database modeling (in slovak),* Žilinská univerzita, 1998, Žilina,
[8] OZSU, VALDURIEZ: *Principles of Distributed Database Systems*, Prentice Hall,Englewood Cliffs, New Jersey 1991
[9] POKORNÝ, J., SOKOLOWSKY, P., PETERKA, J.: *Distributed database systems (in czech)*, Academia, Praha 1992.
[10] MATIAŠKO, K.: *Database systems (in slovak)*, 2002, EDIS Žilina,
[11] MATIAŠKO, K., GRONDŽÁKOVÁ, E.: *Data Migration*, Studies of the Faculty of Management Science 7, 1998
[12] MATIAŠKO K. MARTINCOVÁ: *Principles of Query Processing decomposition and parallel scheduling of the execution Part I.*, Ouerry decomposition, Studies 8/1999, Faculty of Management Science and Informatics, 1999
[13] MATIAŠKO K. MARTINCOVÁ: *Principles of Query Processing decomposition and parallel scheduling of the execution Part II.*, Parallel scheduling of the query execution, Studies 8/1999, Faculty of Management Science and Informatics, 1999.

kamUNikácie
COMMUNICATIONS

Petr Cenek *

INTELLIGENT TRANSPORTATION SYSTEMS

Intelligent transportation system (ITS) stands for a transportation system with improved quality of control using an added "intelligence". Improved control can be attained by optimisation methods, by simualtions, interactive methods or artificial intelligence. Anyhow, a suitable model of the system, correct input data, optimisation routines and user-friendly environment for data input and output are typical parts of such a system. Basic parts of a transportation system, transportation processes and their models, optimisation methods and problems of implementation of an ITS are discussed in the paper in more detail.

1. Introduction

Intelligent transportation systems (further abbreviated as *ITS*) is a term used to describe transportation systems, which use a certain "intelligence" to improve the quality of their control, planning and management and thus can offer better services at possibly low costs. Costs will mean not only the resources directly spent on transportation but also secondary impacts on the environment (by exhausts, noise etc.). When discussing an intelligent transportation system, intelligence or an intelligent system is to be defined at first. The *intelligence* (used in an ITS) can be generally defined as:

"Ability to perceive logical relationships and use one's knowledge to solve problems and respond appropriately (or in an optimal way) to novel situations."

Artificial intelligence in computers is supposed to be a "capability of performing some functions usually associated with human reasoning".

The above mentioned definitions show the common features of an intelligent system, which should accumulate information about the system and its environment, evaluate it in a creative way and choose an optimal way of future behaviour or an optimal control of a system.

Intelligent transportation system should comply with these definitions and accumulate, evaluate an information about the state of the system and use it for an appropriate control of transportation processes. This definition can be used for any process or task in transportation systems. A vehicle should pick up information on a road, vehicle itself and traffic on the road and control its behaviour according to this information to minimise a consumption, to minimise an impact on the environment, to minimise a travel time etc.

The optimisation can be done on a level of a dispatcher control, when a dispatcher can choose a suitable vehicle and its path for a demanded transport. ITS can use a navigation system GPS (Global Positioning System) for a localisation of a vehicle and finding an optimal path to a destination. Information systems on transportation services (free parking places, traffic density on roads, time tables for transit systems etc) serve similarly on a level of a dispatcher control. Optimisation on this level should find an optimal (shortest, fastest, cheapest) path to a destination in a network, optimal routing for a collection or a distribution of goods or passengers, optimal allocation of resources (vehicles, crews) etc.

A strategic decision can include optimal location of transportation and logistic centres or a network design problem, which seeks for an optimal set of connections in a network to serve transportation demands. Practical examples of a location of centres can be location of warehouses, railway or bus stations or postal boxes. The network design can be interpreted also as an optimal choice of roads or railroads to be built or reconstructed, telephone lines or connections among computers in a computer network, which are all strategic and long lasting decisions. The same task can also be defined for some transport services to be introduced such as bus lines, direct freight trains, air lines etc, which are just organisational decisions and can be done or changed with much lower costs.

To summarise the introductory notes: ITS introduces some extra "intelligence" into transportation systems, so that they can provide better services either at a higher quality level or by less operational costs. The way how to achieve these goals is to add a subsystem for collection and evaluation of necessary information, modelling of systems and optimisation methods for a choice of optimal decisions.

There has been no mention about *artificial intelligence* so far, which is known from computer science. The necessary intelligence can be added to a transportation system as a technical subsystem but can be also represented by a human operator who will show the characteristics of intelligence described in the introduction. Specifically, he will be able to use the available information

* **Petr Cenek**
Department of transportation networks, Faculty of Management Science and Informatics, University of Žilina

about the state of a system in an active and creative way and together with his own knowledge to insure an optimal control of a system.

1.1 Transportation system

The transportation system ensures the transportation of different commodities in a transportation network to a final destination in a demanded time and with an appropriate quality (comfort, safety, without a damage etc). Transportation supposes a "mass production", which means a repeated service for many customers or a common transport of many items at a time. The aspect of a mass production can be seen in a sense of:
- *transported quantity* (many passengers or pieces of goods at a time by one vehicle);
- *repetition in time* meaning, for instance, repeated transports on a public road shared by many users during a long time period.

A transportation system consists of:
- a fixed subsystem (transportation network or infrastructure), which is composed of nodes and edges of a network,
- a moving subsystem (vehicles and transportation flows),
- a control subsystem.

The quality improvements can be done in nodes or along edges of a network or can be done in a transportation system as a whole. The improved control can be also designed for various transportation modes (as a railway, road, air or water transport, postal and logistics services, combined and container transportation).

1.2 Systems control

The improved quality of a transportation system control was mentioned as a main goal of introducing ITS. That is why general principles of the control theory should be respected.

A classical approach is a closed loop control where a control subsystem collects the information about a controlled system, compares this information to defined target values and according to this evaluation decides on the control decisions. A similar approach can be applied for a control of transportation systems, namely a collection of information on a state of the system, evaluation of experiments with a model of a transportation system with various control strategies and choice of the best strategy for future control of the system.

2. Processes in transportation systems

As already mentioned, the control tasks can be approached on different levels, which means for individual vehicles, at nodes and edges of a network or for a transportation system as a whole. These different control problems will be discussed in further chapters and at least some examples of possible quality improvements by an improved control in ITS will be shown.

2.1 Vehicles

Control of individual vehicles is a problem of a control theory; even if there is an important difference to other technical systems in a fact that a human operator (a driver) usually operates vehicles. A human operator can use his intelligence to react in an active and creative way on various traffic situations, but he is also the least reliable element of the control subsystem. There are several topics to be approached when improving the control on a vehicle level:
- a man – machine system, driver's reactions on traffic situations and driver's psychology,
- introduction of "artificial intelligence" into a vehicle control subsystem using technical tools such as sensors for an obstacle and/or other vehicles detection, evaluation systems which recognise the traffic situation on a road and recommend a suitable way of driving or possibly give warning signals to prevent dangerous states,
- enhancement of vehicles "intelligence" using communication tools which enable an automatic communication among vehicles; such a communication system provides timely information for following vehicles on a future dangerous states recognised by a leading vehicle or on a critical behaviour of the leading vehicle (for instance, emergency breaking in front of an obstacle on a road); the following vehicles can then react in advance and so they can avoid collisions and in less critical situations may solve the situation at smaller costs (less intensive breaking for instance).
- a global positioning system GPS with other communication systems can similarly improve the quality of a vehicle control; vehicles can be informed about their own position and receive data necessary for their navigation (for finding a path to a destination), but may be also informed about a position of other vehicles and obstacles at their proximity even if such obstacles are invisible for a driver (behind a curve or in conditions of a limited visibility); again it is possible to avoid critical situations or to solve the critical situation at smaller costs,
- systems of a fully automatic vehicle control are the highest level of a vehicle control; such systems will be able to follow automatically the road to a desired destination, choose the optimal path, recognise and respect traffic on a road and prevent from any collisions with other vehicles moving in the same lane (longitudinal collisions) or with vehicles in other lanes (lateral collisions). Automatic control can be accomplished using many different technical devices such as various sensors, balises built in a road and read by sensors in a vehicle or precise satellite navigation systems GPS.

All the mentioned approaches are already technically solvable and automatic guided vehicles are already used on private roads within some industrial plants, amusement parks etc. A general acceptance of such systems is prevented by high costs, insufficient safety, necessity of specialised roads equipped with technical tools

(balises or inductive loops) and low speed of automatically guided vehicles (due to safety in public traffic). Possible collisions with other vehicles and pedestrians in traffic on public roads represent another type of obstacles for an allowance of such systems in a general use.

2.2 Processes on edges of a network

The edges of a network serve for motion of vehicles and transported commodities. The transportation of commodities along an edge is a basic process in all transportation systems and represents the main goal of a transportation activity – the displacement of passengers or other commodities in a transportation system. There are many vehicles running simultaneously on a road and so mutual conflicts have to be avoided. The vehicles must respect safety distances among them, they have to brake and accelerate according to a current traffic situation or they have to overpass slower vehicles. The mutual interaction of vehicles is examined in a traffic flow theory.

An improved quality of ITS control should offer better traffic control on a road, which would prevent any collisions and ensure a fluent motion of vehicles with a minimum fuel consumption and minimum negative impacts on the environment.

The priority goal should be in many cases to enhance a road capacity (the maximum number of vehicles which can go along a road per a time unit). Another important goal is improved safety of traffic. The goals can be achieved by various technical improvements:

- better dynamics of a vehicle (more effective brakes and higher performance of an engine) allows to decrease a necessary safety distance between vehicles and to increase a cruising speed and thus a road capacity can be enhanced,
- the same result can be achieved by reducing a reaction time of a vehicle control subsystem; the reaction time can be decreased by various technical tools for instance by sensors which measure the distance from the leading vehicle and an alert system for a driver or a direct automatic emergency braking; another system may warn a driver about an approaching obstacle on the road.
- the capacity of a railroad can be increased similarly by introducing new interlocking systems, which allow to decrease a follow up distance among the trains.

The problems of a traffic control are quite different in road transport and railway (or rail bound) transport. There is a vast number of possible situations in a road transport which can lead to a conflict with a leading or following vehicle, with a vehicle in an opposite direction or in other road lanes, because vehicles can use any part of the road (any lane). The situation is different in a railway transport, where trains can use only one rail track (or a section of a track) allocated to the train and so only longitudinal conflicts among trains are possible. A variety of conflict situations in a railway transport is thus limited. On the other hand, trains are bound to one lane (one track) only, cannot overpass other trains on a track and they have in general also worse dynamic characteristics. That is why the capacity of railroads is fairly low, especially when trains of different categories move on the track.

2.3 Processes in nodes of a network

Control of processes in a node of a network is much more diversified than on an edge. The node of a network can represent a railway station, a crossing of roads, a village or a town. There are many roads in a node where vehicles can stand or move along. A "mass production" or displacement of many items at a time was described as one of characteristics of a transportation system. This brings new problems of accumulation and sorting of items to be transported. The batches of items can be created by:

- *accumulation* of items which enter the node from outside world (passengers come to a station or postal parcels are brought into a post office)
- *sorting* from incoming vehicles or trains, which brought the transported items into a node and such items have to be sorted according to their destination address.

The node of a network usually represents a transportation network by itself (on a more detailed level of a model). Such a network is created by streets in a town or by trackage in a railway station. There are typically many vehicles, which must be parked on roads or move along them. All the vehicles compete for a position on a road, which is to be shared among the vehicles. The individual vehicles thus represent parallel processes, which have to be co-ordinated in a system. There are no grave problems in a real traffic, where human operators drive vehicles. Drivers know and respect traffic regulations and can creatively solve some exceptions, which may arise in a real traffic.

To introduce intelligence in such systems by technical means is quite a difficult task. An example of such a problem may be a simulation model of traffic in a node, where all decisions have to be modelled in a simulation program. The lack of a human intelligence and its creativity in solving exceptional situations can result in a deadlock in a simulation program. The system enters a situation, which cannot be solved by regular means (no process can continue and be finished) and so vehicles are stacked in their current positions and the simulation run must be stopped. The co-ordination of parallel processes is one of the most demanding tasks in a control of transportation processes.

2.4 Processes in a transportation system

Processes on a level of the whole transportation system should mainly serve to move the commodities among the nodes of a network. The optimisation will concentrate on the optimisation of paths in the network, or problems of space and time planning. The decisions can be done on several levels:

- *strategic level* decides on basic problems of building of a transportation system such as location of nodes and centres in a network, building of new edges (roads, railway tracks or waterways). The optimisation problems are typically of a combinatorial

nature and are very difficult to solve. On the other hand, the importance of such decisions is long-lasting and that is why it is possible to spend quite a long computation time to find the best alternative,
- *tactical level* can also decide on location of centres and building some relations to serve the transportation needs in a network; this time the interpretation of the problem is not building a physical infrastructure but rather taking organisational decisions (which trains and/or bus lines should operate, time tables, location of bus stops etc),
- *operational level* organises a work of individual vehicles, finds shortest paths and optimal paths for distribution and collection of transported items.

Planning and management on the level of ITS should utilise various optimisation methods to find optimal solutions on each control level.

3. Implementation of ITS

Intelligent transportation systems are usually implemented according to a transportation mode and to a service, which is offered to customers. This approach is typical for practical needs when an existing problem should be solved in a possibly fast and efficient way. This can lead to a redundant solving of similar problems, to a variety of different solutions and to their possible incompatibility.

The research in ITS can be organised in the opposite way, too, which means it should concentrate on general formulation of problems and common methods of control in ITS. The developed general tools can be then used in practical applications. A general acceptation of solutions, possibility of mutual communication and a unification of used tools and control methods is a desired characteristic of ITS. In this way a standard data on transportation networks will be used, a standard user environment can be offered to a user and at least basic optimisation tasks can be solved using standard well tested optimisation methods. This approach of a problem generalisation, solution of general problems and application of results is typical for a research at universities and that is why these general problems will be discussed in more detail.

3.1 Model of a transportation infrastructure

Transportation infrastructure is characterised by linear objects (roads, railroads, and waterways) and that is why linear co-ordinates are to be used. Therefore, locations in a transportation system will not be defined by Cartesian co-ordinates but will be addressed by a reference to an element (edge) of a network and by its longitudinal co-ordinate (distance from a beginning of the edge). The objects not lying directly on the network (on a road) can be addressed in a similar way. In this case, two co-ordinates will be used besides the reference on the network element. The first co-ordinate will again describe the distance (longitudinal co-ordinate) and the second co-ordinate will describe the distance from the nearest point in the network (lateral co-ordinate). Linear co-ordinate system is currently being used in many transportation systems and it offers a simple, unambiguous and generally valid identification of all objects in a transportation system.

Data on a network are typically vast files and their input is a costly process. Therefore, it is reasonable to input the data only once and then to share them by many users who can use various computers and operating systems. A *commercial database* provides an access to data for many users using the same operating system. Another tool such as *XML document* should be used to share the data among many computers with different operating systems.

Another important part of an infrastructure model is its graphical visualisation. A *graphical interface* is the best tool to provide a quick and understandable information to a user. Graphical output is suitable also for presentation of a transportation network, position of vehicles and their motion in a network.

3.2 Model of a vehicle and traffic flow

Vehicles are *dynamical systems*, states of which (position, speed, acceleration) are changing in time. A motion of several vehicles along the edge of a network must respect mutual interactions and possible conflicts on a road (a necessity of acceleration or braking, surpassing etc). Concurrent motion of several vehicles on a road is known as *traffic flow*. Some optimisation problems do not deal with a detailed model of a dynamical behaviour of individual vehicles or of a transportation flow as a whole and use only a static value of a *flow intensity* (or number of vehicles passing a point in a network during a unit of time). The flow intensity can thus replace traffic flow model for a certain class of optimisation problems.

Vehicle models are different for road and railway transports from a point of view of kinematics, dynamics and mutual interactions as well. The trajectory of a vehicle movement is unambiguously defined by a rail track and that is why only dynamics of a vehicle movement in one co-ordinate is controllable (along a rail track forwards or backwards). Similarly, kinematics of the movement is quite simple, as the front wheels and rear wheels must follow the geometry of a railroad. The co-ordination of several vehicles on a track is limited to an allocation of a track (or a part of it) to only one vehicle (train) because vehicles cannot pass each other.

Vehicle motion in a road transport (as well as in water or air transport) is not limited to one co-ordinate, but generally at least two co-ordinates can be controlled (the trajectory is not defined by a road but can be controlled by a driver). Kinematics is much more complicated because only the front vheels follow the chosen path while the rear wheels are pulled freely and follow the preceding positions of the front wheels. Similarly, speed of a train motion is defined by a railroad construction, while the speed must comply with a chosen trajectory in a road transport. The speed will thus be limited by a radius of a trajectory curve, by a traffic situation etc.

A different situation is also at co-ordination of several vehicles, when drivers and vehicles are fairly free in a choice of speed and trajectory (lane). The traffic must respect also vehicles in the opposite direction, which is in a railroad transport unthinkable as they are strictly separated.

Models of vehicles can be very different from each other and must reflect all the characteristic of a chosen transportation mode.

3.3 Information systems – collection, processing and storage of information

Control in a closed loop supposes an availability of an information about a state of a controlled system and a comparison of these data with a demanded goal, which allows generating appropriate control actions. The same approach can be used in a control of transportation systems or processes in transportation systems. A characteristic of a transportation system is that it is geographically widespread, which means that elements of the system are located in distant places and the information must be collected over information networks. Transportation systems differ from other technical systems also in an expected response time, which is usually quite long with transportation systems, typically in order of minutes (with an exception of direct driver response).

A perfect and timely information is crucial for a control of transportation processes. For instance, if there is a direct communication with vehicles and fast (on time) information on transportation demands, the vehicle fleet can be managed in much more effective way than when the control can be executed only in depots. Similarly the planes can be better exploited, or the trains can have only necessary number of wagons according to the estimated demand.

A use of telematic systems is an efficient way of collecting and processing information in distributed systems. Telematic systems combine telecommunication services for data transmissions with processing power of computers or computer networks. The effectiveness of such a control was proved by practical experience, when many transportation companies gained a significant competitive advantage by using those systems. That is why they may grow up, provide better quality of service at lower operational costs.

3.4 Control and optimisation methods

The last part of an efficient control subsystem is a suitable optimisation method. The input data and model of a transportation system provide a platform for experiments and search of optimal control decisions. There are several basic approaches how to find an optimal (or sub optimal) decision:

- *exact methods* of operations research or graph theory, which use a mathematical formulation of a problem and an algorithm will find an optimal solution by a strictly defined and proved process,
- *heuristic methods*, which use the same mathematical formulation of a problem, but the algorithms are based on a human experience with solving similar problems and give a possibly good sub- optimal solution; heuristic methods are used preferably for complex combinatorial problems, for which an optimal solution by exact methods would be unrealistic,
- *simulation methods*, which are used for modelling and analysis of stochastic systems, with random input values or changes in the system characteristics,
- *interactive methods*, which profit from an operator's experience with solving similar problems and his ability of fast space orientation in the graphical presentation of a problem; interaction of an operator with a computer can be also used to respect some additional constraints, which are hardly to be formulated mathematically,
- *artificial intelligence* uses expert knowledge of a system and proposes suitable decisions; the artificial intelligence is best suited to solve problems, where the solution is to be chosen from a number of variants.

4. Conclusions

The University of Žilina has been interested in optimisation of transportation processes for a long time. Developments and construction of ITS can profit from this experience and utilise the results of research works. Many control methods discussed in the paper are theoretically known, some of them are already implemented and directly used in practice. Some principles and methods are just suggestions for further research work. These problems encompass:

- Integration of available models and control tools into a unified environment and communication with other subsystems;
- Development of new elements of a control subsystem, which may be directed to
  - Design of a general model of a transportation infrastructure with graphical output [5], choice of relevant parameters for elements of a network, design of a standard database for transportation networks and design of a general template and procedures for storage, reading and transfer of XML documents describing a transportation network;
  - Design of a general model of a vehicle including dynamics, visualisation and animation routines [2];
  - Development of new and modifications of known optimisation methods suitable for a general model of a transportation network [1], [3], [4];
  - Research concerning new control problems such as control of parallel processes in transportation systems, control using time tables and construction of time tables and control under incomplete information and/or influence of a quality and availability of information on a quality of transportation process.
  - Practical application of developed control tools in various transportation modes and for different transportation processes.

References

[1] CENEK P., KLIMA V., JANÁČEK J.: *Optimisation of Transport and Telecommunication systems* (*Optimalizace dopravních a spojových systémů)*. EDIS-ZU, Žilina 1994
[2] CENEK P.: *Vehicle Kinematics and Microsimulation Models*. Journal of Information, Control and Management Systems, Vol. 1, No. 1, FRI-ŽU, Žilina 2003
[3] JANÁČEK J.: *Mathematical Programming (Matematické programování)*. EDIS-ŽU, Žilina 1999
[4] JANÁČEK J.: *Optimisation of Transport Systems (Optimalizace dopravních systémů)*. EDIS-ŽU, Žilina 1999
[5] JÁNOŠÍKOVÁ Ľ., SADLOŇ Ľ., CENEK J.: *Model of Intelligent Transportation Infrastructure.* Journal of Information, Control and Management Systems, Vol. 1, No. 1, FRI-ŽU, Žilina 2003.

kamUNikácie
COMMUNICATIONS

Stanislav Palúch *

BUS SCHEDULING AS A GRAPH COLORING PROBLEM

The fundamental vehicle-scheduling problem (VSP) is usually formulated as a matching problem for which there exists a polynomial algorithm. The formulation of VSP as a graph coloring problem may seem not to be practical since a graph coloring problem is NP-hard and it is not a good idea to solve a polynomial problem using a non polynomial algorithm. However, no polynomial algorithms have been found for generalizations of VSP and hence the graph coloring formulation offers good approximation algorithms for their solution.

This paper was supported by VEGA as a grant No. 1/0490/03.

1. Introduction

In recent vehicle scheduling papers many new scheduling problems arise namely in connection with multiple bus scheduling. Problems of heterogenous fleet are treated in [1], [2], many general problems of fleet management are described in [3]. Peško in [7] deals with multicommodity bus scheduling, in [9] with special fleet optimization problem, my paper [6] designes a graph theory approach to two bus scheduling problem. Many arising problems are NP-hard- a comprehensive overview of NP-hard transportation problems is in [8].

Standard vehicle scheduling is a polynomial problem, which can be polynomially reduced to a matching problem. However, its generalizations are no longer polynomial. This paper brings a graph coloring formulation of a fleet minimization problem. Such formulation is of no use for standard scheduling, but for its generalizations (for two bus scheduling problem) it offers a non polynomial algorithm which can be used as a heuristic, since it offers a good feasible solution in any stage of computing.

Trip s is an arbitrary quadruple $s = (dp, ap, dt, at)$ where dp, ap are the departure place and arrival place of the trip s and dt, at are departure time and arrival time of the trip s. Trip is a travel from a starting point to a finishing point of a route and is considered to be an elementary amount of the work of a bus.

We will say that the trip s_i *precedes* the trip s_j and we will write $s_i \prec s_j$ if the trip s_j can be linked after the trip s_i into a running board for one bus. If $s_i = (dp_i, ap_i, dt_i, at_i)$, $s_j = (dp_j, ap_j, dt_j, at_j)$, then it holds $s_i \prec s_j$ if and only if

$$dt_j \geq at_i + M\,(ap_i, dp_j), \tag{1}$$

where $M\,(ap_i, dp_j)$ is the travel time from arrival place of s_i to departure place of s_j. Relation $\prec$ is irreflexive and transitive.

Running board, or *running board of a bus* is an arbitrary nonempty sequence $T = s_1, s_2, ..., s_m$ of trips with property:

$$s_i \prec s_j \ldots \prec s_m \tag{2}$$

The number m will be used for *length of running board T*. We will write $s_i \rightarrow s_j$, if both trips s_i, s_j are provided by the same bus and the trip s_i is linked immediately behind the trip s_j. Note that $s_i \rightarrow s_j$ implies $s_i \prec s_j$.

For any given set $S = \{s_1, s_2, \ldots s_n\}$ of trips with precedence relation $\prec$ we can construct a bus schedule. *Bus schedule* of the set S of trips is a set of running boards

$O = \{T_1, T_2, \ldots T_k\}$ of the form

$$\left.\begin{array}{l} T_1 = s_{1,1} \rightarrow s_{1,2} \rightarrow \ldots \rightarrow s_{1,n(1)-1} \rightarrow s_{1,n(1)} \\ T_2 = s_{2,1} \rightarrow s_{2,2} \rightarrow \ldots \rightarrow s_{2,n(1)-1} \rightarrow s_{2,n(1)} \\ \ldots \\ T_k = s_{k,1} \rightarrow s_{k,2} \rightarrow \ldots \rightarrow s_{k,n(k)-1} \rightarrow s_{k,n(k)} \end{array}\right\} \tag{3}$$

so that every trip of the set S occurs exactly in one running board of O.

To every bus schedule $O = \{T_1, T_2, ..., T_k\}$ the number $C(O)$ - *the cost of bus schedule O* is assigned. We will say that the cost $C(O)$ is *separable* if it can be expressed as a sum of costs of all running boards $T_1, T_2, ..., T_k$, i. e.

$$C(O) = \sum_{i=1}^{k} c(T_i), \tag{4}$$

where $c(T_i)$ denotes the cost of running board T_i. In most simple cases the cost of running board $T = s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_m$ is the sum of all linkage costs:

$$c(T) = \sum_{i=1}^{m-1} c(s_i, s_{i+1}), \tag{5}$$

* **Stanislav Palúch**
Faculty of Management Science and Informatics, University of Žilina, Slovakia

The linkage cost denoted $c(s_i, s_j)$ represents mainly dead mileage expenses, however, it may include waiting costs and other penalties as well.

In this case we will say that the cost $c(T)$ is *linear*. We will say that the cost $C(O)$ of bus schedule O is *linear*, if C is separable and the cost c of running board is linear.

2. Two fundamental bus scheduling problems

The general goal of vehicle scheduling optimization is to find the bus schedule O with minimum cost $C(O)$. However, it showed useful to decompose the optimization process into two stages – minimization of the number of used buses and the minimization of the bus schedule cost provided the minimum number (or any feasible fixed number) of vehicles is used. This approach follows from practical experience when bus operators demand in most cases the bus schedule with a minimum number of vehicles.

There are two classical fundamental bus scheduling problems:

Fundamental problem I (FP I) To find a bus schedule with a minimum number k_0 of running boards (i.e. with a minimum number of vehicles).

Fundamental problem II (FP II) From all bus schedules with the given number k of running boards to find a bus schedule with a minimum total cost. (In most cases $k = k_0$, where k_0 is the minimum number of vehicles obtained by FP I.)

3. Mathematical model for linear bus schedule cost

For linear bus schedule cost $C(O)$ and $k = k_0$ we have the following mathematical model.

Let's denote

$$d_{ij} = \begin{cases} c(s_i, s_j), & \text{for i, j so that } s_i \prec s_j \\ \infty, & \text{otherwise} \end{cases} \qquad (6)$$

Let $x_{ij} \in \{0,1\}$ be a decision variable. If $x_{ij} = 1$ and $s_i \prec s_j$ it means that the trip s_j is linked after the trip s_i in a running board and $d_{ij} < \infty$ represents the corresponding linkage cost. If $x_{ij} = 1$ and $s_i \not\prec s_j$ it means that the trip s_i is the last trip of a running board and the trip s_j is the first trip in another (or maybe the same) running board. In this case $dij = \infty$. So the sum $\Sigma_{ij}\, d_{ij} \,.\, x_{ij}$ includes all linkage costs and as many items $d_{ij} \,.\, x_{ij} = \infty$ as the number of running boards represented by variables x_{ij}.

The mathematical model for FP II can be formulated as follows:

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}\, x_{ij} \qquad (7)$$

subject to

$$\left.\begin{array}{l} \sum_{j=1}^{n} x_{ij} = 1 \quad \text{for all } i = 1, 2, ..., n \\ \sum_{i=1}^{n} x_{ij} = 1 \quad \text{for all } j = 1, 2, ..., n \\ x_{ij} \in \{0, 1\} \end{array}\right\} \qquad (8)$$

Conditions (8) say that each trip can have only one predecessor and only one successor. Last formulation is a matching problem for which we have several effective polynomial algorithms. The same model works for FP I, too, but it can be even simplified by setting

$$d_{ij} = \begin{cases} 0, & \text{for i, j such that } s_i \prec s_j \\ \infty, & \text{otherwise} \end{cases} \qquad (9)$$

For $k > k_0$ the model for FP II can be several ways – one of them is the following. Replace ∞ by a great number K in (6) and add the constraint $\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}\, x_{ij} \geq k.K$ to (8). The graph theory formulation of FP II with given number k of running boards leads to the minimum cost flow problem with the given flow magnitude equal to k.

4. Heuristic approaches for regular bus schedule cost

There are many practical cases when the cost of bus schedule is separable but no longer linear. Thus for return bus scheduling problem, when the cost of bus schedule $T = s_1 \to s_2 \to ... \to s_m$ includes travel cost from arrival place of s_m to departure place of s_1

$$c(T) = \sum_{i=1}^{m-1} c(s_i, s_{i+1}) + c(s_m, s_1) \qquad (10)$$

the FP II is NP-hard. Other example of similar problems are a multiple depot bus scheduling problem, bus scheduling with special conditions such as constraints for working time, safety break, meal break etc. For none of them any polynomial algorithm has been found till now.

The FP I doesn't depend on the form of objective function, so the matching model gives us the solution with a minimum number of buses. We take this solution as the starting one for one of following neighbourhood search heuristic procedures:

- horizontal splitting – neighbourhood obtained from existing solution by combining trips of two running boards
- vertical splitting – neighbourhood obtained by combining heads and tails respectively bodies and mids of all running boards from the existing bus schedule.

The vertical splitting leads to the multiple application of matching algorithm. From many practical experiences it follows that the mentioned procedures are very successful for separable bus schedule cost.

5. Graph coloring model for FP I

Let S be a set of trips with precedence relation $\prec$. We will say that the trips $s_1 \in S$, $s_2 \in S$ are *incompatible* if $s_1 \not\prec s_2$ and $s_2 \not\prec s_1$. Otherwise we will say that the trips s_1, s_2 are *compatible*. Let $G = (V, E)$ be a graph with vertex set $V = S$ and edge set E defined

$$E = \{(u, v) \mid u \in V, v \in V, u \not\prec v, v \not\prec u\}. \tag{11}$$

In other words the edge set E is the set of all incompatible pairs of trips from S.

Let $T = s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_m$ be a running board with the trips from S. Then for every couple s_i, s_j, $i = 1, 2, ..., m$, $j = 1,2, ..., m$, $i \neq j$ it holds $s_i \prec s_j$ or $s_j \prec s_i$ and consequently $(s_i, s_j) \notin E$ – the set V_T of all trips from running board T is an independent subset of vertices in graph G.

On the other hand the given independent subset of trips $V_T \subseteq V$, for arbitrary pair $s_i \in V_T$, $s_j \in V_T$ it holds $s_i \prec s_j$ or $s_j \prec s_i$ – $\prec$ is a complete ordering of finite set V_T and hence the trips from V_T can be ordered into a linear ordered sequence – running board T.

In the terminology of graph coloring problem the subset $V_T \subseteq V$ is a running board if and only if all vertices from V_T can be colored by the same color. Then FP I can be formulated as follows: To solve FP I means to find a coloring of the graph G with a minimum number of colors.

6. Exact graph coloring algorithm

The following exact graph coloring algorithm comes from the outstanding Demel's book [4]. Suppose $V = \{1, 2, ..., |V|\}$ $n = |V|$. Let V_x be the set of all neighbours of the vertex x in graph G.

$$\text{Set } P(x) := V_x \cap \{1, 2, ..., x - 1\} \tag{12}$$

$$F(x) := \min\{i \mid 1 \leq i, \forall j \in P(x)\ i \neq B[j]\} \tag{13}$$

$$G(x) := \min\{i \mid B[x] < i, \forall j \in P(x)\ i \neq B[j]\} \tag{14}$$

$F(x)$ is the lowest feasible color number for vertex x. Provided the vertex x is colored with color $B[x]$, $G(x)$ is the lowest feasible color number greater than $B[x]$ which can be used for vertex x.

- **Step 0**. Set $B[1] := 1$ and sequentially for every $x = 2, 3, ..., n$ $B[x] := F(x)$.
- **Step 1**. Set $F_{MAX} := \max_{1 \leq x \leq n} \{B[x]\}$.
  Copy array $B[\ ]$ into array $RECORD[\ ]$.
- **Step 2**. Find in array $B[\ ]$ the lowest y such that $B[y] = FMAX$.
- **Step 3**. Set $x := max_{k \in P(y)}\{k\}$
- **Step 4**. If $x = 1$, STOP. Chromatic number of graph G is $\chi(G) = FMAX$ and the corresponding graph coloring of G is in array $RECORD[\ ]$.
- **Step 5**. If $G(x) \geq FMAX$ or if $G(x) > (\max_{1 \leq i < x} \{B[i]\} + 1)$, set $x := x - 1$ and Goto Step 4.
  Otherwise set B[x] := G(x), z := x + 1.
- **Step 6**. $B[z] := F(z)$. If $B[z] \geq FMAX$, set $y := z$ and Goto Step 3.
  If $z < n$ set $z := z + 1$ and repeat Step 6.
  If $z = n$ we have a new better solution. Goto Step 1.

The bad news is that this algorithm is not polynomial and there is no hope that it finishes in reasonable time even for the smallest real word instances. The good news is that after executing Step 1 at least once we have a feasible (but not necessary optimum) solution in the array $RECORD[\ ]$, so the algorithm can be interrupted in any time (after executing Step 1. at least once) and the coloring stored in array $RECORD[\ ]$ can be used as a suboptimum solution. This algorithm was implemented in C-language and practical experiences showed that it was able to find optimum relatively fast, but there still remained plenty of computing to confirm that there is no better solution. Since the matching model for FP I is exact and very fast, in this stage this approach is of no practical importance.

There are graph coloring algorithms with better performance e.g. methods based on constraint programming in [5], but the investigation how to modify them for two bus type problem is not finished yet.

7. Two bus type problem

Suppose we have two types of buses say type 1 and type 2 with different capacities $C_1 > C_2$ and different costs of running board $c_1 > c_2$. The set S of all trips consists of two disjoint subsets $S = S_1 \cup S_2$, $S_1 \cap S_2 = \emptyset$. The trips from S_1 can be performed only by buses of the type 1, because they need higher capacity. The trips from S_2 can use both types of buses.

A type of the bus has to be assigned to every running board $T = s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_m$. If for some the trip s_i from running board T it holds $s_i \in S_1$ the running board T has to be accomplished by a bus of the type 1 (otherwise T is infeasible). If the running board $T = s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_m$ consists only of elements of the subset S_2 it is possible to assign to it both types of buses. However, the better solution is to use smaller type 2 because it is less expensive. So we define the cost $c(T)$ of running board T as follows

$$c(T) = \begin{cases} c_1(T) & \text{if } T \text{ contains least one trip from } S_1 \\ c_2(T) & \text{if all trips from } T \text{ are from } S_2 \end{cases} \tag{15}$$

The cost of bus schedule $O = \{T_1, T_2, ..., T_k\}$ is separable, i.e. it holds (4).

The general two bus type problem is to find a bus schedule with a minimum cost $C(O)$.

The original idea was the same as in a single bus type case described in section 4 of this paper – first to solve FP I (to find the bus schedule with a minimum number of buses regardless of the bus type) and then to use heuristics for minimization of the cost $C(O)$. But the described heuristics working well in most of a single bus type cases failed in a two-bus type problem. The main reason is probably in the fact that there are not enough simple

paths from solution with a larger number of vehicles of the type 1 to the solution with a lower number. Another reason is that the objective function $C(O)$ doesn't express the "distance" of the existing solution to the solution with a lower number of buses of type 1 and hence the neighbour searching procedure cannot distinguish which solution is "closer" to diminishing the number of vehicles of the type 1. If we start from the existing solution with an optimum fleet structure we could avoid the described problems.

There are several possible analogies of FP I for two-bus type scheduling.

Fundamental Two Bus-type Problem a (FTBP Ia). For k_1 – the minimum number (or any given feasible number) of buses of the type 1 to find a bus schedule with a minimum number k of all buses.

Fundamental Two Bus-type Problem b (FTBP Ib). For k – the minimum number (or any given feasible number) of all buses to find a bus schedule with a minimum number k_1 of buses of the type 1.

Formulation of the analogy of FP II. for the two-bus type case is the following:

Fundamental Two Bus-type Problem II (FTBP II). From all bus schedules with given numbers k_1, k_2 of running boards of the type 1 and type 2 to find a bus schedule with a minimum cost $C(O)$.

Both FTBP Ia and FTBP Ib seem to be NP-hard, but I didn't succeed to prove this nor to find mentioned problems can be formulated as a known graph theory problem – list coloring problem.

The list coloring problem is the following generalization of a graph coloring problem: Let $G = (V, E)$ be a graph, $V = \{v_1, v_2, ..., v_n\}$, let $C_1, C_2, ..., C_n$ be sets of colors where C_i is a set of feasible colors for vertex v_i (list of allowed colors for v_i). The problem is to find a feasible coloring – to assign for every $v_i \in V$ a color from C_i such that no two adjacent vertices are assigned the same color.

We can define a further generalization of the list coloring problem – the minimum list coloring problem: Given a graph $G = (V, E)$ and a list of allowed $C_1, C_2, ..., C_n$ to find a feasible coloring with minimum total number of used colors.

Let $G = (S_1 \cup S_2, E)$ be the graph with vertex set $S_1 \cup S_2$ and edge set E containing all incompatible pairs of different trips from $S_1 \cup S_2$. By solving the FP I for the set of the trips S_1 we obtain k_1 – the minimum number of large buses of the type 1. For vertices $v_i \in S_i$ set $C_i = \{1, 2, ..., k_1\}$ for set $C_j = \{1, 2, ..., n\}$. Then to solve FTBP Ia means to solve the minimum list coloring problem in graph G with list of allowed colors $C_1, C_2, ..., C_n$.

The following modification of the exact graph coloring algorithm was designed for the mentioned special case of minimum list coloring problem.

- **Step I.** Set $G_1 = (S_1, E_1)$ were E_1 is the set of all incompatible pairs of different trips from S_1.
  Set $G = (V, E)$ where $V = S_1 \cup S_2$ and where E is the set of all incompatible pairs of different trips from V. .
  Sort the vertex set V so that vertices from S_1 proceed all vertices from S_2.
  Suppose $V = \{1, 2, ..., |V|\}$, $n - |V|$.
- **Step II.** Solve FP I for the set of vertices S_1 to find k_1
- **Step III.** Use exact graph coloring algorithm for the graph G_1 and stop it as soon as you find the first feasible solution with k_1 colors. Keep result coloring in array *RECORD*[ ].
- **Step IV.**
  Set $P(x) := V_x \cap \{1, 2, ..., x - 1\}$ (16)

  $$F(x) := \min[i \,|\, i \in C_x, 1 \le i, \forall j \in P(x)\ i \ne B[j]] \quad (13)$$

  $$G(x) := \min[i \,|\, i \in C_x, B[x] < i, \forall j \in P(x)\ i \ne B[j]] \quad (14)$$

  $F(x)$ is the lowest feasible color number for vertex x with respect to color list C_x. Provided the vertex x is colored with color $B[x]$, $G(x)$ is the lowest feasible color number greater than $B[x]$ which can be used for vertex x with respect to color list C_x. Functions $F(x)$, $G(x)$ are set minimums. Remember that if the set is empty the corresponding minimum $+\infty$. Since $C_x = \{1, 2, ..., n\}$ for $x \in S_2$ then $F(x) \le \infty$ for all $x \in S_2$.
- **Step V.** Set $B[x] := RECORD[x]$ for all $x \in S_1$ and sequentially for every $x \in S_2$ $B[x] := F(x)$.
  (Let's note that it can never happen $F(x) = \infty$ for $x \in S_2$.)
- **Step VI.** Set $F_{MAX} := \max_{1 \le x \le n} \{B[x]\}$.
  Copy array B[ ] into array *RECORD*[ ].
- **Step VII.** Find in array B[ ] the lowest y such that $B[y] = FMAX$.
- **Step VIII.** Set $x := max_{k \in P(y)}\{k\}$.
- **Step IX.** If $x = 1$, STOP. Chromatic number of graph G is $\chi(G) = FMAX$ and the corresponding optimum graph coloring of G is in array *RECORD*[ ].
- **Step X.** If $G(x) \ge FMAX$ or if $G(x) > (\max_{1 \le i < x} \{B[i]\} + 1)$, set $x := x - 1$ and Goto Step IX.
  Otherwise set $B[x] := G(x)$, $z := x + 1$.
- **Step XI.** $B[z] := F(z)$. If $B[z] \ge FMAX$, set $y := z$ and Goto Step VIII.
  If $z < n$ set $z := z + 1$ and repeat Step XI.
  If $z = n$ we have a new better solution. Goto Step VI.

Similarily to the exact coloring algorithm this algorithm will not finish in reasonable time, but it offers a feasible good solution in array *RECORD* in any time after executing Step V at least once.

References

[1] ČERNÁ, A.: *Heterogenous Bus Fleet Exploitation in Regional Bus Transport*, Slovak, (Využitie heterogenneho autobusového parku v mestskej a regionálnej doprave), Proceedings of the 5-th International Conference on Public Transport, Dom techniky, Bratislava, 21. – 22. 11. 2001, pp. 93–96.

[2] ČERNÁ, A.: *Optimization of Regional Bus Transport,* Czech, (Optimalizace regionální autobusové dopravy.) Proceedings of International Conference "Transportation Science"., (Věda o dopravě), Fakulta Dopravní ČVUT Praha, 6. – 7. 11. 2001, pp. 70–75.
[3] ČERNÝ, J.: *Fleet Management.* Selected Optimization Problems. Proceedings of the 8-th IFAC/IFIP/IFORS Symposium Transportation Systems Chania, Greece, june 1997, pp. 607–610.
[4] DEMEL, J.: *Graphs and their applications,* (Grafy a jejich aplikace), Czech, Academia Praha, 2002, ISBN 80-200-0990-6.
[5] JANOŠÍKOVÁ, L., STASINKA, R.: *Constraint Programming: An Application for Graph Coloring.* In: Journal of Information, Control and Management Systems, No. 2/2003. University of Žilina, Žilina. ISSN 1336-1716. (to appear)
[6] PALÚCH, S.: *A Graph Theory Approach to Bus Scheduling with Two Types of Buses.* Studies of the Faculty of Management Science and Informatics, Vol. 9, December 2001, pp. 53–57.
[7] PEŠKO, Š.: *Multicommodity Return Bus Scheduling Problem.* Proceedings, International scientific conference on mathematics, pp. 77–82, Žilina, ISBN 80-7100-578-9, (1998)

Štefan Peško *

SQL ALGORITHM FOR SOLVING MARKOV MODELS BY GRAPH METHOD

A simple graph algorithm for finding stabilized probabilities of the finite Markov models implemented in SQL is presented. The algorithm generates systematically all oriented spanning trees of a transition graph. The method is demonstrated on the computation of probabilities in the $MMPP_2/M/1/K$ queue.

Keywords: *Markov models, queue, graph algorithm, SQL algorithm*

1. Introduction

The original graph study of steady-state distribution for stabile Markov process with finite state sets is presented by Markl [1]. Stabilized probabilities π of the stable Markov process associated with a weighted digraph (called transition graph) $G = (S, E, c)$ are given by formulas

$$\pi i = \frac{B_i}{\sum_{j \in S} B_j} \quad i \in S, \tag{1}$$

where B_i, $i \in S$ is the sum of weights of all the spanning trees of G that have their roots in vertex i.

For processes with many states and many possible transitions between them is not easy to find all spanning trees. We apply a simple algorithm that systematically generates all spanning trees with given roots in SQL.

2. Basic definitions

We start with giving the some definitions from graph theory that will be used in this paper. A (simple) *digraph* $G = (V, E)$ consists of a finite set V of vertices and set E of edges – ordered pairs of distinct vertices; that is, each edge (u, v) is directed from *tail* u to *head* v. A (directed) $u - v$ *path* from vertex u to vertex v is such a sequence $u = v_0, (v_0, v_1), v_1, (v_1, v_2), v_2, ..., (v_{k-1}, v_k), v_k = v$ that $v_i \in V$ for $i = 0. ..., k$ and $v_i \neq v_j$ for $0 < i < j < k$, and that $(v_{i-1}, v_i) \in E$ for $i = 1, ..., k$. If $u = v$ then $u - v$ path is called a *cycle*. A digraph in which each pair of vertices lies on a common cycle is called *strongly connected*. A digraph in which for each pair $\{u, v\} \in V$ exists $u - v$ path or $v - u$ path is called *connected*. The *component* of a digraph is maximal connected subgraph of a digraph.

A digraph that has no cycle is called a *directed acyclic graph* (DAG). A *rooted tree* is a DAG in which one vertex, the *root*, is distinguished and in which all edges are implicitly directed to the root. (Note that in the standard definition [7] of the rooted tree the orientation of the edges is away from the root.) The *weight* of the (edge) weighted rooted tree $T = (V, E_T, w)$ we mean number

$$w(T) = \prod_{h \in E_T} w(h) \tag{2}$$

A *spanning rooted tree* (T, r) with root r, shortly *spanning r-tree*, of a strongly connected digraph G is a rooted tree T that is a subgraph of G and that contains every vertex of G. The *weight* of the a spanning r-tree (T, r) of a (edge) weighted digraph $G = (V, E, w)$ is weight $w(T)$ of the rooted tree $T = (V, E_T)$ given by (2).

3. SQL algorithm

The strongly connected digraph $G = (V, E)$ is given. We may assume without loss of generality that $V = \{1, 2, ..., n\}$ and root $r = 1$. The cases where $r \neq 1$ can be transformed to the case $r = 1$ by renumbering of vertices V. Let the set E be represented by a table *Edge* with two columns:

- *Edge.u* is head of a directed edge (u, v)
- *Edge.v* is tail of a directed edge (u, v)

and a table *Solution* with two columns of *array*[1, ..., *n*]:

- *Solution.tree* is the subtree – subgraf of spanning 1-tree
- *Solution.comp* is the components of *tree.Solution*

In Figure 1 we have the example of the spanning 1-tree $T = (V, H_T)$ where $V = \{1, 2, 3, 4, 5\}$ and $H_T = \{(2, 4), (3, 5), (4, 1), (5, 4)\}$ are represented by array *tree* = [0, 4, 5, 1 4]. In Figure 2

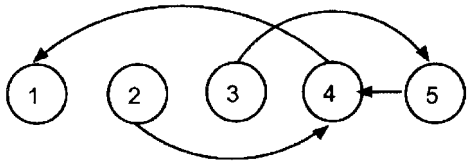


Fig. 1 The spanning 1-tree as array [0, 4, 5, 1, 4]

* **Štefan Peško**
Department of Mathematical Methods, Faculty of Management Science and Informatics, University of Žilina, Slovakia,
E-mail: pesko@frcatel.fri.utc.sk

we have the example of the subtree of the spanning 1-tree which is represented by array *tree* = [0, 4, 0, 1, 4] and has two components T_1 = {1, 2, 4, 5}, {(2, 4), (4, 1), (5, 4)}) and T_2 = ({3}, ∅) which are represented by array *comp* = [1, 1, 3, 1, 1].

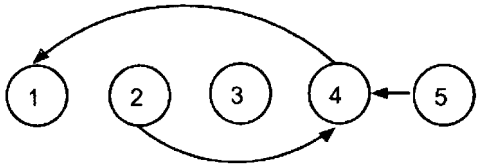


Fig. 2 The subtree of the 1-tree with two components

We can now describe the algorithm which generates all spanning 1-trees:

```
procedure SQL1Trees (Edge )
  for i = 1 to n do (* Initialization *)
    Solution.tree [i] = 0 , Solution.comp [i] = i
  for k = 1 to n − 1 do
    SELECT DISTINCT (* Subgraphs of spanning 1-trees *)
      SetValue (Solution.tree, Edge.u, Edge.v) AS tree
      SetValue (Solution.comp, Edge.u, 1) AS comp
      FROM Solution, Edge
      WHERE
        Solution.comp [Edge.u] ≠ 1 AND Solution.comp
        [Edge.v] = 1
      GROUP BY tree, comp
      TO FILE Solution
```

Edges of the transition graph *G* Tab. 1

u	1	1	2	3	3	3	4	4	4	5	5	6	6
v	2	3	1	1	4	5	2	3	6	3	6	4	5

Before we start the main SQL-algorithm, we calculate the values of *null subtree* (V,∅) with *n* components. One new edge (*u*, *v*) appends in the *k*-step, if it is possible else subtree is deleted. So the subtrees (V, H_k) where $|H_k| = k$ are generated step by step. After the last step we have all spanning 1-trees.

A new subtree and its components are stored in an array *tree* and *comp* which are updated by the function:

```
function x = SetValue(x, i, j)
  x[i] = j
```

Note that after the last step are *comp* = [1, 1, 1, ..., 1] because the spanning 1-trees are connected digraphs.

4. Example of $MMPP_2/M/1/K$ queue

We consider a queueing system studied by Peško [2] with *two-state Markov modulated Poisson process* $MMPP_2$, a single exponential distributed server and a finite waiting room. The problem of switch design and admission control in a high speed network is modeled in [6] as $MMPP_2/GI/1/\infty$ queue. The transition graph for the $MMPP_2/M/1/K$ queue is a weighted digraph

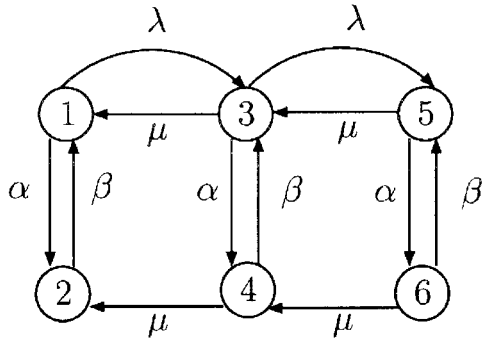


Fig. 3 Transition graph G of $MMPP_2/M/1/3$ queue

$G = (S, E, c)$. In Figure 3 we have the small example of the $MMPP_2/M/1/3$ queue with maximal 2 customers in waiting room where α and β are ON and OFF rate of source, λ is the arrival rate from ON source and μ the service rate.

We have a table *Edge* in the Table 1. The costs of the edges of the transition graph are omitted. Note that the table *Edge* is shown horizontally instead of a usual vertical layout. The initialization table *Solution* is in the first row of the Table 2. After k^{th} step (k = = 1, 2, 3, 4) we have a new *Solution* in Table 2 with added column of step marked k. All spanning 1-trees are generated in the last *Solution* table 3 and described in figure 4. And so B_1 – the sum of weights of all the spanning 1-trees is

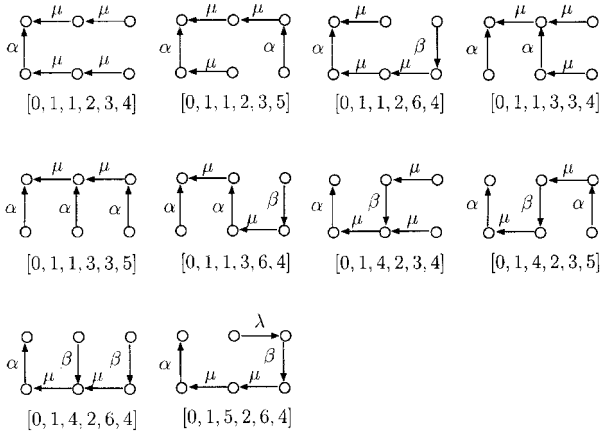


Fig. 4 All spanning 1-trees of G

$$B_1 = \alpha\mu^4 + 2\alpha^2\mu^3 + \alpha^3\mu^2 + 2\alpha\beta\mu^3 +$$

$$+ 2\alpha^2\beta\mu^2 + \alpha\beta^2\mu^2 + \lambda\alpha\beta\mu^2.$$

The sum of weights of all the spanning r-trees B_r for r = 2, ..., 6 can be computed by renumbering set of vertices S.

Solution after k^{th} step — Tab. 2

tree	comp	k
[0, 0, 0, 0, 0, 0]	[1, 2, 3, 4, 5, 6]	–
[0, 1, 0, 0, 0, 0]	[1, 1, 3, 4, 5, 6]	1
[0, 0, 1, 0, 0, 0]	[1, 2, 1, 4, 5, 6]	1
[0, 1, 1, 0, 0, 0]	[1, 1, 1, 4, 5, 6]	2
[0, 1, 0, 2, 0, 0]	[1, 1, 3, 1, 5, 6]	2
[0, 0, 1, 3, 0, 0]	[1, 2, 1, 1, 5, 6]	2
[0, 0, 1, 0, 3, 0]	[1, 2, 1, 4, 1, 6]	2
[0, 1, 1, 2, 0, 0]	[1, 1, 1, 1, 5, 6]	3
[0, 1, 1, 3, 0, 0]	[1, 1, 1, 1, 5, 6]	3
[0, 1, 1, 0, 3, 0]	[1, 1, 1, 4, 1, 6]	3
[0, 1, 4, 2, 0, 0]	[1, 1, 1, 1, 5, 6]	3
[0, 1, 0, 2, 0, 4]	[1, 1, 3, 1, 5, 1]	3
[0, 0, 1, 3, 3, 0]	[1, 2, 1, 1, 1, 6]	3
[0, 0, 1, 3, 0, 4]	[1, 2, 1, 1, 5, 1]	3
[0, 0, 1, 0, 3, 5]	[1, 2, 1, 4, 1, 1]	3
[0, 1, 1, 2, 3, 0]	[1, 1, 1, 1, 1, 6]	4
[0, 1, 1, 2, 0, 4]	[1, 1, 1, 1, 5, 1]	4
[0, 1, 1, 3, 3, 0]	[1, 1, 1, 1, 1, 6]	4
[0, 1, 1, 3, 0, 4]	[1, 1, 1, 1, 5, 1]	4
[0, 1, 1, 0, 3, 5]	[1, 1, 1, 4, 1, 1]	4
[0, 1, 4, 2, 3, 0]	[1, 1, 1, 1, 1, 6]	4
[0, 1, 4, 2, 0, 4]	[1, 1, 1, 1, 5, 1]	4
[0, 1, 0, 2, 6, 4]	[1, 1, 3, 1, 1, 1]	4
[0, 0, 1, 3, 3, 4]	[1, 2, 1, 1, 1, 1]	4
[0, 0, 1, 3, 3, 5]	[1, 2, 1, 1, 1, 1]	4
[0, 0, 1, 3, 6, 4]	[1, 2, 1, 1, 1, 1]	4

Solution after last 5^{th} step — Tab. 3

tree	comp
[0, 1, 1, 2, 3, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 1, 2, 3, 5]	[1, 1, 1, 1, 1, 1]
[0, 1, 1, 2, 6, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 1, 3, 3, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 1, 3, 3, 5]	[1, 1, 1, 1, 1, 1]
[0, 1, 1, 3, 6, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 4, 2, 3, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 4, 2, 3, 5]	[1, 1, 1, 1, 1, 1]
[0, 1, 4, 2, 6, 4]	[1, 1, 1, 1, 1, 1]
[0, 1, 5, 2, 6, 4]	[1, 1, 1, 1, 1, 1]

The research of the autor is supported by the Slovak Scientific Grant Agency under grant No. 1/0490/03.

References:

[1] MARKL, J.: *A Graph Method for Markov Models Solving,* Acta Mathematica et Informatica Universitatis Ostraviensis, Vol. 1, 1993, pp. 75-82

[2] PEŠKO, Š.: *Erlang ON/OFF Moduled Queueing Systems,* Proceedings of the 20'th International Conference, Mathematical Methods in Economics 2002, Ostrava, ISBN 80-248-0153-1, 2002, pp. 209-213

[3] ROSS, S. M.: *Stochastic Processes,* John Wiley & Sons, Inc., 1983, ISBN 0-471-09942-2

[4] KAUKIČ, M.: *Open Source Software in Mathematical Education*, 1. International conference, Aplimat, Bratislava 2002, pp. 233-238

[5] REBO, J., BARTL, O.: *Condition of Stability for Tandem Queues with Blocking and Exponential and Erlangian Service Time Distribution*, Studies of the Faculty of Management Science and Informatics, Vol. 8, Žilina, 1999, pp. 67-74

[6] JEAN-MARIA, A., LIU, Z., NAIN, P., TOWSLEY, D.: *Computational Aspects of the Workload Distribution in the MMPP/GI/1 Queue,* IEEE Journal on Selected Areas in Communications, Vol.16, No.5, 1998, pp.640-652

[7] PALÚCH, S.: *Graph Theory* (*Teória grafov),* Žilinská univerzita, EDIS, 2001, ISBN 80-7100-874-5

[8] MATIAŠKO, K.: *Database Systems (Databázové systémy)*, Žilinská univerzita, EDIS, 2002, ISBN 80-7100-968-7.

kamUNikácie
COMMUNICATIONS

Václav Kořenář *

VEHICLE ROUTING PROBLEM WITH STOCHASTIC DEMANDS

In the introductory section of this paper there is a view of several types of stochastic vehicle routing problems. In the second section of the paper the vehicle routing problem with stochastic demands is described. The following modifications to the vehicle routing problem are required: Customers demand is a random variable with a known probability distribution. Routes must be designed before the actual demands become known.

The presence of nonlinear constraints, caused by random demands, can complicate the solution. Under the specific presumptions it is possible to transfer a nonlinear constraint to a linear form. This transformation enables to use the known algorithms of the deterministic vehicle routing problem for solution of the stochastic vehicle routing problem.

Keywords: *Vehicle routing problem, stochastic demands, artificial capacity, programming system LINGO.*

1. Opening notes

The classical vehicle routing problem (*VRP*) is defined on a graph $G = (V, A)$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of vertices and $A = \{(v_i, v_j): i \neq j, v_i, v_j \in V\}$ is the arcs set. Vertex v_1 represents a depot at which m identical vehicles are based, while the remaining vertices correspond to customers. Matrix $C = (c_{ij})$ is defined on a set A. The coefficients c_{ij} represent distances or travel costs. The count of vehicles can be a given constant or a decision variable. All vehicles have the identical capacity Q. The *VRP* is the problem of constructing m vehicle routes of minimum total cost that each remaining vertex is visited exactly once by one vehicle and satisfying some side constraints.

Types of constraints:

Capacity constraints (each customer v_i has a demand d_i and the total demand of any route may not exceed the vehicle capacity).

Duration constraints (the total length of each route may not exceed a preset constant L).

Time window constraints (each vertex v_i must be visited within a time interval $[a_i, b_i]$).

2. Stochastic vehicle routing problem: An Introduction

Stochastic vehicle routing problems (*SVRP*) arise whenever some elements of the problem are random (e. g. stochastic demand and stochastic travel times).

SVRP differ from the deterministic *VRP* in several fundamental respects. The concept of a solution is different, solution methodologies are considerably more intricate. *SVRP* are often regarded as computationally intractable.

SVRP can be cast within the framework of stochastic programming. Stochastic programs are modeled in two stages. In the first stage a planned solution is determined. In the second stage corrective action is applied to the first stage solution.

Types of stochastic vehicle routing problems:

The vehicle routing problem with stochastic demand is without any doubt the most studied of all *SVRP*. Customer demands are random variables usually assumed to be independent of each other.

The vehicle routing problem with stochastic customers is a direct extension of the traveling salesman problem with stochastic customers. Each customer is represented with some probability but has deterministic demand. The vehicle capacity must be respected and return trips to depot may be necessary whenever it becomes attained.

The vehicle routing problem with stochastic customers and stochastic demands is a combination of the previous models. The definition proposed by Bertsimas [1] seems the most interesting. This vehicle routing problem is an exceedingly difficult problem.

In this paper we study an important variation of the *VRP* in which customer demands are random variables. We consider the situation with unknown demand for each location, but is assumed to follow the known probability distribution.

3. Stochastic vehicle routing problem with stochastic demands

The *SVRP* is at least as difficult as the *VRP*. The presence of nonlinearities in the constraints caused by the probabilistic demands may, in fact, makes it considerably more difficult.

* **Václav Kořenář**
University of Economics Prague, Department of Econometrics, Praha 3, nám. W. Churchilla 4; E-mail: Korenar@vse.cz

The following modifications [2] of the vehicle routing problem are required:
- customers demand is a random variable with a known probability distribution;
- routes must be designed before the actual demands become known.

If the demand at each customer location is a random variable, the problem becomes a stochastic vehicle routing problem and must be treated accordingly.

A mathematical formulation of the *SVRP*:

$$\text{Minimize} \sum_k \sum_{i,j} c_{ij}\, x_{ijk} \tag{1}$$

subject to

$$Prob\left\{\sum_{i,j} d_i\, x_{ijk} \le Q\right\} \ge 1 - \alpha \quad \text{for } k = 1, 2, ..., m, \tag{2}$$

$$x = [x_{ijk}] \in S,$$

where c_{ij} is the distance or cost of travelling from i to j, Q is the vehicle capacity, m is the number of vehicles, S is the set of all m – feasible solutions, $x_{ijk} = 1$ if i and j are joined on route k, 0 otherwise. The demands d_i are random values of the known probability distribution. The parameter α is a probability that in proposed routes the total demand will be greater than the vehicle capacity. Constraint (2) is referred to a chance-constraint.

In the arguments that follow we demonstrate that a chance constraint can be transformed into an equivalent deterministic constraint. An artificial capacity will be used in a *VRP* algorithm in place of the true vehicle capacity. This artificial capacity will be less than the true capacity. Viewing mean customer demands as artificial demands and filling vehicles up to their artificial capacity will perform routing.

Consider a constraint

$$Prob\left\{\sum_i a_i\, y_i \le b\right\} \ge 1 - \alpha \tag{3}$$

If the a_i are independent and identically distributed random variables with the mean $\underline{a_i}$ and standard deviation s_i, and the y_i are decision variables, then the mean is $M = \Sigma \underline{a_i}\, y_i$ and the standard deviation is $S = (\Sigma s_i^2\, y_i^2)^{1/2}$. We seek the constant T such that

$$Prob\left\{\left(\sum_i a_i\, y_i - M\right)/S \le T\right\} = 1 - \alpha. \tag{4}$$

If $b \ge M + TS$, then the left-hand side of inequality (3) is also not less than $1 - \alpha$. Constraint (3) can be replaced by the deterministic constraint

$$M + TS \le b. \tag{5}$$

Constraint (5) is nonlinear. When we rewrite this constraint in terms of y_i, we obtain

$$\sum_i \underline{a_i}\, y_i + T\left(\sum_i s_i^2\, y_i^2\right)^{1/2} \le b \tag{6}$$

The presence of nonlinear constraints caused by the random demands can complicate the solution. Under the specific presumptions it is possible to transfer a nonlinear constraint (6) to a linear constraint. We note that in the context of the *SVRP* the a_i is probabilistic demand and the y_i is 0−1 variable. If M is a mean of customer random demand, S^2 is a variance of random demand and if $S^2 = \lambda M$ for some constant λ, we can transfer a nonlinear constraint (6) to a linear form as

$$\sum_i \underline{a_i}\, y_i + T\left(\lambda \sum_i \underline{a_i}\, y_i\right)^{1/2} \le b \tag{7}$$

or

$$M + T\sqrt{(\lambda M)} \le b \tag{8}$$

We intend to modify this inequality in the simple form

$$M \le \underline{b}\,. \tag{9}$$

Define $w = T\sqrt{\lambda}$, then we obtain from (8)

$$b - M \ge w\, M^{1/2}$$

or

$$(b - M)^2 \ge w^2 M. \tag{10}$$

It follows from (10)

$$M \le [2b + w^2 - \sqrt{(w^4 + 4bw^2)}] \,/\, 2 = \underline{b}\,. \tag{11}$$

If T is determined by $\text{Prob}(z \le T) = 1 - \alpha$, where z is standard normal distributed, we get from (11) the artificial capacity $\underline{b}$. The expression for the artificial capacity like (11) can be formulated also in case of gamma, binomial and Poisson distributions.

4. Illustrative example

As an illustration we solve the vehicle routing problem with random demand of customers, which has Poisson distribution. An expected value of Poisson distribution is equal to its variance and so we can use (11) to compute artificial capacity. And it is satisfied $w = T$, because $\lambda = 1$ for Poisson distribution. Parameter T can be found for a given value α as an quantile of normal distributions in statistical tables.

Let suppose that the true capacity of a vehicle is equal to 30. If we compute (11) with our w and b, we get the artificial capacity and solve the vehicle routing problem as deterministic. Instead of the true capacity vehicles we use the artificial one.

The results of the calculation of artificial capacity $\underline{b}$ using (11) can be found in Table 1.

Table 1

α	T	b
0.9	1.282	23.75
0.95	1.645	22.24
0.99	2.326	19.68

We solved the vehicle routing problem with 10 locations by the system LINGO using branch and bound algorithm.

Summary results are in Table 2:

Table 2

Vehicle capacity	Capacity	Time of calculation (min)	Goal function	Number of iterations
True capacity	30	2:08	114	223 684
Artificial capacity (α = 0,9)	23,75	2:42	117	283 662
Artificial capacity (α = 0,95)	22,24	4:34	122	478 925
Artificial capacity (α = 0,99)	19,68	8:00	127	868 025

References

[1] BERTSIMAS, D. J.: A *Vehicle Routing Problem with Stochastic Demand.* Operations Research, vol. 40, 1992, pp. 574-585.
[2] BODIN, L., GOLDEN, B., ASSAD, A., BELL, M.: *Routing and Scheduling of Vehicles and Crews.* State of Art. Comput. & Ops. Res., Vol. 10, No 2, 1983, pp. 109-111.
[3] GENDREAU, M., LAPORTE, G., SÉGUIN, R.: *Stochastic Vehicle Routing.* European Journal of Operational Res., Vol. 88, 1996, pp. 3-12.
[4] KOŘENÁŘ, V., VÍŠEK, T.: *Branch and Cut Algorithm for Routing Problem.* In: Plevný, M., Friedrich, V. (ed.). Mathematical Methods in Economics. Plzeň, Západočeská univerzita, 1998, s. 65-72. ISBN 80-7082-492-1.
[5] Solver Suite - LINDO, LINGO, *What's Best.* Lindo System, Chicago 1995.

Jaroslav Janáček – Juraj Hurtík *

AN IMPACT OF IMPROVEMENT-EXCHANGE HEURISTICS TO QUALITY OF PROBABILISTIC TSP SOLUTION

This paper deals with a probabilistic travelling salesman problem (PTSP), which differs from a travelling salesman problem (TSP) [6] in the demand for a customer visit. In PTSP is each customer visited with a given probability only. An objective function for PTSP is in general hard to enumerate and it is even harder to estimate an influence of a local change of a PTSP solution on the final value of the objective function of the resulting solution. This hardiness complicates construction of an efficient heuristic for this problem. In this contribution we have focused on research of a relation between type of local change operation and the resulting objective function improvement. We have constrained here ourselves to improvement-exchange heuristics only, whereas inserting heuristics were broadly studied in the previous works [3], [5]. In this work we have tried to suggest several types of feasible solution changes (operations) including estimation of their impact on the objective function value. These operations have been embedded into the best admissible strategy and implemented as a part of software system, which enables exact or approximate evaluation of a PTSP route depending on the problem size. Using this system we have performed a sequence of numerical experiments to reveal relation between complexity of the used heuristics operations and final improvement of the resulting objective function value. This numerical result and associated conclusions are reported in the concluding part of this contribution.

1. Introduction

The probabilistic travelling salesman problem (PTSP) is a generalization of a well-known travelling salesman problem (TSP). Each node in PTSP has to be visited with some probability and visit necessities of all these nodes are stochastically independent. This problem was first introduced by Jaillet [2].

Let us denote the probability of the visit at node i by p_i. An actual visit of the node i is represented by a random binary variable $N_i \in [0, 1]$, for which the value 1 that appears with probability p_i, means a visit of node and the value 0 (with $(1-p_i)$ probability) means that the node is not to be visited. The distance between the nodes i, j is denoted by d_{ij}.

As we can see, the original TSP is just a special case of the problem presented above, where visit necessity probabilities of all nodes are set to 1. This difference makes a PTSP more difficult to solve, as it is ineffective to find a route every day according to the current set of customers (all nodes i with the current value $N_i = 1$). Thus, we need to find a solution for TSP that respects probabilities of all customers and an expected route length of, which, after skipping inactive customers, will be minimal. The main problem is to find the mentioned solution, which is called an a priori route.

In accordance with the above-mentioned problem we denote $\{1, 2, \ldots n\}$ set of customers, which should be visited whenever they are active (when their random variable value is 1). Each customer j raises his demand for visiting with probability p_j. Our goal is to determine such a sequence of all customers so that the expected length of the tour be minimal. An instance of the tour arises by skipping customers that are not active in the given instance from the sequence.

When we fix the position of one customer in the considered sequence, then the sequence $\{j(1), j(2), \ldots j(n)\}$ is uniquely determined by the solution.

The evaluation of objective function constitutes a key problem of PTSP modelling. As we mentioned in the first part of this contribution, the constants d_{ij} represent distances between customers. Let us denote $I = \{i(1), i(2), \ldots i(|I|)\}$ as increasing-way ordered set of indexes of all active customers from the sequence. The probability of the sequence given by the sequence I is:

$$\prod_{i \in I} p_{j(i)} \prod_{i \in N-I} (1 - p_{j(i)}) \tag{1}$$

The corresponding length of the route is then:

$$d_{j(i(|I|))j(i(1))} + \sum_{k=1}^{|I|-1} d_{j(i(k))j(i(k+1))} \tag{2}$$

From (1) and (2), we can write the expected value of route length for general sequence $j(1), j(2), \ldots j(n)$ as:

$$\sum_{I \subseteq N} \left(d_{j(i(|I|))j(i(1))} + \sum_{k=1}^{|I|-1} d_{j(i(k))j(i(k+1))} \right) \cdot \cdot \prod_{i \in I} p_{j(i)} \prod_{i \in N-I} (1 - p_{j(i)}) . \tag{3}$$

The expression (3) is very hard to enumerate because the embedded sum goes over all subsets I of the set N and it means that 2^n items have to be processed. This complicated objective function evaluation constitutes a serious problem when an opera-

* **Jaroslav Janáček, Juraj Hurtík**
University of Žilina, Faculty of Management Science and Informatics, E-mail: jardo@frdsa.fri.utc.sk, jurhur@student.dorm.utc.sk

tion changing a given solution is suggested and the associated change of the objective function should be estimated. This problem, as concerned heuristics starting with a feasible solution, does not arise, when a classical TSP is solved.

2. Improvement-exchange operations for TSP

The improvement-exchange operations for the probabilistic travelling salesman problem were derived from the basic improvement-exchange operations for the classical travelling salesman problem. We have considered two types of the operations and both of them will process so called chains of a current TSP route.

The TSP route will be given by sequence $J = < J(1), J(2), ..., J(n), J(1)>$ of n network nodes, which should be visited in the given order. A chain of this route is an arbitrary subsequence of the sequence J. Let us assume that only such chains will be taken into consideration, which do not contain the node $J(1)$. Then a chain may be determined by index i_p of its predecessor $P = J(i_p)$ and by index i_s of its successor $S = J(i_s)$. The very chain is formed by the following sequence of the nodes $J(i_p + 1), J(i_p + 2), J(i_p + 3), ..., J(i_s - 1)$. The first node $B = J(i_p + 1)$ and the last one $E = J(i_s - 1)$ of the chain are referred as the beginning and the end of the chain respectively. If P and S are neighbouring nodes, then the associated chain is said to be empty what means that it has no node. The number of nodes of the chain is referred to as length of the chain. Then the chain can be determined by index i_p if the chain predecessor and by chain length L. In this case, the following equalities hold: $P = J(i_p)$, $S = J(i_p + L + 1)$,. $i_s = i_p + L + 1$ and under assumption $L > 0$, the additional equalities hold $B = J(i_p + 1)$, $i_b = i_p + 1$, $E = J(i_p - L)$, $i_e = i_p + L = i_s - 1$.

The first type of the suggested operations is a unary operation defined on a set of chains of J, the length of which is greater or equal to 2. This unary operation is called inversion and consists of taking the chain off the sequence J and inserting it back at the same position in the reversed order. If the current solution is given by the sequence $J = <J(1), J(2), ..., J(i_p), J(i^b), J(i_b + 1), ..., J(i_e - 1), J(i_e), J(i_s), ..., J(n), J(1)>$, then, after the inversion of the chain $C = <J(i_b), J(i_b + 1),..., J(i_e - 1), J(i_e)>$, the following solution arises $J = <J(1), J(2), ..., J(i_p), J(i_e), J(i_e - 1), ..., J(i_b + 1), J(i_b), J(i_s), ..., J(n), J(1)>$ (Fig. 1). Whereas the travelled distance of the current solution is $d = \sum_{i=1}^{n} d_{J(i),\, J(i+1)}$, the travelled distance of the new solution is

$$\underline{d} = \sum_{i=1}^{i_p-1} d_{J(i),\, J(i+1)} + d_{J(i_p),\, J(i_e)} + \sum_{i=i_b}^{i_e-1} d_{J(i+1),\, J(i)} + d_{J(i_b),\, J(i_s)} + \sum_{i=i_s}^{n} d_{J(i),\, J(i+1)}.$$

The improvement of the classical travelling salesman problem provided by this operation can be expressed by the following difference of the arc lengths

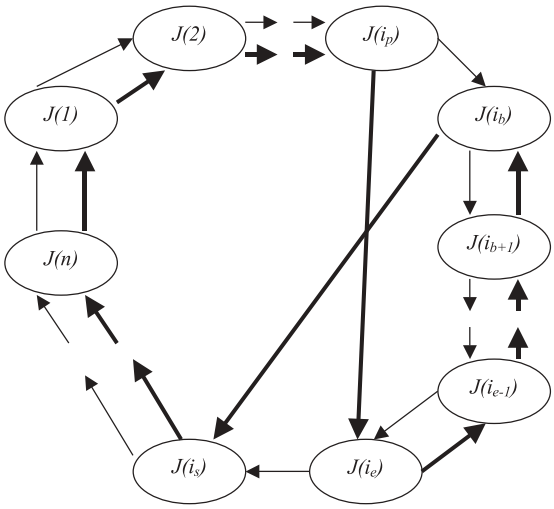


Fig. 1: Improvement of TSP via unary operation

$$d - \underline{d} = d_{J(i_p),\, J(i_b)} + d_{J(i_e),\, J(i_s)} - d_{J(i_p),\, J(i_e)} - d_{J(i_b),\, J(i_s)} + \left[\sum_{i=i_b}^{i_e-1} d_{J(i),\, J(i+1)} - \sum_{i=i_b}^{i_e-1} d_{J(i+1),\, J(i)} \right]. \quad (4)$$

If a symmetrical network is considered, where $d_{ij} = d_{ji}$ holds for arbitrary pair of indices i, j, then the last term of (4) is zero and the improvement is

$$d - \underline{d} = d_{J(i_p),\, J(i_b)} + d_{J(i_e),\, J(i_s)} - d_{J(i_p),\, J(i_e)} - d_{J(i_b),\, J(i_s)} =$$

$$= d_{PB} + d_{ES} - d_{PE} - d_{BS}.$$

The second type of the suggested operations is a binary operation defined on pairs of disjoint chains of J. Two chains of J are said to be disjoint, if they do not share any node nor the predecessor or successor of the other chain. It is only allowed that the successor of one chain is the predecessor of the other one. The further described binary operation is called mutual exchange of two chains and consists of removing both chains from their positions and of inserting them at the opposite positions.

We shall distinguish two cases of chain exchanging operations. In the first case one of these chains is considered to be empty and in the second case the lengths of both chains are greater than zero.

Taking into account the first case, let us study the situation, when the inserted chain must follow its original direction of the travelling salesman route. We denote i_{p1}, i_{b1}, $ie1$ and i_{s1} the indices of the first non-empty chain and i_{p2}, i_{s2} the indices of the second empty chain predecessor and successor. If the current solution J has the form of $J = <J(1), J(2), ..., J(i_{p1}), J(i_{b1}), ..., J(i_{e1}), J(i_{s1}), ..., J(i_{p2}), J(i_{s2}), ..., J(n), J(1)>$, then after the exchange operation the following route comes into being $J = <J(1), J(2), ..., J(i_{p1}), J(i_{s1}), ..., J(i_{p2}), J(i_{b1}), ..., J(i_{e1}), J(i_{s2}), ..., J(n), J(1)>$ (Fig. 2).

The possible improvement can be described by

$$d - \underline{d} = d_{J(i_{p1}),\,J(i_{b1})} + d_{J(i_{e1}),\,J(i_{s1})} - d_{J(i_{p2}),\,J(i_{e2})} -$$

$$- d_{J(i_{p1}),\,J(i_{s1})} - d_{J(i_{p2}),\,J(i_{b1})} - d_{J(i_{e1}),\,J(i_{s2})} =$$

$$= d_{P1B1} + d_{E1S1} + d_{P2S2} - d_{P1S1} - d_{P2B1} - d_{E1S2}\,.$$

where $P_1 = J(i_{p1})$, $B_1 = J(i_{b1})$, $E_1 = J(i_{e1})$, $S_1 = J(i_{s1})$, $P_2 = J(i_{p2})$ and $S_2 = J(i_{s2})$.

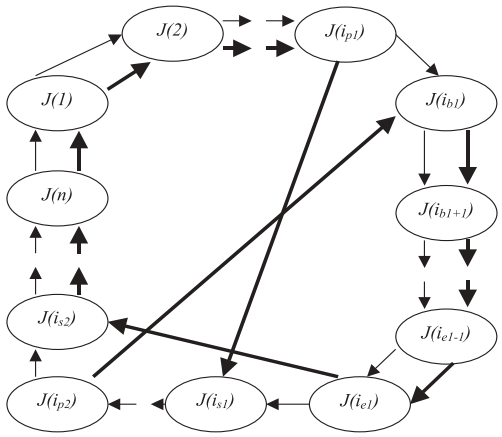


Fig. 2: Improvement of TSP via binary operation

If the non-empty chain is inserted at its new position in the reversed order and the distance matrix is symmetrical, then the associated improvement is

$$d - \underline{d} = d_{P1B1} + d_{E1S1} + d_{P2S2} - d_{P1S1} - d_{P2E1} - d_{B1S2}\,.$$

Considering the second case for a symmetrical distance matrix, we have to distinguish in general four possibilities, how to insert the exchanged chains at their new positions.

The basic possibility is, to let both newly inserted chains follow their former directions from the original route. Then we obtain from the original sequence $J = <J(1), J(2), ..., J(i_{p1}), J(i_{b1}), ..., J(i_{e1}), J(i_{s1}), ..., J(i_{p2}), J(i_{b2}), ..., J(i_{e2}), J(i_{s2}), ..., J(n), J(1)>$, this one: $\underline{J} = <J(1), J(2), ..., J(i_{p1}), J(i_{b2}), ..., J(i_{e2}), J(i_{s1}), ..., J(i_{p2}), J(i_{b1}), ..., J(i_{e1}), J(i_{s2}), ..., J(n), J(1)>$. and the associated improvement is

$$d - \underline{d} = d_{P1B1} + d_{E1S1} + d_{P2B2} + d_{E2S2} - d_{P1B2} -$$

$$- d_{E2S1} - d_{P2B1} - d_{E1S2}.$$

Further possibility is to make inversion of the first chain and to let the second chain follow its former orientation. Then we obtain from the original sequence J the new sequence $\underline{J} = <J(1), J(2), ..., J(i_{p1}), J(i_{b2}), ..., J(i_{e2}), J(i_{s1}), ..., J(i_{p2}), J(i_{e1}), ..., J(i_{b1}), J(i_{s2}), ..., J(n), J(1)>$. with the associated improvement

$$d - \underline{d} = d_{P1B1} + d_{E1S1} + d_{P2B2} + d_{E2S2} - d_{P1B2} -$$

$$- d_{E2S1} - d_{P2E1} - d_{B1S2}.$$

The third possibility lets the first chain follow its former orientation and makes inversion of the second. This exchange leads to $\underline{J} = <J(1), J(2), ..., J(i_{p1}), J(i_{e2}), ..., J(i_{b2}), J(i_{s1}), ..., J(i_{p2}), J(i_{b1}), ..., J(i_{e1}), J(i_{s2}), ..., J(n), J(1)>$. with

$$d - \underline{d} = d_{P1B1} + d_{E1S1} + d_{P2B2} + d_{E2S2} - d_{P1E2} -$$

$$- d_{B2S1} - d_{P2B1} - d_{B1S2}.$$

The last possibility consists of inversion of both chains and of exchange of their positions. This way the following result is obtained from J

$\underline{J} = <J(1), J(2), ..., J(i_{p1}), J(i_{e2}), ..., J(i_{b2}), J(i_{s1}), ..., J(i_{p2}), J(i_{e1}), ..., J(i_{b1}), J(i_{s2}), ..., J(n), J(1)>$. with the associated improvement

$$d - \underline{d} = d_{P1B1} + d_{E1S1} + d_{P2B2} + d_{E2S2} - d_{P1E2} -$$

$$- d_{B2S1} - d_{P2E1} - d_{B1S2}.$$

Considering the above-mentioned operations for the classical travelling salesman problem, the change of the objective function value caused by the particular operation is immediately known.

Unfortunately, we can hardly estimate the impact of the individual improvement to the resulting objective function value. In fact, such a situation can occur that application of the most advantageous operation will prevent the algorithm from following some sequence of less advantageous operations, which could reach a very good solution.

3. Strategies for PTSP Heuristics

Considering the classical travelling salesman problem, at least the local impact of an improvement-exchange operation on the objective function value of the improved solution can be enumerated exactly and fast enough. The similar approach to probabilistic TSP is almost impossible due complexity of the objective function. We can do here some conditioned assumptions and study the particular strategy impact on the total improvement of the objective function of the tested problems. The problem originates from the fact that the shortening of the TSP route need not bring any saving in the expected value of PTSP route length. Nevertheless we can judge that the improvement described in the previous section could take part on the PTSP objective function improvement proportionally to the probability, with which the employed nodes occur in the PTSP route.

The classical strategies of combinatorial heuristics differ in the way in which a so-called neighbourhood of a current solution is processed. The neighbourhood of some solution is defined as the set of solutions, which can be reached from the current one using only one operation from a set of permitted operations.

For example, if inversion of chain of length two is the only permitted operation, then the neighbourhood of a current solution

is formed by all the results which can be obtained by inversion of one chain of length equal to two in the current solution.

There are two basic strategies how to process a current solution neighbourhood. They are the first and best admissible strategies. The strategy "First Admissible" (FA) tests the permitted operations on the current solution in a given order and whenever it finds the operation with positive improvement (admissible operation or move), it performs the associated move to the better solution and updates the current solution. This process is repeated until such a current solution is met where no admissible operation exists. The strategy "Best Admissible" (BA) searches through the whole neighbourhood and updates the best admissible operation, i.e. the operation with the greatest positive improvement. After the neighbourhood searching is completed, the best admissible operation is performed and the current solution is updated. This cycle is repeated until no admissible operation on the current solution is found.

To design PTSP strategy, we refused the first admissible one, because it does not allow us to employ the probabilities of the nodes associated with the given operation. The first admissible strategy performs the first admissible move without comparing it to the others.

That is why we decided to use the best admissible strategy which tests all permitted operations on the current solution and computes the associated improvement as shown in the previous section. To employ the visit probabilities of the nodes which are associated with the considered operation, we expressed the probability that the substantial nodes occur in the necessary configuration.

For example, if the inversion of the chain with predecessor P, beginning B, end E and successor S should be performed, it is necessary for them to be visited in the given instance. The probability that all the nodes will be visited simultaneously is $p_{in} = p_P * p_B * p_E * p_S$.

The best admissible strategy enables to weigh each improvement associated with operation by the associated probability of the node configuration and to select the best expected improvement and the associated operation. In the suggested improvement-exchange heuristic, we assigned to operation of inversion the above-mentioned probability $p_{in} = p_P * p_B * p_E * p_S$.

When chain exchanges are considered, then if the first chain has its length equal to one and the second one has its length equal to zero, then $p_{ex} = p_{P1} * p_{B1} * p_{S1} * p_{P2} * p_{S2}$.

If the first chain has the length greater than one, then $p_{ex} = p_{P1} * p_{B1} * p_{E1} * p_{S1} * p_{P2} * p_{S2}$. If the both chains have their lengths equal to one, then $p_{ex} = p_{P1} * p_{B1} * p_{S1} * p_{P2} * p_{B2} * p_{S2}$ and if both chains have their lengths greater than one, then $p_{ex} = p_{P1} * p_{B1} * p_{E1} * p_{S1} * p_{P2} * p_{B2} * p_{E2} * p_{S2}$.

4. Numerical Experiments

To verify and evaluate our improvement-exchange heuristics, we have decided to study their behaviour on the set of sixty PTSP problems, which were derived from the problems referred in [1]. This set of benchmark problems consists of six subsets, where each subset contains ten problems of the same size. In the subsets, problem sizes of 20, 30, 50, 100, 150 and 250 nodes were considered. The associated subsets were denoted by M020, M030, M050, M100, M150 and M200 respectively. As the improvement-exchange heuristic needs a starting solution and quality of this solution may influence the contribution of the tested heuristic, we used two different dual heuristics to obtain various starting solutions. The improvement phase was performed with each of these solutions. The dual heuristics used to obtain the starting solution were described and studied in [3] and [4]. This set of dual heuristics contains the so-called Pelikan's algorithm [5] and Repeated Nearest Neighbour algorithm [3], which are denoted in the following tables as "Pelikan" and "RNN" respectively. As we have decided to explore the dependence of solution quality and time consumption on the complexity of the algorithm, which searches over current solution neighbourhood, we defined the order of improvement-exchange heuristic for this purpose. This order equals to the maximal length of chains, which is processed by the particular heuristic.

This way, the above mentioned algorithm performing only inversions of chains of length equal to two is denoted as heuristic of order $o = 2$. If an inversion heuristic of order three is considered, then each inversion of chain the length of which equals to two or three, must be inspected to process neighbourhood of a current solution. If the chain exchange algorithm is considered, then heuristic of order two must evaluate each pair of chains, the lengths of which are one – zero, two – zero, one – one, two – one, two – two, to select the best move, which should be accomplished to leave the current solution.

To evaluate the objective function value (expected length according to expression (3)), we employed two approaches, enumeration and simulation [3], [4]. For the problems of a smaller size, we suggested and implemented a straightforward enumeration algorithm. This algorithm is based on a last-in-first-out structure and the enumeration scheme for the probability and length of an instance enumeration, which resembles a complete depth-searching process in a branching tree. In the enumeration process we start with an instance in that all customers are active, then the instance is processed in which all customers but the last are active, further the instance with only last but one active customer comes, and as the next case, the instance will be studied with two last inactive customers etc.

A faster way how to estimate the expected length of the route is the use of simulation technique. This method consists of a performed random generation of individual instances and their probabilistic evaluation. As we have found [3] that the enumeration process has exponential time consumption depending on a problem size, the simulation was the only method, which can settle with objective function of solutions containing more than thirty nodes.

In the following tables the average time consumption of improvement exchange heuristic is given in seconds for the particular subset of problems. The numerical experiments were performed on PC P4 1400 MHz, 128 MB RAM.

The solution quality is described by the average value of starting solutions and of the average improvement. This improvement is given once more in the tables in percentage, where the average starting solution is taken as a base (100 %).

Table 1

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M020	Pelikan	778.48	Inversion	2	-1.62	-0.21	0.00
Enumeration	Pelikan	778.54	Inversion	2	-1.59	-0.20	0.00
M020	Pelikan	778.48	Inversion	3	-1.53	-0.19	0.00
Enumeration	Pelikan	778.54	Inversion	3	-1.58	-0.20	0.00
M020	Pelikan	778.48	Inversion	4	-1.58	-0.23	0.00
Enumeration	Pelikan	778.54	Inversion	4	-1.80	-0.20	0.00
M020	Pelikan	778.48	Inversion	5	-1.55	-0.20	0.00
Enumeration	Pelikan	778.54	Inversion	5	-1.56	0.24	0.00
M020	Pelikan	778.48	Exchange	1	1.89	0.24	0.00
Enumeration	Pelikan	778.54	Exchange	1	1.90	0.78	0.00
M020	Pelikan	778.48	Exchange	2	6.04	0.77	0.00
Enumeration	Pelikan	778.54	Exchange	2	6.00	0.76	0.00
M020	Pelikan	778.48	Exchange	3	5.94	0.77	0.00
Enumeration	Pelikan	778.54	Exchange	3	6.00	0.76	0.00
M020	Pelikan	778.48	Exchange	4	5.88	0.76	0.00
Enumeration	Pelikan	778.54	Exchange	4	6.00	0.77	0.00
M020	Pelikan	778.48	Exchange	5	6.00	0.77	0.00
Enumeration	Pelikan	778.54	Exchange	5	6.00	0.77	0.00

Table 2

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M020	RNN	918.08	Inversion	2	16.36	1.78	0.00
Enumeration	RNN	918.51	Inversion	2	16.48	1.79	0.00
M020	RNN	918.08	Inversion	3	33.27	3.62	0.00
Enumeration	RNN	918.51	Inversion	3	33.33	3.63	0.00
M020	RNN	918.08	Inversion	4	51.59	5.62	0.00
Enumeration	RNN	918.51	Inversion	4	51.77	5.64	0.00
M020	RNN	918.08	Inversion	5	55.51	6.04	0.00
Enumeration	RNN	918.51	Inversion	5	55.67	6.06	0.00
M020	RNN	918.08	Exchange	1	100.31	10.92	0.00
Enumeration	RNN	918.51	Exchange	1	100.43	10.93	0.00
M020	RNN	918.08	Exchange	2	134.34	14.63	0.00
Enumeration	RNN	918.51	Exchange	2	134.50	14.93	0.00
M020	RNN	918.08	Exchange	3	137.14	14.94	0.00
Enumeration	RNN	918.51	Exchange	3	137.23	15.73	0.00
M020	RNN	918.08	Exchange	4	144.50	15.75	0.00
Enumeration	RNN	918.51	Exchange	4	144.66	15.74	0.00
M020	RNN	918.08	Exchange	5	144.54	15.75	0.00
Enumeration	RNN	918.51	Exchange	5	144.66	15.74	0.00

Table 3

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M030	Pelikan	1014.61	Inversion	2	0.51	0.05	0.00
Enumeration	Pelikan	1014.57	Inversion	2	0.42	0.04	0.00
M030	Pelikan	1014.61	Inversion	3	-0.77	-0.08	0.00
Enumeration	Pelikan	1014.57	Inversion	3	-0.72	-0.07	0.00
M030	Pelikan	1014.61	Inversion	4	-0.16	-0.02	0.00
Enumeration	Pelikan	1014.57	Inversion	4	-0.30	-0.03	0.00
M030	Pelikan	1014.61	Inversion	5	-0.24	-0.02	0.00
Enumeration	Pelikan	1014.57	Inversion	5	-0.32	-0.03	0.00
M030	Pelikan	1014.61	Exchange	1	-2.63	-0.26	0.00
Enumeration	Pelikan	1014.57	Exchange	1	-2.61	-0.26	0.00
M030	Pelikan	1014.61	Exchange	2	-15.76	-1.57	0.00
Enumeration	Pelikan	1014.57	Exchange	2	-15.92	-1.57	0.00
M030	Pelikan	1014.61	Exchange	3	-16.20	-1.60	0.00
Enumeration	Pelikan	1014.57	Exchange	3	-16.27	-1.60	0.00
M030	Pelikan	1014.61	Exchange	4	-7.42	-0.73	0.01
Enumeration	Pelikan	1014.57	Exchange	4	-7.57	-0.75	0.00
M030	Pelikan	1014.61	Exchange	5	-5.66	-0.56	0.01
Enumeration	Pelikan	1014.57	Exchange	5	-5.82	-0.57	0.00

Table 4

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M030	RNN	1330.74	Inversion	2	16.49	1.24	0.00
Enumeration	RNN	1330.93	Inversion	2	16.52	1.24	0.00
M030	RNN	1330.74	Inversion	3	18.52	1.39	0.00
Enumeration	RNN	1330.93	Inversion	3	18.68	1.40	0.00
M030	RNN	1330.74	Inversion	4	31.98	2.40	0.00
Enumeration	RNN	1330.93	Inversion	4	31.72	2.38	0.00
M030	RNN	1330.74	Inversion	5	64.04	4.81	0.00
Enumeration	RNN	1330.93	Inversion	5	64.10	4.82	0.00
M030	RNN	1330.74	Exchange	1	113.23	8.51	0.00
Enumeration	RNN	1330.93	Exchange	1	113.48	8.53	0.00
M030	RNN	1330.74	Exchange	2	226.74	17.04	0.01
Enumeration	RNN	1330.93	Exchange	2	226.92	17.05	0.01
M030	RNN	1330.74	Exchange	3	280.20	21.06	0.02
Enumeration	RNN	1330.93	Exchange	3	280.24	21.07	0.02
M030	RNN	1330.74	Exchange	4	285.11	21.53	0.02
Enumeration	RNN	1330.93	Exchange	4	285.28	21.52	0.02
M030	RNN	1330.74	Exchange	5	292.71	22.00	0.03
Enumeration	RNN	1330.93	Exchange	5	292.96	22.01	0.03

Table 5

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M050	Pelikan	9098.88	Inversion	2	15.34	0.14	0.00
M050	Pelikan	9098.88	Inversion	3	-12.28	-0.13	0.00
M050	Pelikan	9098.88	Inversion	4	-9.24	-0.10	0.00
M050	Pelikan	9098.88	Inversion	5	-20.79	-0.22	0.00
M050	Pelikan	9098.88	Exchange	1	-3.83	-0.04	0.00
M050	Pelikan	9098.88	Exchange	2	-98.68	-1.08	0.01
M050	Pelikan	9098.88	Exchange	3	-89.73	-0.99	0.02
M050	Pelikan	9098.88	Exchange	4	-39.99	-0.44	0.03
M050	Pelikan	9098.88	Exchange	5	30.40	0.33	0.05
M050	RNN	13967.38	Inversion	2	94.63	0.67	0.00
M050	RNN	13967.38	Inversion	3	196.94	1.41	0.00
M050	RNN	13967.38	Inversion	4	284.12	2.06	0.00
M050	RNN	13967.38	Inversion	5	357.90	2.56	0.00
M050	RNN	13967.38	Exchange	1	2433.36	14.40	0.02
M050	RNN	13967.38	Exchange	2	3451.34	24.68	0.06
M050	RNN	13967.38	Exchange	3	3868.67	27.66	0.11
M050	RNN	13967.38	Exchange	4	4162.85	29.77	0.18
M050	RNN	13967.38	Exchange	5	4577.41	32.73	0.26

Table 6

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M100	Pelikan	11291.04	Inversion	2	38.29	0.33	0.00
M100	Pelikan	11291.04	Inversion	3	139.75	1.23	0.00
M100	Pelikan	11291.04	Inversion	4	110.33	0.97	0.00
M100	Pelikan	11291.04	Inversion	5	437.66	3.25	0.00
M100	Pelikan	11291.04	Exchange	1	200.93	1.77	0.02
M100	Pelikan	11291.04	Exchange	2	279.40	2.47	0.07
M100	Pelikan	11291.04	Exchange	3	330.47	2.92	0.18
M100	Pelikan	11291.04	Exchange	4	310.40	2.74	0.28
M100	Pelikan	11291.04	Exchange	5	309.22	2.73	0.43
M100	RNN	17501.28	Inversion	2	318.64	1.82	0.00
M100	RNN	17501.28	Inversion	3	562.79	3.21	0.00
M100	RNN	17501.28	Inversion	4	829.97	4.71	0.00
M100	RNN	17501.28	Inversion	5	1038.6	5.93	0.00
M100	RNN	17501.28	Exchange	1	3447.5	19.69	0.15
M100	RNN	17501.28	Exchange	2	4473.37	25.77	0.41
M100	RNN	17501.28	Exchange	3	4831.20	27.60	0.76
M100	RNN	17501.28	Exchange	4	5128.79	29.30	1.09
M100	RNN	17501.28	Exchange	5	5197.54	29.69	1.59

Table 7

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M150	Pelikan	28194.35	Inversion	2	12.23	0.04	0.01
M150	Pelikan	28194.35	Inversion	3	20.59	0.07	0.00
M150	Pelikan	28194.35	Inversion	4	40.97	0.136	0.00
M150	Pelikan	28194.35	Inversion	5	30.35	0.10	0.00
M150	Pelikan	28194.35	Exchange	1	287.88	1.02	0.08
M150	Pelikan	28194.35	Exchange	2	430.04	1.52	0.31
M150	Pelikan	28194.35	Exchange	3	454.66	1.61	0.59
M150	Pelikan	28194.35	Exchange	4	470.26	1.65	0.91
M150	Pelikan	28194.35	Exchange	5	475.27	1.68	1.45
M150	RNN	45080.75	Inversion	2	827.10	1.83	0.00
M150	RNN	45080.75	Inversion	3	1146.54	2.54	0.00
M150	RNN	45080.75	Inversion	4	1493.51	3.31	0.00
M150	RNN	45080.75	Inversion	5	1791.81	4.13	0.00
M150	RNN	45080.75	Exchange	1	10578.41	23.46	0.54
M150	RNN	45080.75	Exchange	2	12715.01	28.20	1.39
M150	RNN	45080.75	Exchange	3	13964.41	30.97	2.48
M150	RNN	45080.75	Exchange	4	14953.12	33.16	4.07
M150	RNN	45080.75	Exchange	5	15076.52	33.44	5.71

Table 8

Problem	Dual heuristic	Average starting value	Primal heuristic	Order	Average improvement	Percentage	Average time [s]
M250	Pelikan	11637.91	Inversion	2	4.21	0.03	0.00
M250	Pelikan	11637.91	Inversion	3	-3.21	-0.02	0.00
M250	Pelikan	11637.91	Inversion	4	13.60	0.11	0.00
M250	Pelikan	11637.91	Inversion	5	25.36	0.21	0.00
M250	Pelikan	11637.91	Exchange	1	118.83	1.02	3.96
M250	Pelikan	11637.91	Exchange	2	150.52	1.29	4.02
M250	Pelikan	11637.91	Exchange	3	231.43	1.98	3.60
M250	Pelikan	11637.91	Exchange	4	263.78	2.27	5.62
M250	Pelikan	11637.91	Exchange	5	288.40	2.47	8.04
M250	RNN	17824.70	Inversion	2	321.80	1.80	0.00
M250	RNN	17824.70	Inversion	3	455.90	2.55	0.00
M250	RNN	17824.70	Inversion	4	681.64	3.82	0.00
M250	RNN	17824.70	Inversion	5	884.54	4.96	0.00
M250	RNN	17824.70	Exchange	1	3352.14	18.80	4.35
M250	RNN	17824.70	Exchange	2	3980.14	22.32	10.87
M250	RNN	17824.70	Exchange	3	4732.49	26.11	15.52
M250	RNN	17824.70	Exchange	4	4820.04	27.04	24.89
M250	RNN	17824.70	Exchange	5	4883.50	27.39	33.70

As can be seen in the Tables 1–4 in the column "Average starting value", the results obtained by simulation and enumeration processes are almost identical. It supports our hypothesis that the simulation process can be used for evaluation of results in the cases, when the enumeration process fails due to time complexity.

From the comparison of the results obtained by a two-phase procedure Pelikan – improvement exchange and the results obtained by Nearest-Neighbour – improvement exchange, it can be found that the Pelikan's dual heuristic provides very good results which can not be improved by the studied primal heuristics. It can be noticed that this improvement exchange heuristic may deteriorate the input solution to some small extent. On the contrary to the procedure Pelikan – improvement exchange, the improvement exchange heuristic approved its use when a less quality input solution produced by the Nearest-Neighbour algorithm is used as a starting solution.

As regards of larger size from 100 to 250 customers (Tables 6–8), a small improvement from 1 to 2 % can be found even when improvement – exchange heuristic is used to Pelikan's resulting solution. Comparing it with the results obtained in the experiments with the Repeated Nearest-Neighbour procedure, it can be seen that the improvement ranges from 2 to 30 % in these cases.

5. Conclusions

The improvement – exchange heuristic, whose advantage was tested in the above mentioned experiments, turned out to be very useful when a good solution of the probabilistic travelling salesman problem is sought. Especially in the larger instances when the starting solution is obtained by common dual heuristic, there is employment of an improvement – exchange heuristic unavoidable. Furthermore, the experiments showed a considerable increase of the obtained improvement connected with raising order of the heuristic, especially when chain exchange is performed instead of simple inversion.

Similar to our previous results [3], [4], the Pelikan's heuristic proved to be able to provide excellent solutions which are hard to improve and which seem to be a local minima with respect to the used exchange operations. We think that it is caused by the fact that our exchange - improvement heuristic took into consideration only probability of the inspected improvement, which is evaluated using only the edges between neighbouring nodes in a current route. This approach does not explore the mutual positions of the customers with higher probability, which are separated by customers with lower probability in the current route.

In contradiction to this insufficiency the Pelikan's heuristic inserts customers to a route in accordance to their decreasing probability and at each step it locates a current customer into the current route, in which each customer has greater probability than the current one. We can suggest the study of improvement – exchange heuristic overcoming above-mentioned insufficiency for a future research.

Acknowledgement
This paper is supported by the grant VEGA 1/0498/03.

References

[1] ftp://ftp.zib.de/pub/Packages/mp-testdata/tsp/tsplib/., internet server with TSP benchmarks
[2] JAILLET, P.: *A priori solution of the traveling salesman problem in which a random subset of customers are visited.*, Operation Research 36, 1988
[3] JANÁČEK, J., HURTÍK, J.: *Heuristics algorithm and evaluation of probabilistic traveling salesman problem solution.* Journal of Information, Control and Management Systems, ŽU Žilina, 2003, to appear
[4] JANÁČEK, J., HURTÍK, J.: *Simulation and evaluation of a probabilistic traveling salesman problem.* In: Proceedings of the conference Transcom 2003, Žilina
[5] PELIKÁN, J.: *A modified insert algorithm for the probabilistic traveling salesman problem.* In: Proceedings of the 8th International Scientific Conference Quantitative Methods in Economy and Business-Methodology and Practice in the New Millenium, Bratislava September 18-20, 2002, pp 634-638
[6] PEŠKO, Š., HURTÍK, J.: *Solving TSP via SQL queries.* In: Proceedings of the conference Quantitative Methods in Economics (Multiple Criteria Decision Making XI), Nitra December 5-6, 2002, pp 194-197
[7] PEŠKO, Š.: *The pyramid method for traveling salesman problem (in slovak).* Communications, 2000, No. 4, pp. 29-34

Peter Hanuliak – Peter Varša *

GRID IMPLEMENTATION OF PARALLEL ALGORITHM FOR LAPLACEAN EQUATION COMPUTATION BY JACOBI ITERATION METHOD

The article is devoted to the problem of parallel algorithms and their practical implementations. On the basis of parallel computer analysis in the world the parallel systems are divided into two basic groups – synchronous and asynchronous systems – which are very different from the system point of view. This article describes the development of real parallel algorithms for Jacobi iteration. This individual practical example demonstrates the influence of decomposition strategies for performance evaluation of parallel Jacobi iteration and discusses the ways for their parallel implementations.

1. Introduction

For the contemporary technical and programmable level of the reachable computer means (personal computers, minicomputers, supercomputers, etc.) the use of various forms of basic principles of the parallel activity is dominant [8, 15, 16]. For example in section of technical equipment the continuous speeding up of the individual processor performance is achievable mainly through the parallel activity of the pipeline execution in combination with blowing up the capacity and number of various buffer memories (caches).

In the field of programming equipment the parallel support goes also in two levels[15, 17, 18]. The first level forms the district of the operation systems and in general the system supporting programming tools. The second level creates the user developing programming environments, which support the development of the modular application programs as the basic condition to their potential parallel activity. This parallel support goes in this time up to the level of the elementary program elements in the form of the objects (OOP – object oriented programming).

The architectures of the parallel systems

In system classification we can divide parallel systems to the two very different groups [1]:

- synchronous parallel architectures. To this group belong practically all known parallel architectures except the computer networks. The basic system properties are given through the existence of some kind of the common shared memory M by parallel processors P_i, which in substantial measure simplifies their application programming using. The principal model is illustrated in Fig. 1.
- asynchronous parallel architectures. This group covers the field of various forms of computer networks. Their basic property is the mutual interconnection both in the remote form of the distributed memory moduls M_k and the parallel processors P_i with using the existed telecommunication lines (WAN networks) and in the local form in reaching range of the used fixed lines (LAN networks), respectively. There is, in contrast to the first discussed group, no form of common shared memory in the connected system. The principal model is in Fig. 2.

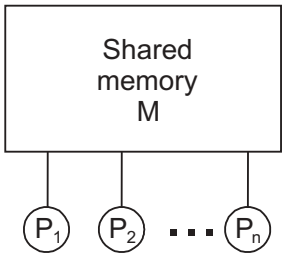


Fig. 1. Model of the system with shared memory.

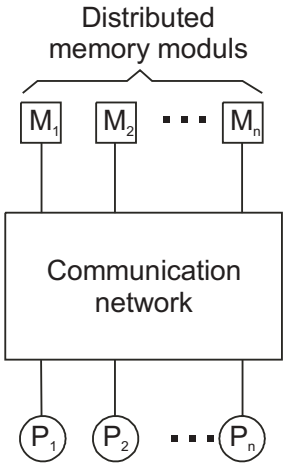


Fig. 2. Model of the parallel system with distributed memory moduls.

* **Peter Hanuliak, Peter Varša**
University of Žilina, Faculty of Management Science and Informatics, Slovakia
E-mail: hanuliak@frtk.fri.utc.sk, varsa@fri.utc.sk

The load balancing, inter-processor communication and transport protocol for such machines are being widely studied [6, 7, 13, 14, 15, 18]. With the availability of cheap personal computers, workstations and networking devises, the recent trend is to connect a number of such workstations to solve computation-intensive tasks in parallel on such clusters. To exploit the parallel processing capability of a NOW (Network of workstations), the application program must be parallelised. The effective way how to do it for a particular application problem (decomposition strategy) belongs to the most important step in developing a effective parallel algorithm [7, 16, 17].

The development of the parallel network algorithm includes the following activities:
- Decomposition – the division of the application into a set of parallel processes and data
- Mapping – the way how processes and data are distributed among the nodes
- Inter-process communication – the way of corresponding and synchronisation among individual processes
- Tuning – alternation of the working application to improve performance (performance optimisation)

When designing a parallel program the description of the high-level algorithm must include also the method we intend to use to divide the application into processes and to distribute data to different nodes – the decomposition strategy. The chosen decomposition method drives the rest of program development. This is true in case of developing new application as porting serial code. The decomposition method says how to structure the code and data and defines the communication topology.

To choose the best decomposition method for these applications, it is necessary to understand the particular application problem, the data domain, the used algorithm and the flow of control in given application. Therefore, according to the character of given task the following decomposition models are used:
- perfectly parallel decomposition
- domain decomposition
- control decomposition
- object-oriented programming – OOP.

2. The role of performance

Quantitative evaluation and modelling of hardware and software components of the parallel systems are critical for the delivery of high performance. Performance studies apply to initial design phases as well as to procurement, tuning, and capacity planning analysis. As performance cannot be expressed by quantities independent of the system workload, the quantitative characterisation of application resource demands and their behaviour is an important part of any performance evaluation study. Among the goals of parallel systems performance analysis is to estimate the performance of a system or a system component or an application, to investigate the match between requirements and system architecture characteristics, to identify the features that have a significant impact on the application execution time, to predict the performance of a particular application on a given parallel system, to evaluate different structures of parallel applications. To the performance evaluation we briefly review the techniques most commonly adopted for the evaluation of parallel systems and its metrics.

2.1 Performance evaluation methods

To the performance evaluation we can use the following methods:
- analytical methods
  - application of queueing theory [7, 8, 9, 10, 11, 12]
  - Petri nets [7, 14]
- simulation methods [2, 5]
  - experimental measurement [7, 14]
  - benchmarks [14, 16]
- direct measuring of particular developed parallel application.

In order to extend the applicability of analytical techniques to the parallel processing domain, various enhancements have been introduced to model phenomena such as simultaneous resource possession, fork and join mechanism, blocking and synchronisation. Hybrid modelling techniques allow to model contention both at hardware and software levels by combining approximate solutions and analytical methods. However, the complexity of parallel systems and algorithms limits the applicability of these techniques. Therefore, in spite of its computation and time requirements, simulation is extensively used as it imposes no constraints on modelling.

Evaluating system performance via experimental measurements is a very useful alternative for parallel systems and algorithms. Measurements can be gathered on existing systems by means of benchmark applications that aim at stressing specific aspects of the parallel systems and algorithms. Even though benchmarks can be used in all types of performance studies, their main field of application is competitive procurement and performance assessment of existing systems and algorithms. Parallel benchmarks extend the traditional sequential ones by providing a wider set of suites that exercise each system component targeted workload. The Parkbench suite especially oriented to message passing architectures and the SPLASH suite for shared memory architectures are among the most commonly used benchmarks [14].

2.2. Performance evaluation metrics

For evaluating the parallel algorithms there have been developed several fundamental concepts [1, 6]. Tradeoffs among these performance factors are often encountered in real-life applications.

2.2.1. Performance concepts

Let $O(s, p)$ be the total number of unit operations performed by p-processor system for size s of the computational problem and

$T(s, p)$ be the execution time in unit time steps. In general, $T(s, p) < O(s, p)$ if more than one operation is performed by p processors per unit time, where $p \geq 2$. Assume $T(s, 1) = O(s, 1)$ in a single-processor system (sequential system). The *speedup factor* is defined as:

$$S(s, p) = \frac{T(s, 1)}{T(s, p)} \quad (1)$$

(Note that this speedup is a limit case and almost never obtained.) It is a measure of the speedup obtained by given algorithm when p processors are available for the given problem size s. Ideally, since $S(s, p) \leq p$, we would like to design algorithms that achieve $S(s, p) \approx p$.

The system *efficiency* for an p-processor system is defined by:

$$E(s, p) = \frac{S(s, p)}{p} = \frac{T(s, 1)}{p\ T(s, p)} \quad (2)$$

A value of $E(s, p)$ for some p approximately equal to 1 indicates that such a parallel algorithm, using p processors, runs approximately p times faster than it does with one processor (sequential algorithm).

2.2.2. The isoefficiency concept

The workload w of an algorithm often grows in the order $O(s)$, where s is the problem size. Thus, we denote the workload $w = w(s)$ as a function of s. In parallel computing is very useful to define an isoefficiency function relating workload to machine size p needed to obtain a fixed efficiency E when implementing a parallel algorithm on a parallel system. Let h be the total communication overhead involved in the algorithm implementation. This overhead is usually a function of both machine size and problem size, thus denoted $h = h(s, p)$.

The efficiency of a parallel algorithm implemented on a given parallel computer is thus defined as

$$E(s, p) = \frac{w(s)}{w(s) + h(s, p)} \quad (3)$$

The workload $w(s)$ corresponds to useful computations while the overhead $h(s, n)$ is the useless time attributed to synchronisation and data communication delays. In general, the overhead increases with respect to both increasing values of s and p. Thus, the efficiency is always less than 1. The question is the relative growth rates between $w(s)$ and $h(s, p)$.

With a fixed problem size (fixed workload), the efficiency decreases as p increase. The reason is that the overhead $h(s, p)$ increases with p. With a fixed machine size, the overhead h grows slower than the workload w. Thus the efficiency increases with increasing problem size for a fixed-size machine. Therefore, one can expect to maintain a constant efficiency if the workload w is allowed to grow properly with increasing machine size.

For a given algorithm, the workload w might need to grow polynomially or exponentially with respect to p in order to maintain a fixed efficiency. Different algorithms may require different workload growth rates to keep the efficiency from dropping, as p is increased. The isoefficiency functions of common parallel algorithms are polynomial functions of p; i. e., they are $O(p^k)$ for some $k \geq 1$.

We can rewrite the equation for efficiency $E(s, p)$ *as* $E(s, p) = 1/(1 + h(s, p)/w(s))$. In order to maintain a constant E, the workload $w(s)$ should grow in proportion to the overhead $h(s, p)$. This leads to the following relation:

$$w(s) = \frac{E}{1 - E} h(s, p) \quad (4)$$

The factor $C = E/(1 - E)$ is a constant for a fixed efficiency E. Thus we can define the isoefficiency function as follows: $f_E(p) = C \cdot h(s, p)$. If the workload grows as fast as $f_E(p)$, then a constant efficiency can be maintained for a given algorithm-architecture combination.

3. Theoretical part

Partial differential equations (PDE) are used to model a variety of different kinds of physical systems: weather, airflow over a wing, turbulence in fluids, and so on. Some simple PDE's can be solved directly, but in general it is necessary to approximate the solution at a finite number of points using iterative numerical methods. Here we show how to solve in parallel one specific PDE – Laplace's equation in two dimensions – by means of a grid computation method that employs a finite-difference method. Although we focus on this specific problem, the same programming techniques are used in grid computations for solving other PDE's and in other applications such as image processing etc.

3.1. Laplace's Equation

Laplace' equation is a practical example of Jacobi iteration application. The equation for two dimensions is as follows:

$$\frac{\partial^2 \Phi}{\partial x^2} + \frac{\partial^2 \Phi}{\partial y^2} = 0 \quad (5)$$

Function $\Phi(x, y)$ represents some unknown potential, such as heat or stress.

Given a two-dimensional region and values for points of the region boundaries, the goal is to approximate the steady-state solution $\Phi(x, y)$ for points in the interior by the function $u(x,y)$. We can do this by covering the region with a grid of points and to obtain the values of $u(x_i, y_j) = u_{ij}$ as follows:

Let us consider square region $(a, b) \times (a, b)$, thus for coordinates of grid points is valid $x_i = i * h$, $y_j = j * h$, $h = (b-a)/N$ for $i,j = 0, 1, ..., N$. We replace partial derivations of $\Phi \sim u(x, y)$ by the differences of $u_{i,j}$:

$$\frac{\partial^2\Phi}{\partial x^2} \cong \frac{\Delta_x^{(2)} u_{i,j}}{h^2} = \frac{\Delta_x^{(1)} u_{i,j} - \Delta_x^{(1)} u_{i-1,j}}{h^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}$$

$$\frac{\partial^2\Phi}{\partial y^2} \cong \frac{\Delta_y^{(2)} u_{i,j}}{h^2} = \frac{\Delta_y^{(1)} u_{i,j} - \Delta_y^{(1)} u_{i,j-1}}{h^2} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}. \quad (6)$$

and after substituting (6) in (5) we obtain

$$u_{i,j} = (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1})/4 \quad i,j = 1, 2, \ldots N;$$

Each interior point is initialised to some value. The steady-state values of the interior points are then calculated by repeated iterations. On each iteration the new value of a point is set to a combination of the previous values of neighbouring points. The computation terminates either after a given number of iterations or when every new value is within some acceptable difference $\epsilon > 0$ of the previous value.

There are several iterative methods for solving Laplace's equation, including Jacobi iteration, Gauss – Seidel iteration, successive over-relaxation (SOR), and multigrid [1, 6, 19]. In this paper we show parallel implementation of Jacobi iteration using Message Passing Interface (MPI). The algorithms for other methods converge more rapidly but are somewhat more complex than Jacobi iteration. Their parallel implementations have similar communication and synchronisation patters and these aspects are the most important.

3.2. Jacobi iteration

We applied Jacobi point iterative method to the grid on given region $(0, 1) \times (0, 1)$ for which the boundary conditions are known. $(N-1)*(N-1)$ is the number of interior grid points. The Laplace' equation will be in form:

$$u_{i,j} = (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1})/4 \;\; i,j = 1, 2, \ldots N-1;$$

$$u_{0,j} = -y_j^2;\; u_{N,j} = 1 - y_j^2; \quad j = 1, 2, \ldots N-1; \quad (7)$$

$$u_{i,0} = x_i^2;\; u_{i,N} = x_i^2 - 1; \quad i = 1, 2, \ldots N-1;$$

where

$$x_i = i/N; \qquad y_j = j/N;$$

In this case the exact solution is known:

$$U_{i,j} = x_i^2 - y_j^2$$

3.3. Local communication

A local communication structure is obtained when an operation requires data from a small number of other tasks. It is then straightforward to define channels that link the tasks responsible for performing the operation (the consumers) with the tasks holding the required data (the producers) and to introduce appriopriate send and receive operations in the producer and consumer tasks, respectively.

For Jacobi finite difference method a two-dimensional grid is repeatedly updated by replacing the value at each point with some function of the values at a small fixed number of neightbouring points. The following expression uses a four-point stencil to update each element $X_{i,j}$ of a two -dimensional grid X:

$$X_{i,j}^{(t+1)} = (X_{i-1,j}^{(t)} + X_{i+1,j}^{(t)} + X_{i,j-1}^{(t)} + X_{i,j+1}^{(t)})/4 \quad (8)$$

This update is applied repeatedly to compute a sequence of values $X_{i,j}^{(1)}, X_{i,j}^{(2)}, \ldots$ and so on. The notation $X_{i,j}^{(t)}$ denotes the value of grid point $X_{i,j}$ at step t.

Let us assume that a partition has used domain decomposition techniques to create a distinct task for each point in two dimensional grid. Hence, a task allocated the grid point $X_{i,j}$ must compute the sequence

$$X_{i,j}^{(1)}, X_{i,j}^{(2)}, X_{i,j}^{(3)}, \ldots \quad (9)$$

This computation requires in turn the four corresponding sequences

$$\begin{aligned} &X_{i-1,j}^{(0)}, X_{i-1,j}^{(1)}, X_{i-1,j}^{(2)}, \ldots \\ &X_{i+1,j}^{(0)}, X_{i+1,j}^{(1)}, X_{i+1,j}^{(2)}, \ldots \\ &X_{i,j-1}^{(0)}, X_{i,j-1}^{(1)}, X_{i,j-1}^{(2)}, \ldots \\ &X_{i,j+1}^{(0)}, X_{i,j+1}^{(1)}, X_{i,j+1}^{(2)}, \ldots \end{aligned} \quad (10)$$

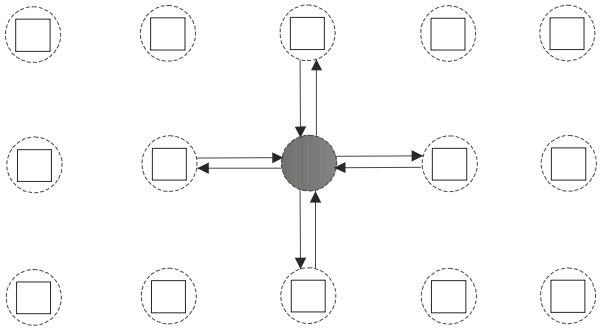


Fig. 3. Local communication.

which are produced by four tasks handling grid points $X_{i-1,j}$, $X_{i+1,j}$, $X_{i,j-1}$ and $X_{i,j+1}$, that is, by its four neighbours in the grid. For these values to be communicated, we define channels linking each task that requires a value with the task that generates that value. This yields the channel structure illustrated in Fig. 3. For first T-steps each task then executes the following logic:

```
for t = 0 to T - 1 do
  begin
    send X(t)i,j to each neighbour;
    receive X(t)i-1,j, X(t)i+1,j, X(t)i,j-1, X(t)i,j+1 from neighbours;
    compute X(t+1)i,j using Equation (8);
  endfor
```

4. Experimental part and results

We used two decomposition strategies in order to analyse their influence to the performance evaluation:

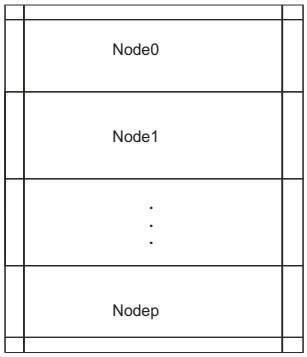


Fig. 4. Domain decomposition 1.

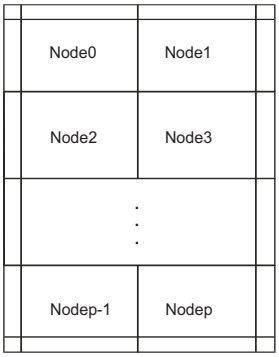


Fig. 5. Domain decomposition 2.

a) the decomposition of Jacobi iteration according to Fig. 4 (domain decomposition 1). In this strategy there was given each computation node a horizontal strip of U_{ij}. After each iteration „boundary conditions"of the strip have to be shared with neghbouring nodes (Fig. 6 and 7).

b) the decomposition according to Fig. 5 (domain decomposition 2). In this strategy we used twice much computation nodes as in the first case.

In Fig. 8. we illustrate the performance of both decomposition strategies. We can see that when using more computation processors we did not come to increased performance. The causes are in increasing the overheads (control, communication, synchronisation) more than the speed-up of higher computation nodes. The realised experiments were done on parallel system at EPCC Edinbourgh (parallel system Cray T3E).

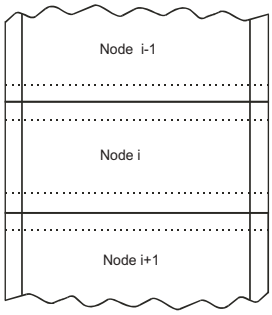


Fig. 6. The shared points.

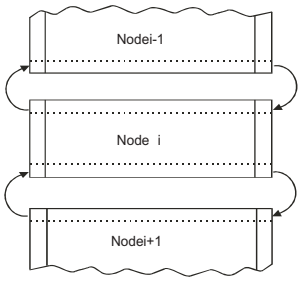


Fig. 7. Exchange of the values.

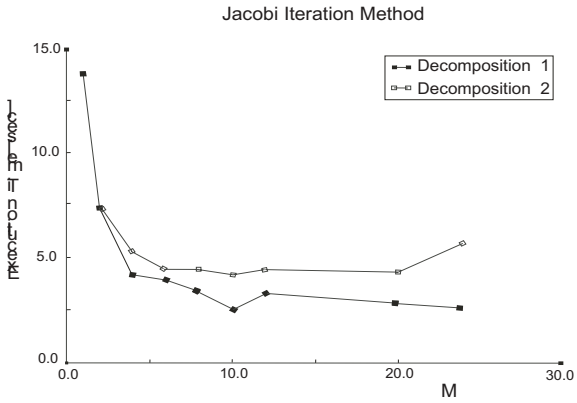


Fig. 8. The results (M = number of used processors)

References

[1] ANDREWS G. R., *Foundations of Multithreaded, Parallel, and Distributed Programming*, Addison Wesley Longman, Inc., 664 pp., 2000, USA

[2] BANKS J., DAI J. G.: *Simulation studies of multiclass queueing networks,* IEEE Transactions, Volume 29, 1997, pp. 213-219

[3] BASOGLU CH., LEE W., KIM Y.: *An efficient FFT algorithm for Super-scalar and VLIW Processor Architectures,* Real Time Imaging 3, pp. 441-453, 1997, USA

[4] ČERNÁ M., MACHALICKÝ M., VOGEL J., ZLATNÍK Č.: *A first course in numerical mathematics and programming (in czech),* Alfa/SNTL, Praha, 1987

[5] FODOR G., BLAABJERG S., ANDERSEN A.: *Modelling and simulation of mixed queueing and loss systems,* Wireless Personal Communication, N. 8, 1998, pp. 253-276

[6] GREENBERG D. S., PARK J. K., SCHVABE E. J.: *The cost of complex communication on simple networks*, Journal of Parallel and Distributed Computing 35, pp. 133-141, 1996

[7] HANULIAK I.: *Parallel architecture - multiprocessors, computer networks (in slovak),* 187 strán, 127 obr.,17 tab., Júl 1997, vyd.: Knižné centrum, Žilina

[8] HANULIAK I.: P*arallel computers and algorithms (in Slovak)*, Košice (Slovakia), ELFA Press 1999, 327 pp.
[9] HANULIAK J.: *To a complexity of parallel algorithms,* In Proceedings: TRANSCOM 2001 (4-th European Conference in Transport and Telecommunications), 25-27 June 2001, pp. 51-54, Žilina, Slovak Republic
[10] HANULIAK J., HANULIAK I., MATIASKO K.: *To parallel implementation of Discrete Fast Fourier Transform,* Journal of the Applied Sciences Mittweida, No. 15, 2000, pp. 3-10, Mittweida, Germany
[11] HANULIAK M.: *To the behaviour analysis of mobile data networks,* in Proceedings of 7th Scientific conference, November 9-10, pp. 170-175, 2001, T_RGU - JIU, Romania
[12] HARRISON P. G., PATEL N.: *Performance modelling of communication networks and computer architectures,* Addison – Wesley Publishers 1993, 480 pp.
[13] HSU W. T., PEN-CHUNG Y.: *Performance Evaluation of Wire-Limited Hierarchical Networks,* Parallel and Distributed Computing 41, 1997, pp. 156-172
[14] HESHAM EL-REWINI, TED. G. LEWIS: *Distributed and parallel computing,* 467 pp., Manning Publications Co., 1997, USA
[15] HWANG K., XU Z.: *Scalable Parallel Computing: Technology, Architecture, Programming,* Mc Graw-Hill Companies, 802 pp., 1998, USA
[16] KUMAR V., GRAMA A., GUPTA A., KARYPIS G.: *Introduction to Parallel Computing (Second Edition)*, Addison Wesley, 856 pp., 2001
[17] MARINESCU D. C., RICE J. R.: *On the scalability of asynchronous parallel computations,* Parallel and Distributed Computing 31, pp. 88-97, 1995
[18] NANCY A. L.: *Distributed Algorithms,* 872 pp., 1996, Morgan Kaufmann Publishers, Inc., USA
[19] VAJTERŠIC M.: *Modern algorithms for solving some elliptic pariial differential equations (in slovak),* Veda, Bratislava, 1988
[20] VARŠA P.: *Contribution to complexity of distributed parallel algorithms (in Slovak)*, Dissertation theses, March 2003, 94 pp., University of Zilina, Žilina, Slovakia
[21] WILLIAMS R.: *Computer Systems Architecture – A networking approach,* Addison Wesley, 660 pp., 2001, England.

kamUNikácie
COMMUNICATIONS

Petr Fiala *

SUPPLY CHAINS IN NETWORK ECONOMY

A supply chain can be shown as a set of units interconnected by material, financial, information and decision flows from initial suppliers to ultimate customers. Supply chain management is more and more affected by network business environment and by information and communication technologies. The paper analyzes a combination of non-cooperative and cooperative decision making of supply chains in network economy.

Keywords: *Supply chain, network economy, e-commerce, non-cooperative and cooperative decision-making*

1. Supply chain management

Supply chain management is now seen as a governing element in strategy and as an effective way of creating value for customers. Supply chain management benefits from a variety of concepts that were developed in several different disciplines as marketing, information systems, economics, system dynamics, logistics, operations management, and operations research. There are many concepts and strategies applied in designing and managing supply chains (see [13]). The expanding importance of supply chain integration presents a challenge to research to focus more attention on supply chain modeling (see [14], [17]).

A structure of supply chains is composed of potential suppliers, producers, distributors, retailers and customers etc. The units are interconnected by material, financial, information and decisional flows (see Fig.1).

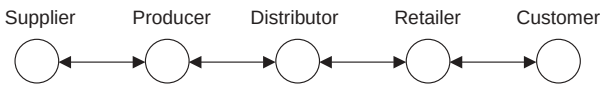


Fig.1. Structure of a supply chain

Most supply chains are composed of independent units with individual preferences. Each unit will attempt to optimize its own preference. Behavior that is locally efficient can be inefficient from a global point of view. In supply chain behavior are many inefficiencies. An increasing number of companies in the world subscribe to the idea that developing long-term coordination and cooperation can significantly improve the efficiency of supply chains and provide a way to ensure competitive advantage.

Double marginalization is a well-known cause of supply chain inefficiency (see [14]). Double marginalization problem occurs whenever the supply chain's profits are divided among two or more firms and at least one of the firms influences demand. Each firm only considers its own profit margin and does not consider the supply chain's margin.

We consider a supply chain with a producer and a retailer that sells a product. The producer produces each unit for a cost c and sells each unit to the retailer for a wholesale price $p^{(1)}$. The retailer chooses an order quantity x and sells x units at price $p^{(2)}(x)$, assuming that $p^{(2)}(x)$ is decreasing, concave and twice differentiable function.

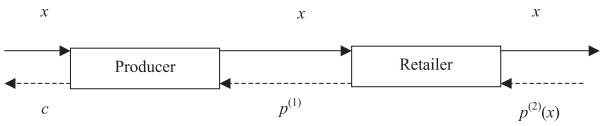


Fig. 2. Double marginalization problem

A centralized solution assumes a single agent has complete information and controls the entire supply chain (this is referred as the first-best solution) to maximize supply chain profit

$$z(x) = x\,(p^{(2)}(x) - c).$$

The centralized solution of the problem is denoted x^0.

A decentralized solution assumes the firms have incomplete information and make choices with the objective of maximizing their own profits. The retailer's profit and the producer's profit are

$$z_r(x) = x\,(p^{(2)}(x) - p^{(1)}), \quad z_p(x) = x\,(p^{(1)} - c)$$

The decentralized solution of the problem is denoted x^*.

If the centralized and decentralized solutions differ, we investigate how to modify the firm's payoffs so that a new decentralized solution corresponds to the centralized solution. It can be shown that the retailer orders less than the supply chain optimal quantity ($x^0 > x^*$) whenever the producer earns a positive profit and it holds

$$z(x^0) > z_r(x^*) + z_p(x^*)$$

* **Petr Fiala**
Department of Econometrics, Faculty of Informatics and Statistics, University of Economics, W. Churchill Sq. 4, 130 67 Prague 3
E-mail: pfiala@vse.cz

Marginal cost pricing ($p^{(1)} = c$) is one solution to double marginalization problem, but the producer earns a zero profit. A better solution is a profit sharing contract, where the producer earns $v\,z(x)$ and the retailer earns $(1-v)\,z(x)$, for $0 \leq v \leq 1$. The wholesale price $p^{(1)}$ is now irrelevant to each firm's profit and the supply chain earns the optimal profit.

In today's global world there are many shifts. Supply chain management is more and more affected by network business environment and by information and communication technologies (see [13]). The global production systems move from rigid supply chain structures into globally distributed dynamic networks of agile business units. The shift in power from the supplier to the customer should lead to analyzing supply chains from a demand perspective. The paper analyzes a combination of non-cooperative and cooperative decision-making of supply chains in network economy.

2 Network economy environment

The network economy (see [6], [9] [11], [12]) is a term for today's global relationship among economic subjects characterized by massive connectivity. The central act of the new era is to connect everything to everything in deep web networks at many levels of mutually interdependent relations, where resources and activities are shared, markets are enlarged and costs of risk are reduced. Network industries play a crucial role in modern life. Today network systems provide the infrastructure and foundation for the functioning of societies and economies. They come in many forms and include physical networks such as: transportation and logistical networks, communication networks, energy networks, as well as more abstract networks comprising: economic and financial networks, environmental networks, social, and knowledge networks. The decisions made by the users of the networks, in turn, affect not only the users themselves but others, as well, in terms of profits and costs, timeliness of deliveries, the quality of the environment, etc. The behavior of the users of the networks themselves may be non-cooperative. One of the principal facets of the network economy is the interaction among the networks themselves. The unifying concept of global networks with associated methodologies (see [10]) allows exploring the interactions among such networks as transportation networks, telecommunication networks, as well as financial networks.

Network economy drives and is driven by dramatic acceleration in technological innovation, in information and communication technologies especially. New technologies provide a permanent feedback that enables activity modifications and quick responses and therefore fundamentally change business models. Business process modeling is the using of models and methods for understanding and change of the processes in relation to information systems of firms. The relation between business models and information systems becomes more and more tighter. E-business (see [16]) can enhance supply chain management decision making by making it possible to collect real time information and access and analyze the data in order to facilitate cooperation between trading partners in a supply chain. Using technology to connect ever more closely with network users creates "connectivity paradox" in which new realms of opportunity coexist with new forms of risk.

To achieve joint optimization of key supply chain decisions, it is preferable that there be a free flow of all relevant information across the entire network. Modeling of supply chains in network economy includes coordination of flows in different network structures and combination of non-cooperative and cooperative decisions. The presented model considers three-layer network structures of producers, retailers and customers. An overview of material, financial an informational flows is schematically shown in Fig. 3.

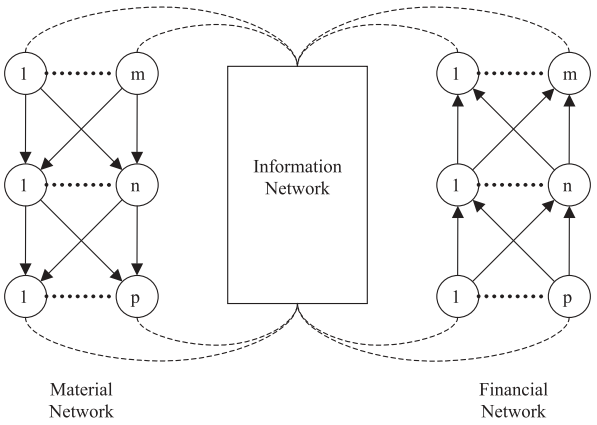


Fig.3. Material, financial an information flows in network environment

3. Modeling of supply chain networks with e-commerce

In the model we consider a three-layer network consists of m producers ($i = 1, 2, ..., m$), n retailers ($j = 1, 2, ..., n$) and p types of consumers (markets) ($k = 1, 2, ..., p$). Producers sell the homogenous product to retailers by material and/or electronic channels. Producers can sell the product also directly to consumers by electronic channels. Retailers sell the product to consumers by material and/or electronic channels. Consumers can purchase the product in three ways, electronically from producers and/or retailers and by material channels from retailers. E-commerce is associated with the ability of business to perform transactions automatically. Electronic links between producers and retailers represent the business-to-business (B2B) component and electronic links between retailers and consumers represent the business-to-consumer (B2C) component of e-commerce.

Figure 3 depicts the three-layer network of interconnected producers, retailers and consumers. Continues lines represent material links and dash lines represent electronic links.

We define some notation:

$x_{ij}^{(1)}$ = a quantity of product shipment from producer i to retailer j by material links,

$y_{ij}^{(1)}$ = a quantity of product shipment from producer i to retailer j by electronic links,

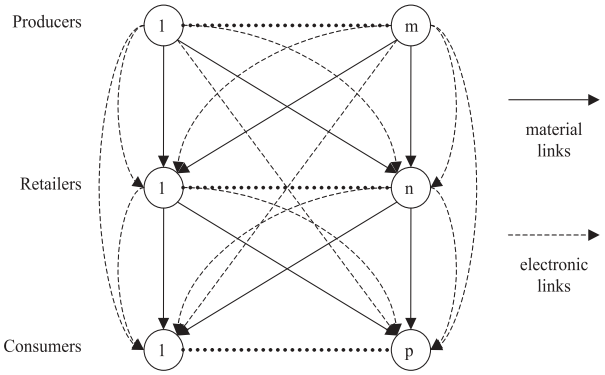


Fig. 4. Supply chain network with e-commerce

$x_{jk}^{(2)}$ = a quantity of product shipment from retailer j to consumer k by material links,

$y_{jk}^{(2)}$ = a quantity of product shipment from retailer j to consumer k by electronic links,

$y_{ik}^{(3)}$ = a quantity of product shipment from producer i to consumer k by electronic links,

$x^{(1)} = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}^{(1)}$ = a quantity of product shipment from all producers to all retailers by material links,

$y^{(1)} = \sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij}^{(1)}$ = a quantity of product shipment from all producers to all retailers by electronic links,

$x^{(2)} = \sum_{j=1}^{n}\sum_{k=1}^{p} x_{jk}^{(2)}$ = a quantity of product shipment from all retailers to all consumers by material links,

$y^{(2)} = \sum_{j=1}^{n}\sum_{k=1}^{p} y_{jk}^{(2)}$ = a quantity of product shipment from all retailers to all consumers by electronic links,

$y^{(3)} = \sum_{i=1}^{m}\sum_{k=1}^{p} y_{ik}^{(3)}$ = a quantity of product shipment from all producers to all consumers by electronic links,

$p^{(1)} = p^{(1)}(x^{(1)})$ = a price charged for the product transacted by material links by producers to retailers as a function of the total quantity $x^{(1)}$,

$p^{(2)} = p^{(2)}(x^{(2)})$ = a price charged for the product transacted by material links by retailers to customers as a function of the total quantity $x^{(2)}$,

$q^{(1)} = q^{(1)}(y^{(1)})$ = a price charged for the product transacted by electronic links by producers to retailers as a function of the total quantity $y^{(1)}$,

$q^{(2)} = q^{(2)}(y^{(2)})$ = a price charged for the product transacted by electronic links by retailers to customers as a function of the total quantity $y^{(2)}$,

$q^{(3)} = q^{(3)}(y^{(3)})$ = a price charged for the product transacted by electronic links by producers to customers as a function of the total quantity $y^{(3)}$,

$c_i^{(1)}$ = production unit cost for producer i,

$c_j^{(2)}$ = handling unit cost for retailer j,

$c_{ij}^{(1)}$ = transaction cost for production unit from producer i to retailer j by material links,

$d_{ij}^{(1)}$ = transaction cost for production unit from producer i to retailer j by electronic links,

$c_{jk}^{(2)}$ = transaction cost for production unit from retailer j to consumer k by material links,

$d_{jk}^{(2)}$ = transaction cost for production unit from retailer j to consumer k by electronic links,

$d_{ik}^{(3)}$ = transaction cost for production unit from producer i to consumer k by electronic links,

The model is a generalized oligopoly game (see [7]). The producers and retailers compete in a non-cooperative way. Nash equilibrium concept is used that each maximizes his profit given the actions of the others.

Quantities of production shipment are nonnegative variables and satisfy the obvious constraints

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}^{(1)} \geq \sum_{j=1}^{n}\sum_{k=1}^{p} x_{jk}^{(2)},$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij}^{(1)} \geq \sum_{j=1}^{n}\sum_{k=1}^{p} y_{jk}^{(2)}.$$

Prices are nonnegative variables as functions of the quantities.

Consumers buy the product through three different channels: material links from retailers, electronic links from retailers and electronic links from producers. Consumer's (market) k ($k = 1, 2, ..., p$) demands are decreasing functions of prices

$$D_k^{(1)} = D_k^{(1)}(p^{(2)}),$$

$$D_k^{(2)} = D_k^{(2)}(q^{(2)}),$$

$$D_k^{(3)} = D_k^{(3)}(q^{(3)}).$$

Producer's i profit is expressed as follows:

$$z_i^{(1)} = \sum_{j=1}^{n} p^{(1)}x_{ij}^{(1)} + \sum_{j=1}^{n} q^{(1)}y_{ij}^{(1)} + \sum_{k=1}^{p} q^{(3)}y_{ik}^{(3)} - c_i^{(1)}\left(\sum_{j=1}^{n} x_{ij}^{(1)} + \sum_{j=1}^{n} y_{ij}^{(1)} + \sum_{k=1}^{p} y_{ik}^{(3)}\right) - \sum_{j=1}^{n} c_{ij}^{(1)}x_{ij}^{(1)} - \sum_{j=1}^{n} d_{ij}^{(1)}y_{ij}^{(1)} - \sum_{k=1}^{p} d_{ik}^{(3)}y_{ik}^{(3)}$$

Equilibrium conditions for producers ($i = 1, 2, ..., m$) are given by the following equality system:

$$\frac{\partial z_i^{(1)}}{\partial x_{ij}^{(1)}} = \frac{\partial p^{(1)}}{\partial x_{ij}^{(1)}} x_{ij}^{(1)} + p^{(1)} - c_i^{(1)} - c_{ij}^{(1)} = 0,\ j = 1, 2, ..., n,$$

$$\frac{\partial z_i^{(1)}}{\partial y_{ij}^{(1)}} = \frac{\partial q^{(1)}}{\partial y_{ij}^{(1)}} y_{ij}^{(1)} + q^{(1)} - c_i^{(1)} - d_{ij}^{(1)} = 0,\ j = 1, 2, ..., n,$$

$$\frac{\partial z_i^{(1)}}{\partial y_{ik}^{(3)}} = \frac{\partial q^{(3)}}{\partial y_{ik}^{(3)}} y_{ik}^{(3)} + q^{(3)} - c_i^{(1)} - d_{ik}^{(3)} = 0,\ k = 1, 2, ..., p. \tag{1}$$

Retailer's j profit is expressed as follows:

$$z_i^{(2)} = \sum_{k=1}^{p} p^{(2)} x_{jk}^{(2)} + \sum_{k=1}^{p} q^{(2)} y_{jk}^{(2)} - c_j^{(2)} \left(\sum_{k=1}^{p} x_{jk}^{(2)} + \right.$$

$$\left. + \sum_{j=1}^{n} y_{jk}^{(2)} \right) - \sum_{k=1}^{p} c_{jk}^{(2)} x_{jk}^{(2)} - \sum_{j=1}^{n} d_{jk}^{(2)} y_{jk}^{(2)}$$

Equilibrium conditions for producers ($j = 1, 2, ..., n$) are given by the following equality system:

$$\frac{\partial z_j^{(2)}}{\partial x_{jk}^{(2)}} = \frac{\partial p^{(2)}}{\partial x_{jk}^{(2)}} x_{jk}^{(2)} + p^{(2)} - c_j^{(2)} - c_{jk}^{(2)} = 0,\ k = 1, 2, ..., p,$$

$$\frac{\partial z_j^{(2)}}{\partial y_{jk}^{(2)}} = \frac{\partial q^{(2)}}{\partial y_{jk}^{(2)}} y_{jk}^{(2)} + q^{(2)} - c_j^{(2)} - d_{jk}^{(2)} = 0,\ k = 1, 2, ..., p. \tag{2}$$

Equilibrium conditions for customers ($k = 1, 2, ..., p$) are given by the following equality system:

$$D_k^{(1)} = \sum_{j=1}^{n} x_{jk}^{(2)},$$

$$D_k^{(2)} = \sum_{j=1}^{n} y_{jk}^{(2)}, \tag{3}$$

$$D_k^{(3)} = \sum_{i=1}^{m} y_{ik}^{(3)}.$$

The equilibrium state for the supply chain network is given by a set of equilibrium conditions (1), (2) and (3) for quantities of product shipment and prices.

4. Modeling of cooperation in supply chains

The layers of producers and retailers compete in a non-cooperative way but the partners in individual supply chains can profit from cooperative decision-making. The strategic partnership (see [8], [15]) means cooperation and coordination of actions through the supply chain. The strategic partnerships change material, financial and information flows among participants in the supply chain. Information centralizing using information technology changes the way of information sharing. The expected result is a mutually beneficial, win-win partnership that creates a synergistic supply chain in which the entire chain is more effective than the sum of its individual parts. Supply chain partnership leads to increased information flows, reduced uncertainty, and a more profitable supply chain.

The general supplier-customer relations in supply chain can be taken as centralized or decentralized (see [3]). The partnership relations are based on supply contracts. Contracts provide a means for bringing the decentralized solution to the centralized solution. Contracts also facilitate long-term partnership by delineating mutual concessions that favor the persistence of the relationship, as well a specifying penalty for non-cooperative behavior. The contracts are evaluated by multiple criteria as price, quantity, costs, time and quality and so on. There are different approaches to modeling multicriteria negotiation processes to reach a consensus among partners. A cooperative decision-making requires free communication among agents and gives synergical effects in a conflict resolution. The basic trend in the cooperative decision-making is to transform a possible conflict to a joint problem.

Some basic ideas of formal approaches to the problem solving can be introduced to cooperative decision-making. There are two aspects of the problem solving – representation and searching. The state space representation introduces the concepts of states and operators. An operator transforms one state into another state. A solution could be obtained by a search process, first applies operators to the initial state to produce new states and so on, until the goal state is produced. Communication between producers and retailers can be provided through information sharing (schematically see Fig. 5).

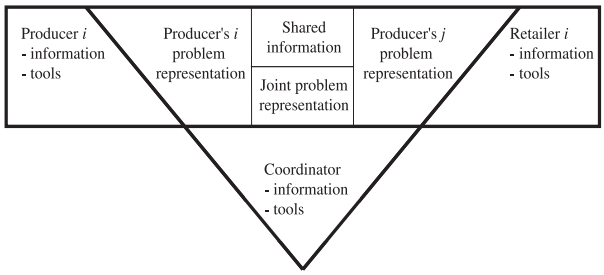


Fig. 5. Communication through information sharing

The proposed model is a discrete dynamic model and the cooperation of units is based on contracts and formal agreements achieved in negotiation process. We propose a two phases' interactive approach for solving cooperative decision making problems (see [2]):

1. Finding the ideal solution for individual agents.
2. Finding a consensus for all the agents.

In the first phase every decision maker search the ideal alternative by the assertivity principle. In the second phase the search process could obtain a consensus and the principle of cooperativeness is applied. The heuristic information for the decision-making unit is the distance between his proposal and the opponent's proposal.

The negotiation process modeling in general is a complex problem based on several kernel ideas. The framework (see [4]) of the proposed discrete dynamic model is separated to three parts:

- deterministic part,
- logical part,
- stochastic part.

According to these three parts the modeling framework is composed from three inter-related network structures:

- flow network,
- Petri network,
- neural network.

The flow network represents the supply chain network with information, material and financial flows between partners. Petri network is used to coordinate asynchronous events of different units in the supply chain and to model negotiation process. The neural network serves as an instrument for inductive learning of negotiation strategies.

Producers and retailers in individual supply chains negotiate contracts about prices, quantities, costs, time etc. The double marginalization problem can be generalized and producers and retailers negotiate the profit sharing contracts.

5. Conclusions

Supply chain management has generated a substantial amount of interest both by managers and researchers. The interest has also been fueled by the growth in the development and application of e-business technologies. E-business is associated with business models and practices enabling continuous improvements in supply chains. Supply chain management is more and more affected by network business environment and by information and communication technologies. The network economy is characterized by massive global connectivity relationship among economic subjects.

The paper is devoted to the modeling of supply chains in network economy environment. In the model some important features of this environment are established. The presented model considers three-layer network structures of producers, retailers and customers. A possibility of e-commerce is also introduced. The layers of producers and retailers compete in a non-cooperative way but the partners in individual supply chains can profit from cooperative decision-making. The combination of non-cooperative and cooperative behavior of network users is more realistic.

The research project was supported by Grant No. 402/01/0771 from the Grant Agency of the Czech Republic "Modeling of Supply Chains" and CEZ: J 18/98: 311401001 from the University of Economics "Models and Methods for Economic Decisions".

References

[1] ECONOMIDES, N.: *The Economics of Networks.* International Journal of Industrial Organization 14, no. 2, 1996.
[2] FIALA, P.: *Models of cooperative decision making.* In: Multiple Criteria Decision Making (T. Gal and G. Fandel, eds.), pp. 128-136. Springer, Berlin 1997.
[3] FIALA, P.: *Modeling of relations in supply chains.* Vision: The Journal of Business Perspective. Special Issue on Supply Chain Management, Vol. 7, 2003,pp. 127-131.
[4] FIALA, P., KUKAL, J.: *Modeling of cooperation in network production systems.* APORS 2000, Singapore 2000.
[5] ISHIDA, T. (ed.): *Community Computing – Collaboration over Global Information Networks.* Wiley, New York 1998.
[6] KELLY, K.: *New Rules for the New Economy: 10 Radical Strategies for a Connected World.* Viking Press, New York 1998.
[7] KREPS, D.: *Game Theory and Economic Modelling.* Oxford University Press, Oxford 1991.
[8] MACBETH, D. K., FERGUSON, N.: *Partnership Sourcing: An Integrated Supply Chain Management Approach.* Pitman Publishing, London 1994.
[9] A. NAGURNEY, A.: *Network Economics: A Variational Inequality Approach.* Kluwer Academic Publishers, Dordrecht 1999.
[10] NAGURNEY, A., DONG, J.: *Supernetworks: Decision-Making for the Information Age.* Edward Elgar Publishers, Cheltenham 2002.
[11] SHAPIRO, C., VARIAN, H.: *Information Rules: A Strategic Guide to the Network Economy.* Harvard Business School Press, Boston 1999.
[12] SHY, O.: *The Economics of Network Industries.* Cambridge University Press, Cambridge 2001.
[13] SIMCHI-LEVI, D., KAMINSKY, P., SIMCHI-LEVI, E.: *Designing and Managing the Supply Chain: Concepts, Strategies and Case studies.* Irwin/ Mc Graw-Hill, 1999.
[14] TAYUR, S., GANESHAN, R., MAGAZINE, M.: *Quantitative models for supply chain management,* Kluwer, Boston 1999.
[15] THOMAS, D. J., GRIFFIN, P. M.: *Coordinated Supply Chain Management.* European Journal of Operational Research 94, 1996, 1-15.
[16] TIMMERS, P.: *Electronic Commerce: Strategies and Models for Business-to-Business Trading.* John Wiley, Chichester 1999.
[17] VIDAL, C. J., GOETSCHALCKX, M.: *Strategic production-distribution models:* A critical review with emphasis on global supply chain models. European Journal of Operational Research 98, 1997, 1-18.

Peter Czimmermann *

A NOTE ON USING GRAPHS IN REGULAR SCHEDULING PROBLEMS

This paper deals with regular permutation scheduling on graphs. Peško and Czimmermann introduced this problem (in [3]) and it is generalisation of a matrix permutation problem. The goal is to minimise differences between row sums of a real matrix that represents a schedule, but external conditions don't allow moving matrix elements arbitrarily. The conditions can be represented by permutation obtained from a certain graph.

1. Introduction

The matrix permutation problem (MPP) is to minimise differences between row sums by permuting elements in columns of a matrix. This problem was defined first by Š. Peško in [5]. The exact definition of MPP (from [1]) is:

There is given matrix $(a_{i,j}) \in R^{m \times n}$ (we will call it scheduling matrix). We need to find permutations of the numbers 1, ..., m, $\pi_j = (\pi_j(1), ..., \pi_j(i))$ for $j = 1, ..., n$ such that a certain Schur-convex function $f(s_1, ..., s_m)$ is minimal (where s_i is the sum of elements in i-th row $s_i = \sum_{j=1}^{n} a_{\pi_j(i),j}$). The most used Schur-convex functions are:

- $f_{sqr}(s_1, ..., s_m) = s_1^2 + s_2^2 + ... + s_m^2$
- $f_{dif}(s_1, ..., s_m) = \max(s_1, ..., s_m) - \min(s_1, ..., s_m)$
- $f_{max}(s_1, ..., s_m) = \max(s_1, ..., s_m)$
- $f_{sqr}^{\delta}(s_1, ..., s_m) = (s_1 - \delta)^2 + ... + (s_m - \delta)^2$ where $\delta = \frac{(s_1, ..., s_m)}{m}$

In [7] it was shown that MPP is *NP*-hard except two column case for which the polynomial algorithm was found ([2]).

We can gain a generalisation of MPP, if we will not limit ourselves to column permutations and permutations will be allowed by a certain graph (its vertices represent elements of a scheduling matrix, element $a_{i,j}$ is represented by the vertex $v_{i,j}$). We will call it *regular permutation scheduling on graphs* (RPSG). In [3] there were given two ways how to generate needed permutations. We can use:

1. graph of permitted moves,
2. graph automorphisms.

(The graph of permitted moves will be denoted GPM and the related problem GPM-P.)

2. Graph of permitted moves

In this section we will continue in our work started in [3]. It is known from this work that MPP can be solved as a special case of GPM-P and if GPM is a complete digraph with loops on every vertex, two-column case can be solved for irregularity measure fsqr as a minimal perfect matching in a certain complete graph. Unfortunately there exist graphs of permitted moves for which two column case can't be solved as a minimal perfect matching in a graph (it was one of the open problems introduced in [3] related to two column GPM-P).

Example 1. *In Figure 1 we can see the graph of permitted moves G_M and related graph G whose vertex set is the same as the vertex set of G_M and vertices $v_{i,j}$, $v_{k,l}$ are connected by an edge of weight $w = (a_{i,j} + a_{k,l})^2$ if and only if there exist directed edges $(v_{i,j}, v_{x,y})$ and $(v_{k,l}, v_{x,z})$ in G_M ($y, z \in [1, 2]$ $y \neq z$) it means that $a_{i,j}$ and $a_{k,l}$ could form the row in a new matrix). We can see that for perfect matching with edges $v_{1,1}$, $v_{1,2}$, $v_{2,1}$, $v_{2,2}$ and $v_{3,1}$, $v_{3,2}$ there doesn't exist any permitted permutation of elements of the matrix A for which elements $a_{1,1}$, $a_{1,2}$ form some row of new scheduling matrix and elements $a_{2,1}$ $a_{2,2}$ another one.*

If we use for two column case another Schur-convex function f and we can transform GPM to a graph in which we find perfect matching then the solution of GPM-P will be perfect matching that minimises f.

Example 2. *There is given a matrix $A_{3x2} = (a_{i,j})$, GPM is a complete directed graph with loop on every vertex and fmax $(s_1, s_2, s_3) = max\ (s_1, s_2, s_3)$ is the irregularity measure. We can solve this problem as finding the perfect matching with minimal $f_{max}(s_1, s_2, s_3)$ in a complete graph with the same vertex set as GPM has (where s_1, s_2, s_3 are weights of edges used in this matching and edge $[v_{i,j}, v_{k,l}]$ has weight $a_{i,j} + a_{k,l}$).*

* **Peter Czimmermann**
University of Žilina, Faculty of Managements and Informatics, Slovakia, E-mail: petocimo@frcatel.fri.utc.sk

We present an algorithm that finds perfect matching in the graph minimizing the function f_{max} in graph $G = (V, E)$. (A similar problem is presented in [6] p. 260.)

1. let $n = |V|$ is even and $m = |E|$
2. sort the edges into nondecreasing sequence S

$$S = (\underbrace{e_1, ..., e_{i_1}}_{c_1}, \underbrace{e_{i_1+1}, ..., e_{i_2}}_{c_2}, ..., \underbrace{e_{i_{k-1}+1}, ..., e_{i_k}}_{c_3})$$

where $i_k = m$ and $c_1, ..., c_k$ are weights of the edges

3. find $j \in \{1, ..., k\}$ for which $i_{j-1} + 1 \leq n/2 \leq i_j$
4. if such j does not exist *then* STOP (there is no perfect matching in G) else let $r := j$
5. delete every edge e in G for which its weight $c(e) > c_r$ (we obtain a graph

$$G' = G - \{e; e \in E, c(e) > c_r\})$$

6. for every edge $e = \{u, v\}$ $(u, v \in V)$ of weight $c(e) = c_r$ do:
   - let $G'' = G' - u - v$
   - find perfect matching in G''
   - if M is perfect matching in G'' then STOP (edge e and edges of M form perfect matching in G for which is function f_{max} minimal)
7. if $r < k$ then $r := r + 1$ and go to 5 else STOP there isn't any perfect matching in G.

It isn't hard to show that the presented algorithm is polynomial. In step 2 it makes $O(n^2 \log n)$ steps. An $O(n^{5/2})$ algorithm is known for finding perfect matching in graphs [4, 6] and this algorithm must be made $m < n^2$ times at most so that our algorithm has complexity $O(n^{9/2})$.

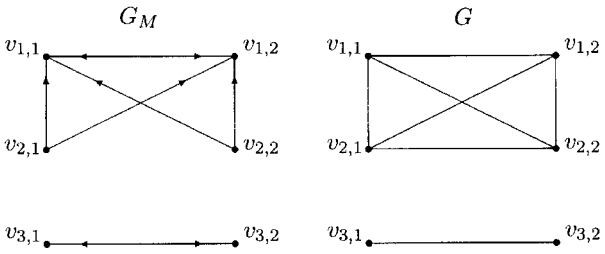


Fig. 1. Graphs GM and G

3. Conclusions

In this paper we dealt with using the graphs in regular scheduling problems. It seems that perfect matching that is minimal with consideration to some Schur-convex function is an important concept. For most Schur-convex functions f it remains an open problem to find an polynomial algorithm for perfect matching that is minimal with consideration to f. Similarly, two column case is unsolved if perfect matching can't be used for representing a convenient solution.

The research of the author is supported by Slovak Scientific Grant Agency under grant NO.1/0490/03.

References

[1] ČERNÝ, J., KLUVÁNEK, P.: *Principles of Mathematical Theory of Transport* (in Slovak), VEDA, Bratislava, (1991)
[2] ČERNÝ, J., VAŠEK, K., PEŠKO, Š., PALÚCH, S., ENGELTHALLER, D.: *Transport schedulings and their optimization* (in Slovak), Research report III-8-9/03, Research Institute of Transport, Žilina, (1986)
[3] CZIMMERMANN P., PEŠKO, Š.: *The Regular Permutation Scheduling on Graphs,* Journal of Information, Control and Management Systems, vol. 1, (2003)
[4] EVEN, S., KARIV, O.: *An $O(n^{5/2})$ Algorithm for Maximum Matching in General Graphs*, Proc. 16th Annual Symp. on Foundations of Computer Science, IEEE, New York, (1975)
[5] PEŠKO, Š., VAŠEK, K.: *Optimization of Transport Scheduling* (in Slovak), Research report III-8-6/09.3, Research Institute of Transport, Žilina, (1983)
[6] PLESNÍK, J.: *Graph Algorithms* (in Slovak), VEDA, Bratislava, (1983)
[7] TEGZE, M., VLACH, M.: *The Matrix Permutation Problem,* Tech. Univ. Graz Bericht 84-54, (1984).

Karel Šotek – Hynek Bachratý – Ján Ružbarský – Viliam Tavač *

NEW REAL ENVIRONMENT SIMULATION MODELS ON RAILWAY NETWORK

Transport network simulation models become inseparable part of the innovation processes in transport technology. They approach current operation especially in conjunction with the realization of the information and control systems. References about partial applications were published in various occasions. The authors try to offer integrated up-to-date information about result of their work concerning described area. This name is used for the models whose parameters and data structures encompass terms and parameters adapted from the real system in a considerable amount.

1. Introduction

Transport network simulation models become an integral part of innovation processes in transport technology. They approximate a current operation especially in conjunction with realization of information and control systems. References concerning partial applications were published on various occasions. The authors try to offer integrated up-to-date information about the results of their work in this area.

2. Real Environment Models

Models the parameters and data structures of which encompass terms and parameters adapted from the real system to a considerable amount are referred under this name.

With regard to railway operation, the first concept of network diagram is mostly concerned. Nodes of the diagram represent the stations and edges represent track sections. The topology of the diagram matches the topology of the railway network or the topology of the monitored interactions. The numbers of station tracks and line tracks, other capacity data and elements of operation technology represent other typical attributes. Consecutive detailing can describe the length of tracks, crossing intervals, train data, description of track sections between stations, capacity parameters of the line tracks, running times, track profiles, headways etc.

Operation on railway infrastructure can be simulated on a structure described above in different ways – mostly by train movements. Their movement is mostly described as a sequence of used nodes and edges, whereas the most often tracked data, e.g. runnig times or headways of trains, are mostly interpreted by the occupation time of these objects. Generally the number of the train movements realized in the model can characterize the operation. More complex models target more complicated processes of the railway operation, e.g. railway operation control, lock-out activities, etc.

A significant element of real environment's models is an ability of their close connection with the existing transport system. Retrospectively many parameters for the construction of a hypothetic model of a realistic range can be obtained by the analysis of the existing system. Such procedure ensures the high reliability and authenticity. Possibilities of direct connection with railway operation control information systems are used successfully in the operation, addressing railway timetable construction, train assembling, rolling stock and crew circulation plans and also operational management systems.

Moreover it offers the possibility to use real data records for some components of the model (e.g. infrastructure, operation, running time, etc).

In comparison with more common (theoretical) models, the problems with the applicability and credibility become non-existent thanks to generalization.

The interconnection between a model and a real system can be also seen in possibilities of practical application of obtained results. Frequent application is e.g. in the process of quality evaluation of the real transport systems and their operation (the quality of the railway timetable, etc.)

The model is often created targetting a specific solution on the basis of operation requirements.

3. Basic data

- Information describing railway infrastructure (basic data):
  - description of the railway network (mainly at the level of railway track, permanent and temporary speed limits, gradient attributes, etc.).

* [1]Karel Šotek, [2]Hynek Bachratý, [2]Ján Ružbarský, [2]Viliam Tavač
[1]Department of Informatics in Transport, Transport Faculty of Jan Perner, University of Pardubice, Studentská 95, 532 10 Pardubice,
E-mail: Karel.Sotek@upce.cz
[2]Department of Software Technologies, Faculty of Management Science and Informatics, University of Žilina. Moyzesova 20, 010 26 Žilina,
E-mail: hynek.bachraty@fri.utc.sk, jan.ruzbarsky@fri.utc.sk, viliam.tavac@fri.utc.sk

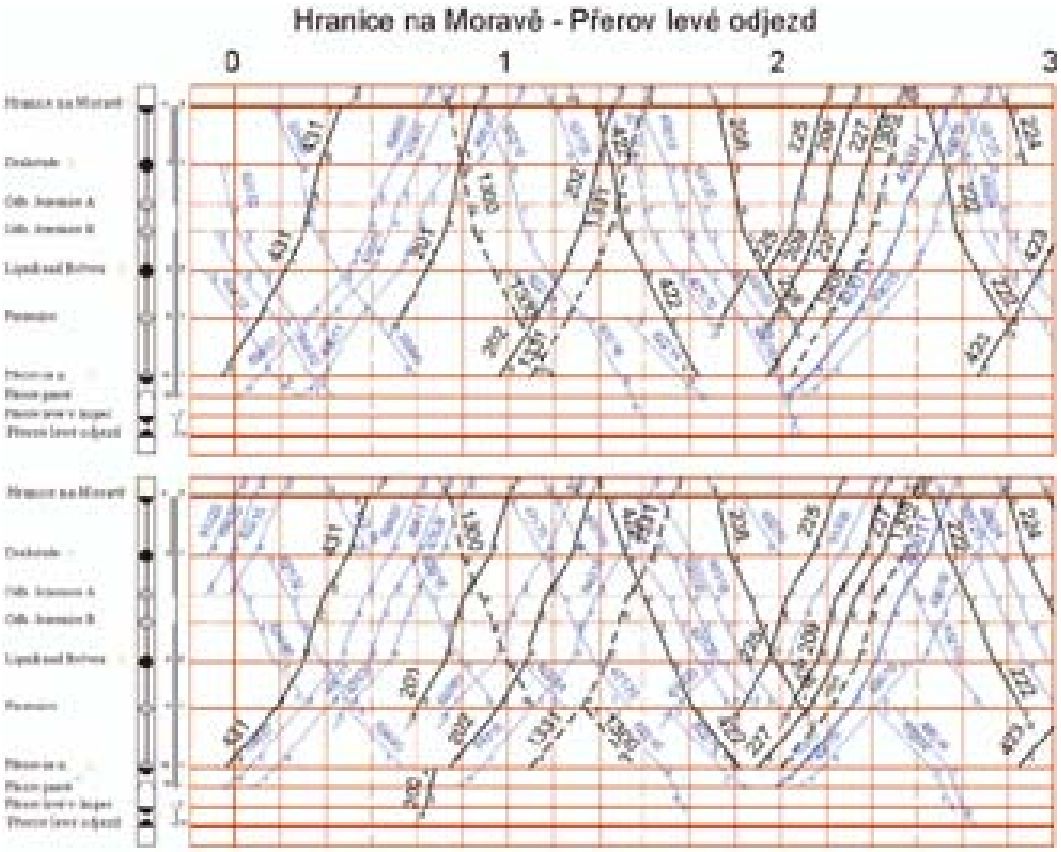


Fig. 1. Variant graphic timetable, April and August 2000 – line section of the 2nd corridor

Fig. 2. Speed-distance diagram of the train from graphic timetable

- Information about railway vehicles:
  - basic characteristics,
  - additional features,
  - number of vehicles used.
- Railway operation data:
  - running times, traction energy consumption characteristics of the train movement
  - train characteristics from the point of view of the commercial requirements,
  - train characteristics from the point of view of the operational level (rolling stock and crew's circulation plans).
- Organization and management of the railway operation:
  - timetable construction,
  - train planning, train routings on a railway network,
  - service optimalization of transport nodes (from the point of view of a single node and also the whole network),
  - real time operational control.

4. Key tasks and algorithms

Some areas of problems must be solved at a high level to enable a construction of the realistic models:

- Algorithm for calculation of running times which requires relatively complicated input data preparation. It must be possible to calculate also the consumption of traction energy needed for the train movement.
- Algorithm for crossing intervals in stations and headways in line sections, which is based on algorithm for running time calculations and technological service times of the signalling and interlocking equipment. The algorithm must be connected with real data.
- Algorithm for searching and solving conflicts in railway operation. It includes conflicts in nodes and line sections.
- Investigation of characteristics and evaluation of the timetable quality, construction and evaluation of an alternative timetable, construction of the lock-out timetables.
- Analysis of disturbation of the regular operation by large lock-out activities (track reconstruction, civil engineering works).

5. Several practical examples of the application of the real environment model in railway transport

- Construction of the conflictless train timetable:
  A large segment of the main line was loaded with data from the valid operational timetable which according to the valid methodology of the Czech Railways contained some conflict situations (e.g. conflicting train paths). The algorithms for searching and solving conflicts were applied to these data iteratively. The goal was to achieve a timetable without any conflicts. This goal was accomplished. The capacity parameters of the line (the occupation of the station and line tracks, number of spare train paths for different train types etc.) were analyzed and verified on this modified timetable.
- Construction of the lock-out timetable due to large building activities:
  A model of the variant lock-out line timetables for line sections during reconstruction (1st and 2nd transit corridor). SENA-JŘ-VT project. Specific solution for the needs of Infrastructure Division ČD (Fig. 1).
- Model of train movement expressed in the form of the speed-distance diagram. Each train path drawn in graphic timetable can be converted into the speed-distance diagram (Fig. 2). SENA-JŘ-VT project.
- GTN-DOZ project (Graphic-technological extension of remotely controlled signalling and interlocking equipment AŽD). A display

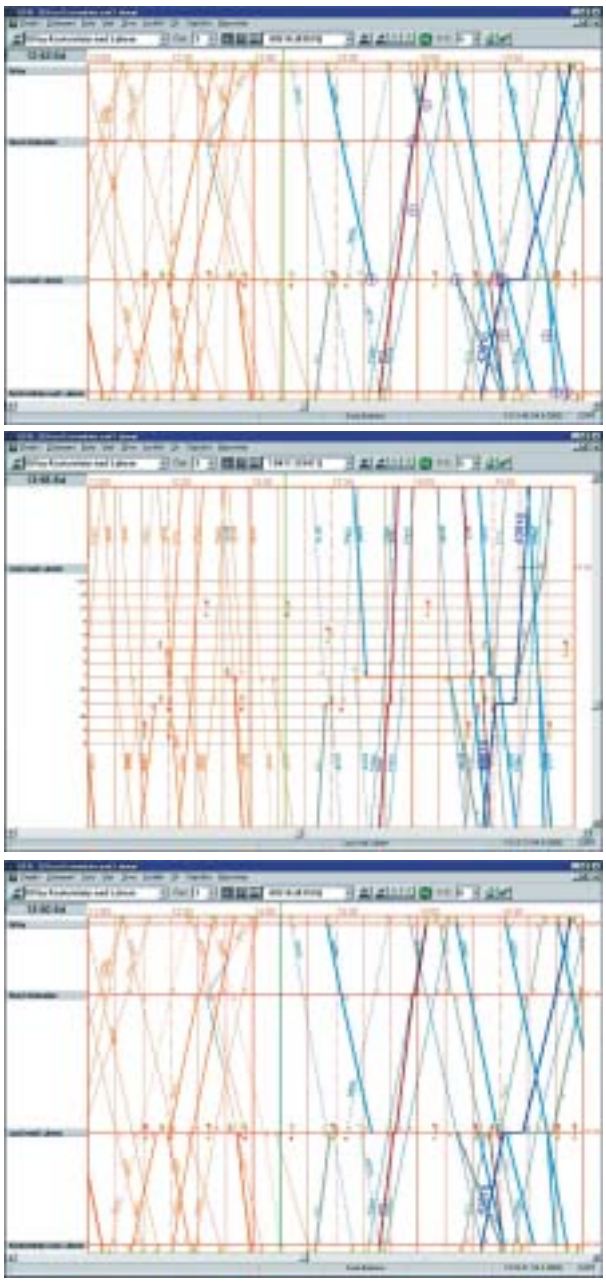


Fig. 3 (a-b-c) Examples of working environment of the GVD-DOZ application

of the real operational situation in the controlled line section and the railway station is an integral part of GTN module. This model not only monitors the operational situation, but also generates prognosis of the operational situation in the controlled line section for a given period (mostly two hours) based on the obtained data. As a matter of fact it is one of the first simulation models operating in real time in the environment of our railway transport (Fig. 3).
- SENA simulation module.

It enables construction of variant models of railway operation on the basis of an alternative timetable.

Key features of the modeled system:
- Calculation of the crossing intervals and headways
- Searching and solving of conflicts
- Calculation of theoretical and regular running times
- Calculation of traction energy consumption and power demands of the timetable
- Ability to enter train paths diverted onto alternative routes (mainly during lock-outs)
- Possibility to work with more variants of the infrastructure data
- Possibility to work with more variants of the train data
- Algorithms for input of initial delays of trains and irregularities of train movement
- Possibility to modify pre-set priorities of the trains, conditions for diverting and rerouting train paths to the alternative routes or cancellation of the trains, possibility to substitute the passenger train on the part its route by bus transport, etc.
- Resources for monitoring of simulation and evaluation of its results
- Resources for setting-up simulation parameters and database administration
- Basic time loop of simulation
- The present infrastructure database of the network of the Czech and Slovak railways and real or planned timetable can be used as a input database for simulation
- Wide and rich features of the IS SENA can be used both for entering input data of simulation and presentation of its results, too.
- The level of detail and depth of the description of the infrastructure and train database meets challenging requirements of the IS SENA
- Future development of the simulation model is in progress in close co-operation with operators of the customer

The simulation run and all relevant interventions into simulation by means of changing input data are continuously recorded into simulation log which can be viewed in graphic form (see Fig. 4 and Fig. 5). Further resources for statistical and qualitative evaluation, for example:
- Delay on entry into and exit out of the simulated area (for separate trains and sum total, too)
- Values of running times and stopping times within the simulated area (for separate trains and sum total, too)
- Occupation times of stations and particular tracks
- Power consumption (gas and electricity)

Parity of train numbers, direction of traffic, course (trains running between particular stations) can be specified. The results are available in text or spreadsheet files forms. New types of statistics are being prepared as the field od simulation model usage extends.

The user controls the resources for preparation and management of simulation data. If he uses graphic timetable construction data, he can remove various flags and data used only during the timetable construction, reduce timetable data into simulation area only, etc. A further possibility is to copy data of the “master” trains into three “simulation” variants and by the gross recalculate the running times of all trains (when the infrastructure data has been changed), etc. These service algorithms are being completed according to actual requirements.

Particular running of individual simulations can be influenced by input of some parameters. So far the user can select different types of searched and solved conflicts during simulation, specify the lines or the line sections submitted to simulation, etc. Next development of the simulation model will encompass extended set of optional parameters, which will enable selecting “style” of conflicts’ solutions.

Essentially the proper simulation consists of constructing the real transport plan, i.e. the plan without any conflicts, which is based on real data from the simulated area. The operation consists of three basic stages. In the first stage the user prepares basic infrastructure and train data for the simulation. They can be based on valid timetable or can be created by the user. This proposal can represent both conflicting or conflictless transport plan. During the second stage the user can modify these input data by means of infrastructure data update, applying of lock-outs, incorporating delays and irregularities of trains’ running times, adding or cancelling trains, etc. The third stage is the proper simulation consisting of iterative searching and solving of traffic conflicts. Conflicts are solved in order of their origin in time with regard to pre-set parameters. The user specifies time range of the simulation and the simulation time step. The simulation can be inerrupted after each step, the user can change the simulation parameters or data and then resume it.

6. Conclusion

The shown results prove the applicability of the real environment simulation models as parts of information systems used in daily operation. Customers keep at their disposal an effective tool, which makes possible not only evaluating the quality of IS outputs, but also essential extending of their practical application.

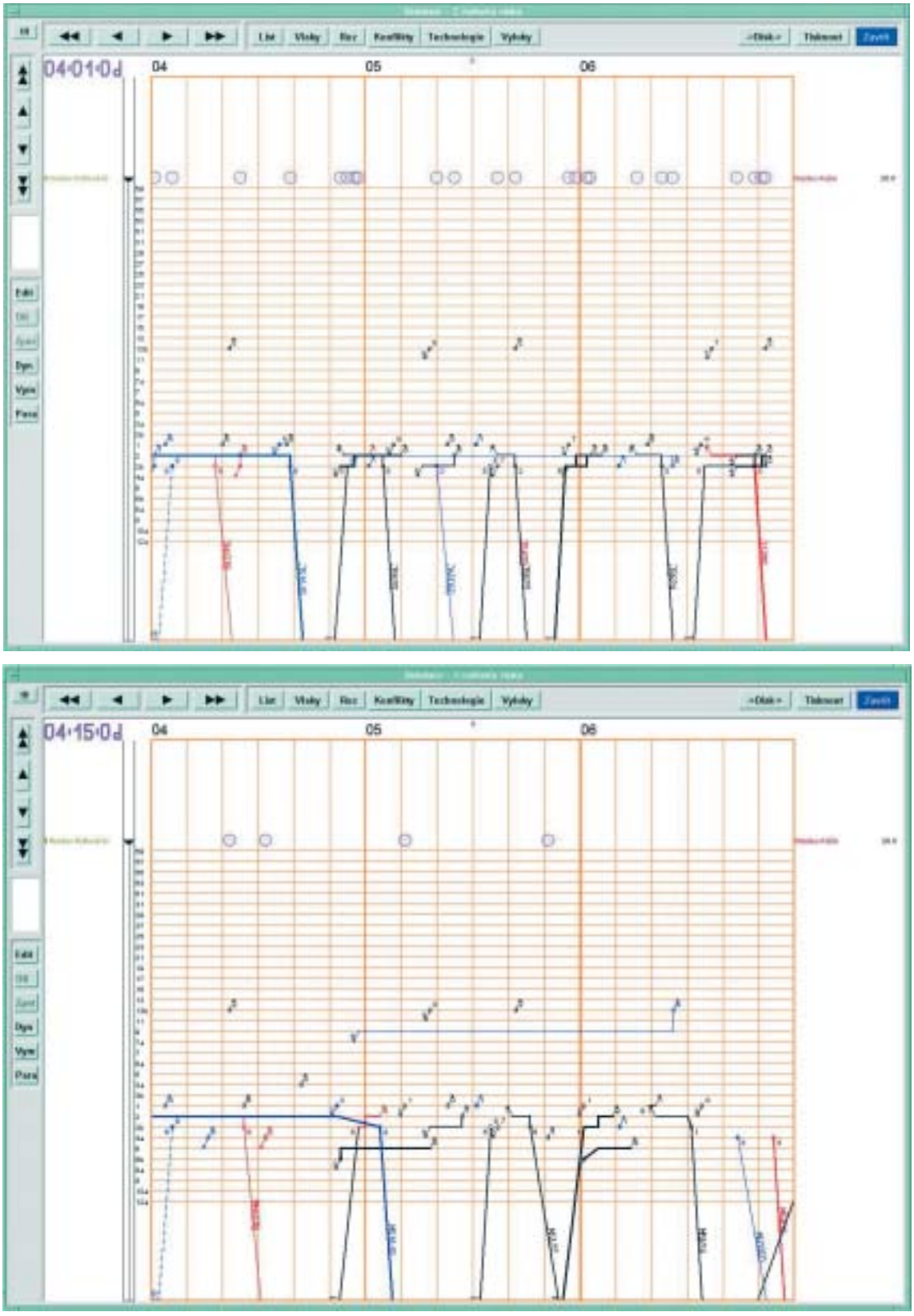


Fig. 4 and 5. Station conflicts before and after simulation

References

[1] ŠOTEK, K., BACHRATÝ, H., GREINER, K., TAVAČ, V.: *New Modules of SENA System,* Žel 2001, 8. International Conference, May, 29-30, 2001, Žilina, Slovak Republic, p. 251-257.
[2] ŠOTEK, K., SADLOŇ, L.: *Simulation Models of the Real Environment In Transport,* MOSIS'01, Modeling and Simulation of Systems, 35th Spring International Conference, Hradec nad Moravicí, May, 9-11, 2001, p. 61-67.

Dušan Marček – Michaela Ačová *

ECONOMIC TIME SERIES FORECASTING: BOX-JENKINS METHODOLOGY, SIGNAL PROCESSING AND NEURAL NETWORK APPROACH

Abstract: This paper is devoted to the presentation of methods of economic time series analysis and modelling using the Box-Jenkins methodology, the signal processing approach and the feedforward neural network technique. Some results of our research on time series modelling with emphasis on potential improving forecast accuracy are presented here. The assessment of the particular models has been made using the root mean square error.

1. Introduction

Artificial neural network (ANN) is now being applied to many problems far removed from their first beginnings. Application to management problems have included predicting bankruptcy [3], predicting ratings of corporate bonds [11], forecasting financial markets [6] and time series forecasting [7]. Their main strengths lie in pattern recognition and have been a hot topic of research for many years now.

There is much controversy about the application of traditional statistical or econometric models and the ANN approaches within the field of economic time series modelling and forecasting. These controversies are based on the assumptions that there is no consensus at all on whether there is chaos in economic time series or not. Various tests for nonlinear pattern and chaos in time series have been proposed to illustrate the nonlinear nature of certain processes. A survey of these tests is presented in [1].

The goal of this paper is to illustrate those three areas: probabilistic, adaptive signal processing and computational networks may be used to economic time series modelling. In Section 2, we can see that a random process of time series of stock prices may be generated as the output of linear filter driven by white noise. Section 3 is focused on the behaviour of the GL filter and LSL when they are used to forecast future observations of stationary AR processes. In Section 4 of this paper, we report on an ANN application that was designed and run by [9] to investigate the problem of forecast accuracy across proposed models.

2. Application of the Box-Jenkins methodology in the stock prediction problem

In this section, we give an example that provides one kind of possible results. We will regard these results as referential values for the approach of adaptive signal processing procedures and ANN modelling. Many of modelling techniques of autoregressive processes are based on recent developments in time series analysis recently consolidated and presented by Box and Jenkins [4].

To illustrate the Box-Jenkins methodology, consider the stock price time readings of a typical company (say VAHOSTAV company). We would like to develop a time series model for this process so that a predictor for the process output can be developed. The data was collected for the period January 2, 1997 to December 31, 1997, which provided a total of 163 observations (see Fig. 1).

To build a forecast model the sample period for analysis y_1, ..., y_{128} was defined, i.e. the period over which the forecasting model was developed and the ex post forecast period (validation data set), y_{129}, ..., y_{163} as the time period from the first observation after the end of the sample period to the most recent observation. By using only the actual and forecast values within the ex post forecasting period only, the accuracy of the model can be calculated.

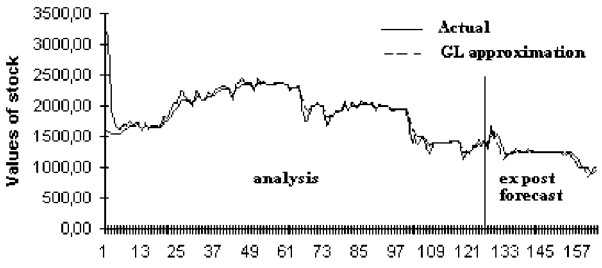


Fig. 1 The data for VAHOSTAV stock prices (January 1997 - December 1997) and the values of the AR(7) model for VAHOSTAV stock prices estimated by GL algorithm

To determine appropriate Box-Jenkins model, a tentative ARMA model in identification step is identified. In Figure 2, the estimate

* [1]Dušan Marček, [2]Michaela Ačová
[1]Faculty of Management Science and Informatics, University of Zilina, E-mail: marcek@fria.utc.sk
[2]Institute of Computer Science, Silesian University Opava, E-mail: michaela.acova@fpf.slu.cz

of autocorrelation ($\hat{r}_k$) and partial autocorrelation $\hat{a}_{kk}$ function (ACF, PACF) of the data are given. To test whether the autocorrelation and partial autocorrelation coefficients are statistically equal to zero, we use the t-statistic $t_r = \hat{r}_k / S(\hat{r}_k)$ and $t_a = \hat{a}_{kk} / S(\hat{a}_{kk})$ where

$$S(\hat{r}_k) = N^{-1/2}\left[1 + 2\sum_{j=1}^{k-1}\hat{r}_j\right]$$

and

$$S(\hat{a}_{kk}) = N^{-1/2}$$

denote standard errors of the kth sample autocorrelation or partial autocorrelation coefficient, respectively, N is the number of data points, k is the lag. Since the ACF decays in an exponential fashion, and the PACF truncates abruptly after lag 2, we may tentatively identify the model for this time series as AR(2).

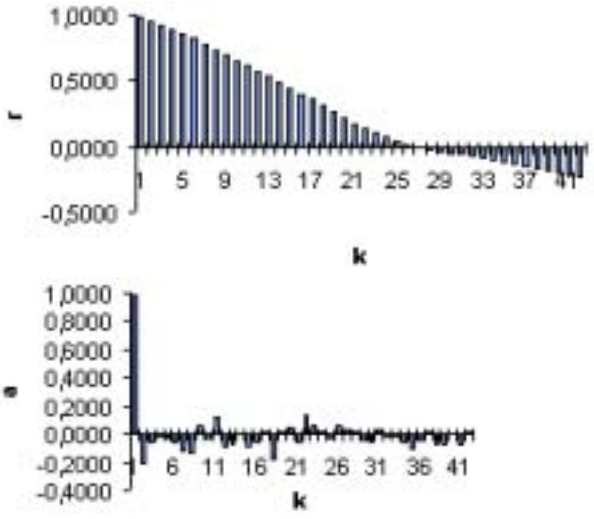


Fig. 2 Autocorrelation function and partial autocorrelation function of the data for VAHOSTAV stock prices (period for analysis)

In the estimation step, we compute estimates for the parameters of the AR(2) model

$$y_1 = \xi + a_1 y_{t-1} + a_2 y_{t-2} + \epsilon_t \quad t = 1, 2, ..., N-2 \tag{1}$$

or with of obvious matrix notation

$$y = Xa + \epsilon$$

by OLS (Ordinary Least Squared)

$$\hat{a} = (X'X)^{-1}X'y = \begin{bmatrix}\hat{\xi}\\ \hat{a}_1\\ \hat{a}_2\end{bmatrix} = \begin{bmatrix}26.693\\ 1.113\\ -0.127\end{bmatrix} \tag{2}$$

In the diagnostic checking step, we test adequacy and closeness of fit of the model to the data by sample autocorrelation function of the residuals say

$$e_t = y_t - \hat{y}_t, \quad t = 1, 2, ..., N-2 \tag{3}$$

The sample autocorrelation function of the residuals is shown in Fig. 3.

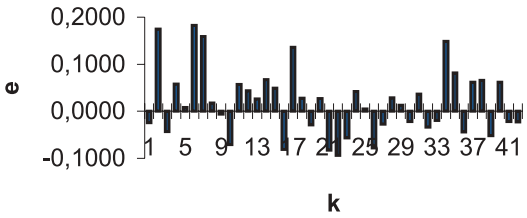


Fig. 3 Sample autocorrelation function of the residuals for model (2)

The modified Box-Pierce statistic Q is used for collectively testing the magnitudes of the residual autocorrelations for insignificance. The statistic is [5]

$$Q = (N - d)\sum_{k=1}^{K} r_{ek}^2 \tag{4}$$

where r_{ek}^2 is the square of the residual autocorrelation coefficients, for lags $k = 1, 2, ..., K$, d is dth differences of the data. For our stock price time series the Box-Pierce statistic for lag $k = 42$ was computed to be 27.78. This value is less than the critical chi square value of 55.7585 (degrees of freedom is $42 - 2 = 40$, $= 0.005$). Hence, we can conclude that the error terms are random and the model (1) is an adequate model.

3. Stock price prediction using adaptive signal processing procedures

In practice, the modelling of a set of data as we shown in Section 2, is a much more complex process than the one of fitting and testing. Most of deterministic methods to signal processing have the goal of representing a given sequence $\{y_t\}$ as the impulse response of a rational linear system [2], [12]. The AR model involves a linear filter with transfer function $H(z)$, where

$$H(z) = [A(z)]^{-1} = \frac{1}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{5}$$

and is also known as the all-pole model which has only a nontrivial denominator polynomial, generating the random process $\{y_t\}$ from the white noise $\{\epsilon_t\}$. The linear filter is represented by the inverse of the polynomial $[A(z)]^{-1}$. The difference equation for the input-output relationship for this filter is

$$y_t = \sum_{k=1}^{p} a_k y_{t-k} = \epsilon_t \tag{6}$$

or

$$y_t = \sum_{k=1}^{p} a_k y_{t-k} + \epsilon_t \tag{7}$$

where a_k are the filter parameters that determine the location of the poles of linear filter $H(z)$. To forecast the observation we can

take expectation at origin $t-1$ of the model (7) written at time $t+1$, namely

$$E[y_t] = \hat{y}_t = -\sum_{k=1}^{p} a_k y_{t-k} \tag{8}$$

For selecting the model order p, we will now monitor prediction Mean Square Error (MSE) of the model (8). If the data is truly described by a finite-order AR model, then the theoretical MSE becomes constant once the model order is reached. A realization of this criterion is shown in Fig. 4.

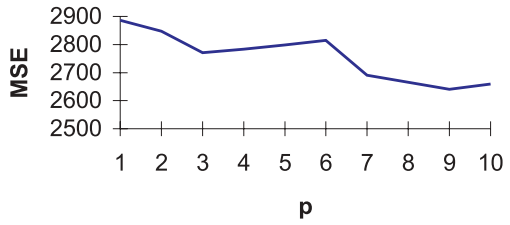


Fig. 4 Graph of MSE (analysis period) versus model order p.

As can be seen from Fig. 4, the MSE's seen to become constant at $p \geq 7$. Then the linear filter (7) has the form

$$y_t = -\sum_{k=1}^{7} a_k y_{t-k} + \epsilon_t \tag{9}$$

The final estimates of predictor parameters (9) are obtained using two of adaptive filtering algorithms in signal processing. The Gradient Lattice (GL) adaptive algorithm and Least Squares Lattice (LSL) form for the parameter estimates of the predictor (9) were used. These algorithms are coded in MATLAB and described in [2]. In this case the process of calculating predictor parameters are updated as each new data point becomes available to track the changing statistics. In Tab. 1 parameters of AR process and the corresponding RMSE's (Root Mean Square Errors) for models (1) and (9) are given.

The RMSE's are called standard deviations of the single-period-ahead forecast errors. For this measure, the AR models estimated by OLS and GL procedures not exceed 5 % limit of the variation coefficient, while the variation coefficient (V) computed as $V = RMSE/\overline{y}$, where $\overline{y}$ is the mean of the stock price time series, for AR model estimated by LSL procedure is 5.59%. Fig. 1 shows the GL prediction results and actual values for stock price time series in both analysis and ex post forecast period. The GL approximations in both intervals visually match the actual stock series quite well. However the corresponding RMSE value for analysis period is 214.48. This is greater than that for ex post-forecast period. The RMSE statistic is here not adequate, because most errors are fairly small, i.e. the model is a good fit to the historical data but there are only first three large errors, these are magnified by using RMSE (since all the errors are squared). This phenomena is produced by GL algorithm itself and this does not influence the ex post forecast errors.

4. Neural network approach

The structure of an ANN is defined by its architecture, its activation functions and learning algorithm. While many variations are possible we suggested an alternative of the most common form of ANN which was suggested and discussed in [8]. This alternative of ANN is pictured in Fig. 5.

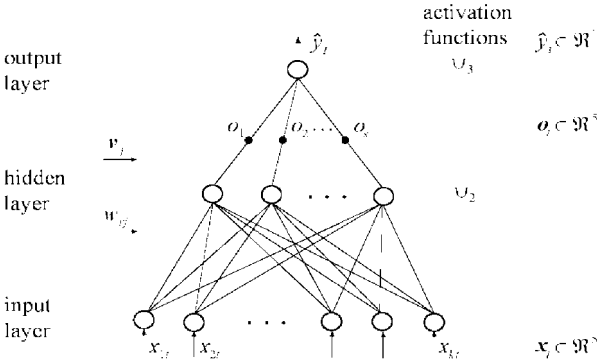


Fig. 5 Fully connected single hidden layer network

Fig. 5 shows a fully connected and strictly hierarchical ANN with variational number of inputs, further variational number of hidden layer units and one output unit. Processing units of the hidden layer have an activation function S - shaped tanh, which produces values of outputs o_j, $j = 1, 2, ..., s$ ranging from -1 to 1. Processing units of hidden layer have the associated weights w_{rj}, $r = 1, 2, ..., k$, $j = 1, 2, ..., s$. Input data x_r of the ANN are standardized variables. The standardized version of the variables is created in a data-preprocessing unit. The system in a preprocessing unit subtracts the mean of the variable from each observation in the variable and divides the result by the standard deviation of that

OLS, GL and LSL estimates of AR models Tab. 1

Model	Order	Est.proc.	$\hat{\xi}$	$\hat{a}_1$	$\hat{a}_2$	$\hat{a}_3$	$\hat{a}_4$	$\hat{a}_5$	$\hat{a}_6$	$\hat{a}_7$	RMSE*
(1)	2	OLS	26.639	1.113	−0.127						67.787
(1)	7	OLS	45.930	1.085	0.0861	−0.2531	0.0836	−0.0057	0.2081	−0.2281	76.548
(9)	7	GL		−0.7513	−0.1701	−0.0230	−0.0128	−0.0028	−0.0472	0.0084	68.540
(9)	7	LSL		−0.8941	−0.6672	0.7346	−0.2383	0.1805	−0.5692	0.4470	94.570

*ex post forecast period

variable. After standardization all input variables x_r have values ranging from -1 to 1 and the bias equals to zero. Hidden layer weights w_{rj} are estimated from data according to the learning technique and choice of measure of accuracy in any ANN application. The processing units of hidden layer produce output values o_j such as

$$o_j = \tan H\left(\sum_{r=1}^{s} w_{jr} x_r\right) \quad j = 1, 2, ..., s$$

A dependent variable $\hat{y}$ is produced in an output unit. The output layer unit produces a dependent variable $\hat{y}$ so that the hidden layer outputs $o_j, j = 1, 2, ..., s$ are each multiplied by an additional parameter (weight) estimated from the data. These weights have a clear interpretation. They show how hidden layer outputs o_j contribute to the dependent variable $\hat{y}$ (total model). The dependent variable is transformed to the origin scale in postprocessing unit.

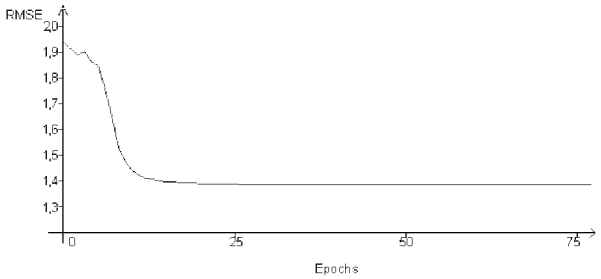


Fig. 6 RMSE's - validation set (normalized data)

Our ANN was trained on the training data set by Back-Propagation algorithm. Periodically, during the training period, the RMSE of the ANN were measured not only on the training set but also on the validation set. The final ANN chosen for the stock price prediction is the one with the lowest error on the validation set (see Fig. 6).

The RMSE's of our predictor models are shown in Tab. 2.

Tab.2

Model	RMSE*
AR(2) - OLS estimates	67.7
AR(7) - OLS estimates	76.5
AR(7) - GL estimates	68.5
AR(7) - LSL estimates	94.6
Neural network	67.2

* Validation set

According to the results of our experiments, the predictor based on the ANN forecasting model is the best, but only slightly better than the AR(2) model. As is stressed in [10], neural networks can outperform standard forecasting procedures at least for certain types of situations. Namely, where the relationship between inputs and outputs are highly nonlinear. Because the results were based on chosen stock price time readings, they were difficult to generalize in other situations. Yet, the results certainly provide a rational way for improvement of forecasting ability in nonlinear economic systems.

Acknowledgement

This work was supported by the Slovak grant foundation under the grant No. 1/0499/03, 1/9183/02.

References:

[1] BARNET, W., HE, Z.: *Unsolved econometric problems in nonlinearity, chaos, and bifurcation.* CEJOR (2001) 9, 147-182, Heidelberg: Physica Verlag, 147-182

[2] BAYAHAN, G. M.: *Sales Forecasting Using Adaptive Signal Processing Algorithms.* Neural Network World 4-5/1997, Vol. 7, pp. 579-589

[3] BELL, T. B., RIBAR, G. S., VERCHIO, J. R.: *Neural nets vs. Logistic regression: A comparison of each model's ability to predict commercial bank failures.* Presented at the 1990 Deloitte & Touche Auditing Symposium. University of Kansas.

[4] BOX, G. E. P., JENKINS, G. M.: *Time Series Analysis, Forecasting and Control,* Holden-Day, San Francisco, CA, 1970

[5] BOX, G. E., PIERCE, D. A.: *"Distribution of Residual Autocorrelation in Autoregressive-Integrated Moving Average Time Series Models".* Journal of the American Statistical Association, vol. 64, 1977

[6] FISHMAN, M. D., BARR, LOICK, W.: *Using neural networks in market analysis.* Technical analysis of Stocks and Commodities. 18-25

[7] HILL, T. L., MARQUEZ, M., O'CONNOR, REMUS, W.: *Neural network models for forecasting and decision making.* IntenationalInternational Journal of Forecasting 10, 1994, 5-15

[8] MARČEK, D.: *Neural networks and fuzzy time series with applications in economics.* Silesian University, 2002 (in Czech)

[9] MARČEK, D.: *Stock Price Prediction Using Autoregressive Models and Signal Processing Procedures.* Proceedings of the 16th Conference MME'98, Cheb 8.-10.9.1998, 114-121

[10] PELIKÁN, E.: *Time series forecasting by neural networks.* International Conference „Artificial neural networks and their application possibilities". TU in Košice, November 18.-19. 1996, 96-101

[11] SURKAN, A., SINGLETON, J.: *Neural networks for bond rating improved by multiple hidden layers.* International Joint Conference on Neural Networks. Vol 2, San Diego, CA, IEEE, New York, NY, 157-162

[12] THERIEN, C. W.: *Discrete Random Signals and Statistical Signal Processing.* Prentice-Hall, USA, 1992.

Marek Repcik *

VOIP TRANSMISSION QUALITY INCREASE USING ADAPTIVE PLAY-OUT BUFFER

In this paper we consider some quality aspects of VoIP (Voice over Internet Protocol) transmission. We try to increase the voice transmission quality using the adaptive play-out buffer instead of common buffer at the receiver's side. The hypothesis is that the less packet-loss probability increases the transmission quality more that the additional jitter decreases it. The aim of this paper is to formulate the problem.

1. Introduction

In this paper we consider some quality aspects of voice transmission over the Internet. Voice transmission over the Internet, or IP telephony, has been taking over by the analogue telephony because of its cheaper operation.

1.1 Problem

VoIP transmission consists of several steps described in ITU-T Recommendations. The most important steps are: equidistant voice sampling, coding, compression, equidistant packetization, transmission that violates the inter-packet equidistance, buffering, decompression, and finally playing out, Fig. 1, and Fig. 2 respectively.

Each step that has been performed at the voice processing decreases the quality of the transmission.

The main consequences of decrease of the transmission quality are: packet delay, jitter and packet loss.

Scientists have already investigated each single segment of the information chain in order to decrease the consequences of the

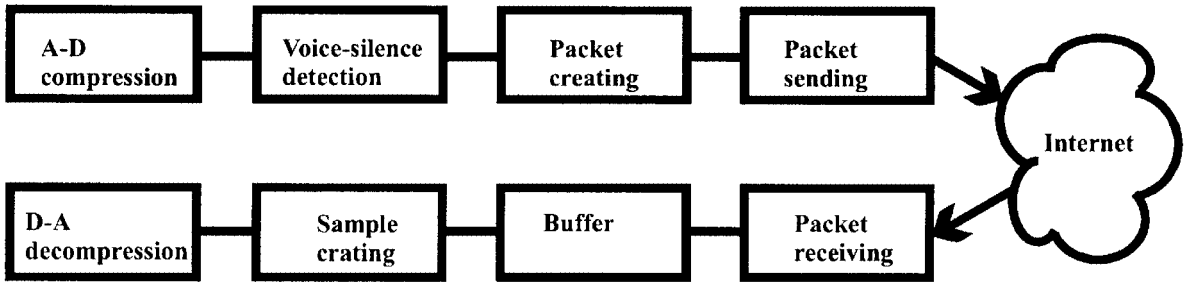


Fig. 1 Information chain of VoIP

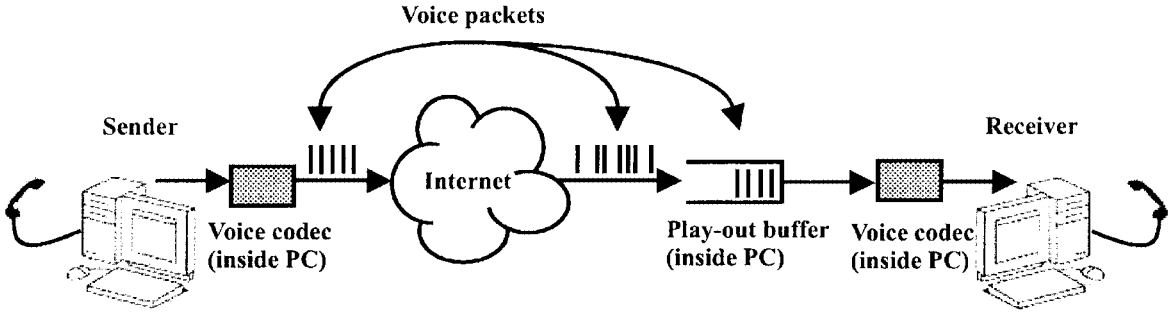


Fig. 2 Packet transmission over Internet.

* **Marek Repcik**
Department of Information Networks, Faculty of Management Science and Informatics, University of Žilina, Slovakia,
E-mail: pi@frcatel.fri.utc.sk

segment's negative influences. We focused to the buffer segment believing that our idea had not been studied yet.

What is a buffer? A buffer is an output memory, where the incoming packets have been stored until they can be preceded to be played out equidistantly at the receiver's side. What is the main task of a buffer?

1. A buffer helps to eliminate the random deflections (jitter) between the incoming packets that have appeared during the transmission process. Remember that the voice has been sampled in equidistant intervals at the sender's side and therefore it is reasonable to expect the equidistance at the receiver's side too.
2. A buffer decreases the packet loss.

How can we increase the voice transmission quality paying attention to the buffer segment? The way is – an adaptive play-out buffer.[1)]

What is an adaptive play-out buffer? It is a buffer, in which the incoming packets have been stored until they can be preceded to be played out at the receiver's side not equidistantly but adaptively.[2)] It means that the packet processing at the receiver's side has been performed once faster and once slower depending on the buffer load. When the buffer is fuller, the service intervals are shorter tending to empty the buffer and prevent the buffer overload and vice versa, when the buffer is emptier, the service intervals are longer tending to load the buffer and prevent the buffer underload.

What is a difference between a common buffer and an adaptive play-out buffer?

1. An adaptive play-out buffer generally creates a jitter because the adaptive play-out violates the inter-packet equidistance. The jitter causes a noise. This is a disadvantage of the adaptive play-out buffer.
2. There exists such an adaptive play-out buffer that produces less packet-loss than a common buffer. This is an advantage of the adaptive play-out buffer.

Because the common buffer is a special case of an adaptive play-out buffer, for sure there is an adaptive play-out buffer that causes at least the same voice transmission quality as the common buffer!

We should say something about other researchers dealing with adaptive play-out buffers. In [1] an adaptive play-out of packet voice is considered. The authors try to find out how much of a voice spurt should be buffered. They do not consider their adaptive play-out buffer the same way as we do. They are constricted only to a question how much the beginning of a voice spurt should be buffered (delayed) otherwise their buffer works as a common buffer. In [2] the approach is the same. Adaptive techniques perform continuous estimation of the network delays and dynamically adjust the play out delay at the beginning of each talk spurt. The play-out adjustment is performed during silent periods between talk spurts. The adjustment is done on the first packet of the talk spurt; all packets in the same talk spurt are scheduled to play out at fixed intervals following the play-out of the first packet. Similar approach is also in [3] and [4]. So these are definitely different approaches from ours. We continue and develop an approach mentioned in [5].

We can formulate the problem:

- to find the best set-up of an adaptive play-out buffer parameters such that the less packet-loss increases the transmission quality more than the additional jitter decreases it. Then the overall effect is a quality increase. What does it mean 'set-up of an adaptive play-out buffer'? It means a determination of service time lengths for particular buffer loads.
- to find out how much such an innovation (using an adaptive buffer) increases the voice transmission quality.

2. Way of Advancing

In order to find out the answers we may divide the problem into four steps:

1. to define the transmission quality function,
2. to make a mathematical model of voice transmission using the adaptive buffer,
3. to incorporate the model results into the quality function,
4. to optimize the buffer set up through maximizing the quality function through its parameters.

Let's say something more about each of these steps.

3. Voice Transmission Quality

To define and measure the quality of VoIP service we follow the ITU-T Recommendations. To determine the quality it is necessary to determine the relation amongst the QoS (*Quality-of-Service*) parameters and NP (*Network-Performance*) parameters.

The last recommendations G.175, G.107 and G.109 prefer expressing the NP parameters influence with the Transmission Rating Factor R, which is increasing with increasing quality. For its evaluation so called E-model is used. The E-model is based on the assumption that the all impairment influences can be transformed into the psychological factors whose influences are additive.

3.1 E-model

According to the recommendations ITU-T G.175 and ITU-T G.109 Definitions of Grades of Quality all influences at the voice

[1)] Adaptive play-out buffer in our meaning is significantly different from until now used adaptive play-out buffers but with its nature it still can be considered to be an adaptive play-out buffer.

[2)] We said that it is reasonable to expect the equidistance at the receiver's side but as we'll see later it's not exactly like that. Sorry for that:-).

transmission are expressed aggregately by the transmission rating factor R, which reaches the values in the interval $\langle 0, 100 \rangle$, where $R = 100$ means a very high quality and $R = 0$ means a very bad quality. Transmission with the parameter less than 50 is not recommended; the rest of the values is uniformly distributed in 5 classes.

3.1.1 Pre-Defined Values of Parameters and Acceptable Ranges

According to the recommendation ITU-T G.107 there are the default values of the input parameters of the E-model. There is a strict recommendation to accept these values for those parameters, which are not going to be changed during the planning process. If all of the default values are applied, the results lead to a very high value of the transmission rating factor $R = 94.1$.

See illustration of the general reference connection of the E-model in Fig. 3.

3.1.2 Calculation of the Transmission Rating Factor *R*

According to the equipment impairment factor method, the fundamental principle of the E-Model is based on a concept that all psychological factors on the psychological scale are additive. The result of any calculation with the E-Model in the first step is a transmission rating factor R, which combines all transmission parameters relevant for the considered connection, according to the recommendations ITU-T G.107 and ITU-T G.175. This rating factor R is composed of:

$$R = Ro - Is - Id - Ie + A \quad (1)$$

Ro represents the basic signal-to-noise ratio, including noise sources such as circuit noise and room noise. The factor *Is* is a combination of all impairments which occur more or less simultaneously with the voice signal. Factor *Id* represents the impairments caused by delay and the equipment impairment factor *Ie* represents impairments caused by low bit rate codecs. The advantage factor *A* allows compensation of impairment factors when there are other advantages of access to the user.

Using the adaptive play-out buffer just the parameters *Is* and *Id* will be touched. Why? As said before the factor *Is* is a combination of all impairments, which occur more or less simultaneously with the voice signal. The adaptive play-out buffer causes an additional jitter that causes a noise. The factor *Id* will be touched too, as it represents the impairments caused by delay. It is clear that for different adaptive buffer set-ups different delays of packets in buffer are expected.

3.1.3 Case of more impairment sources

The problem may be how to incorporate the adaptive play-out's influence to the *Is* impairment. We suggest to incorporate it to the *Q* parameter of the *Is* impairment as another impairment source, which is a result of noise created by the adaptive play-out.

Let $s(t)$, $t \in R$ be the original signal with the mean power σ_s^2, let $n_1(t)$, ..., $n_M(t)$ are independent noises with the mean powers $\sigma_{n_1}^2$, ..., $\sigma_{n_M}^2$. Then the overall signal-to-noise ratio Q is:

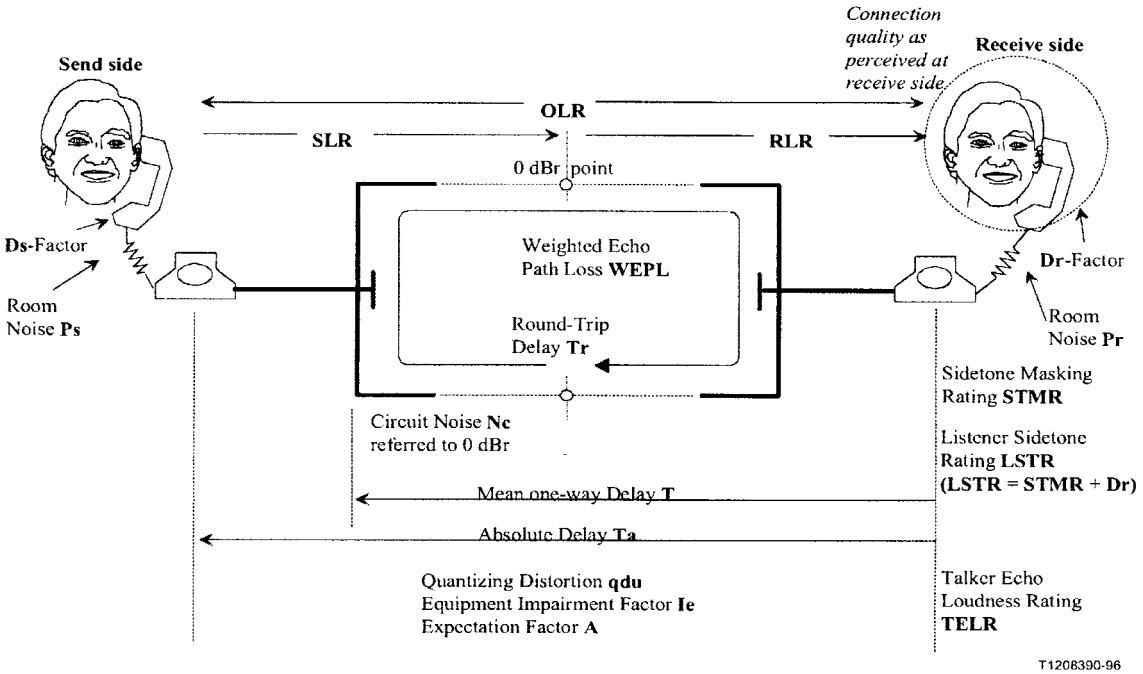


Fig. 3 General reference connection of the E-model.

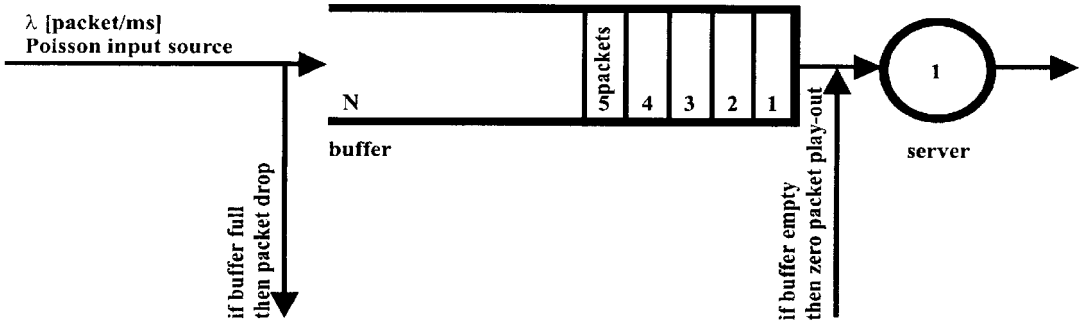


Fig. 4 Stochastic model $M|D_k|1|N$ for adaptive packet play-out process.

$$Q = 10\log\frac{\sigma_s^2}{\sum_{i=1}^{M}\sigma_{n_i}^2} = 10\log\sum_{i=1}^{M}10^{-\frac{Q_i}{10}} \qquad (2)$$

where $Q_i \overset{def}{=} 10\log\frac{\sigma_s^2}{\sigma_{n_1}^2},\ i = 1, 2, ..., M$.

According to ITU-T P.11, Annex E, for the overall signal-to-noise ratio it is recommended to use the 'correlated signal-to-noise ratio' Q, which considers the subjective factors and recommends combining the values Q_i on the $15\log_{10}$ base, i.e.:

$$Q_i \overset{def}{=} -15\log\sum_{i=1}^{M}10^{-\frac{Q_i}{15}} \qquad (3)$$

In our case, if *Qjl* is the signal-to-noise ratio, where the noise is caused by the jitter of the adaptive buffer play-out and the packet-loss, i.e. $Qjl \overset{def}{=} 10\log\frac{\sigma_s^2}{\sigma_{njl}^2}$

where *njl*(*t*) is the noise caused by the adaptive buffer play-out and packet-loss, then:

$$Q = -15\log\left[10^{-\frac{37}{15}}\,qdu + 10^{-\frac{Qjl}{15}}\right] \qquad (4)$$

4. Transmission Model

We need to model the voice transmission over the Internet using a stochastic model of bulk service.

Using the state probabilities of the model we should be able to determine:

- the noise caused by the jitter being caused by the network and by the adaptive play-out buffer,
- the delay of a packet in the adaptive buffer.

The model should describe:

- playing the packets out adaptively depending on the buffer load,
- refusing packets if the buffer is full,
- playing so called zero packets in a case that packets are late (if they haven't arrived to the time of their supposed playing-out). Playing a zero packet means that we play an artificial packet that contains samples of zero signal – silence, as it is preformed in reality.

We suggest a stochastic model $M|D_k|1|N$ where the input source is the Poisson process, the buffer capacity is equal to N and there is one server working deterministically and adaptively depending on the buffer load, see Fig. 4. The request of modelling the zero packet playing-out could be modelled by the negative states of the system. We realize that the Poisson process is not a good approximation of an input packet source but it is quite easy to analyse and it represents the worst case of an input source.

5. Conclusion

We have proposed an adjustment of the voice transmission over the Internet using an adaptive play-out buffer. We have suggested a way to model and determine whether and how much such adjustment is good and relevant.

References

[1] PINTO, J., CHRISTENSEN, K. J.: *An Algorithm for Playout of Packet Voice based on Adaptive Adjustment of Talkspurt Silence Periods,* Department of Computer Science and Engineering, University of South Florida, Tampa, Florida, 1999.

[2] NARBUTT, M., MURPHY, L.: *Adaptive Play-out Buffering for Audio/Video Transmission over the Internet,* Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland.

[3] ROSENBERG, J., QIU, L., SCHULZRINNE, H.: *Integrating Packet FEC into Adaptive Voice Play-out Buffer Algorithms on the Internet,* Cornell University, Columbia University, USA, INFOCOM 2000.

[4] FUJIMOTO, K., ATA, S., MURATA, M.: *Adaptive Play-out Buffer Algorithm for Enhancing Perceived Quality of Streaming Applications,* Osaka University, Osaka City University.

[5] KLIMO, M.: *Buffer Management for CBR Traffic,* University of Žilina, Communications – Scientific Letters of the University of Žilina, ISSN 1335-4205.

Samuel Alexík *

NOVEL ADAPTIVE METHOD OF SETTING PARAMETERS FOR MARSIK CONTROL ALGORITHM

A novel method of Marsik algorithm improvement is described in the paper. The KappaZ parameter in Marsik algorithm representing a required rate of control error oscillations that influences the quality of control most can be changed adaptively to obtain a better process performance. In the original version of the algorithm a constant value of KappaZ = 0,5 is recommended by the author. Here, a concept how to improve control performance by adaptively changing KappaZ is described. An analysis of the parameter influence is made.

1.Introduction

An advantage of the algorithm is that controlled system identification and special signals introducting is not required. A control error only is calculated to define a new criterion type. The new criterion does nol lead to finding optimization function extremes as usual, so the adaptation goes as a common control on. The new criterion is an oscilation index that specifies the rate of control error and its first derivation cross the zero level [1][2]. The measure-controlled value must be properly filtrated from the high frequency noises that could influence the process of specifying oscilation coefficient negatively.

2. How to change KappaZ adaptively

After analysing the *KappaZ's* influence on the control process quality, some relationship between *KappaZ* and the control loop response time has been discovered. It was made possible to either slow down by its decreasing. There is still a problem how to detect whether a process is "too slow" or "too fast".

A block diagram of Marsik algorithm is shown in Fig. 2. The blue line is showing the feedback to change KappaZ adaptively.

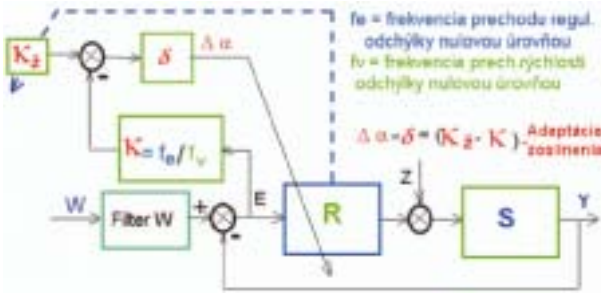


Fig. 1. Block diagram of improved Marsik control algorithm

Marsik algorithm also estimates the global time constant named Tau. Its value can be used to compute the *KappaZ's* changing time and direction. If the system output reaches the setpoint in time shorter than Tau it is supposed that process is "too fast" and there is a need to damp it by decreasing KappaZ (see Fig. 2 line 1) and vice versa, if the system output is significantly smaller than the setpoint at time *KappaZ* should be increased to speed the process up (see Fig. 2 line 2).

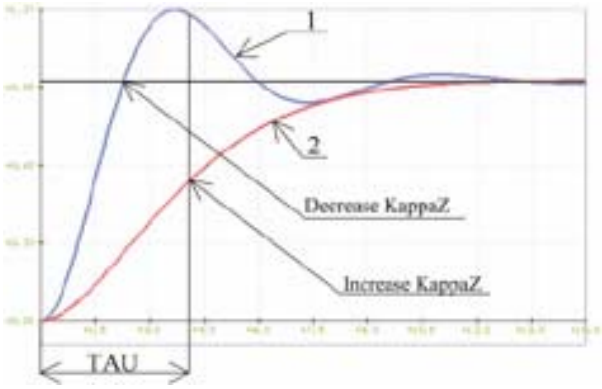


Fig. 2. Principle of changing KappaZ

3. Simulations

From the set of benchmark systems recommended by Aström [3], three system types were chosen: a system with multiple equal poles, a non-minimal phase system and dead-time system. In the next three figures, performances of Marsik algorithm with and without *KappaZ* adaptation are depicted and compared to analytically tuned PID. [4]. The green line represents a Pid controller, magenta representing Marsik with adaptation and the blue one Marsik without adaptation. The red line shows the changing of *KappaZ*. *KappaZ* is adaptated first after the setpoint change but

* **Samuel Alexík**
Žilinská univerzita, Faculty of Management Science and Informatics, Department of Cybernetics, 010 26 Žilina,
E-mail: samo@frtk.fri.utc.sk

the algorithm can be changed in the future to be able to adapt KappaZ even during disturbancies from the steady state.

In Fig. 3 the main difference between On and Off Adaptation is that there are less oscillations and higher stability in the case with adaptation. It is visible how *KappaZ* changes the process behaviour at the points of red line step changes.

In Fig. 4, decrease of *KappaZ* in time is to be seen. The reason is that the process is more oscillating and the algorithm is trying

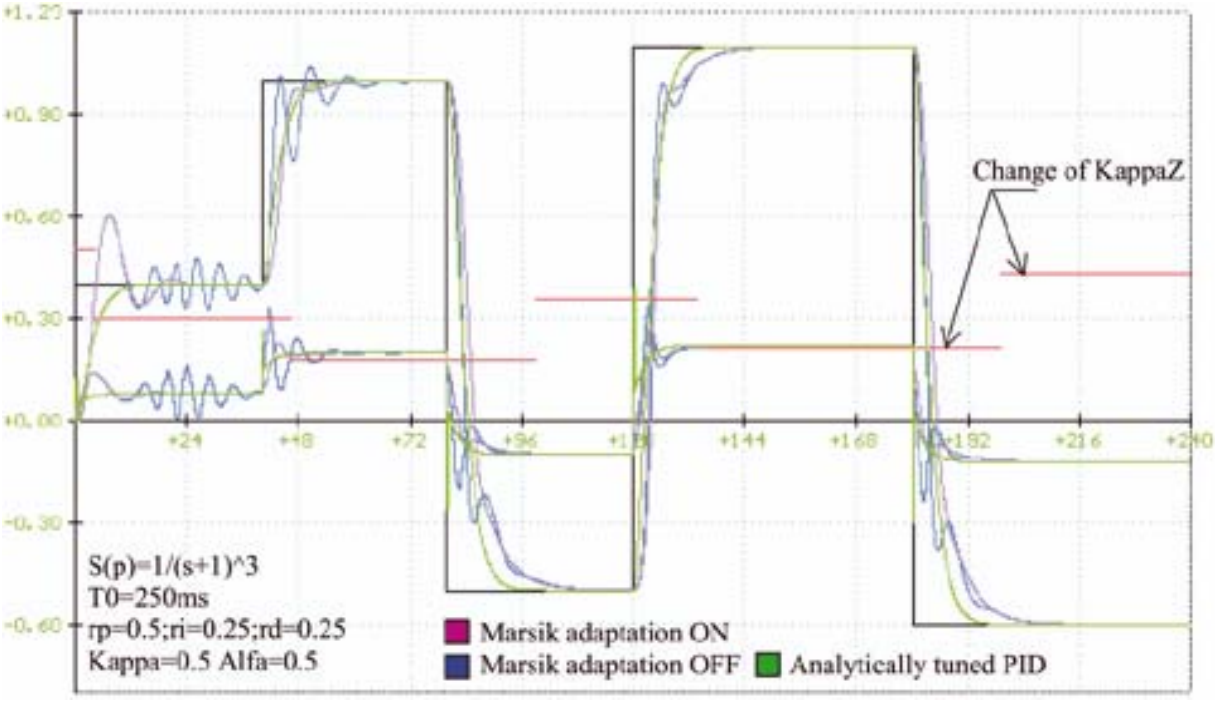


Fig. 3. Third order system

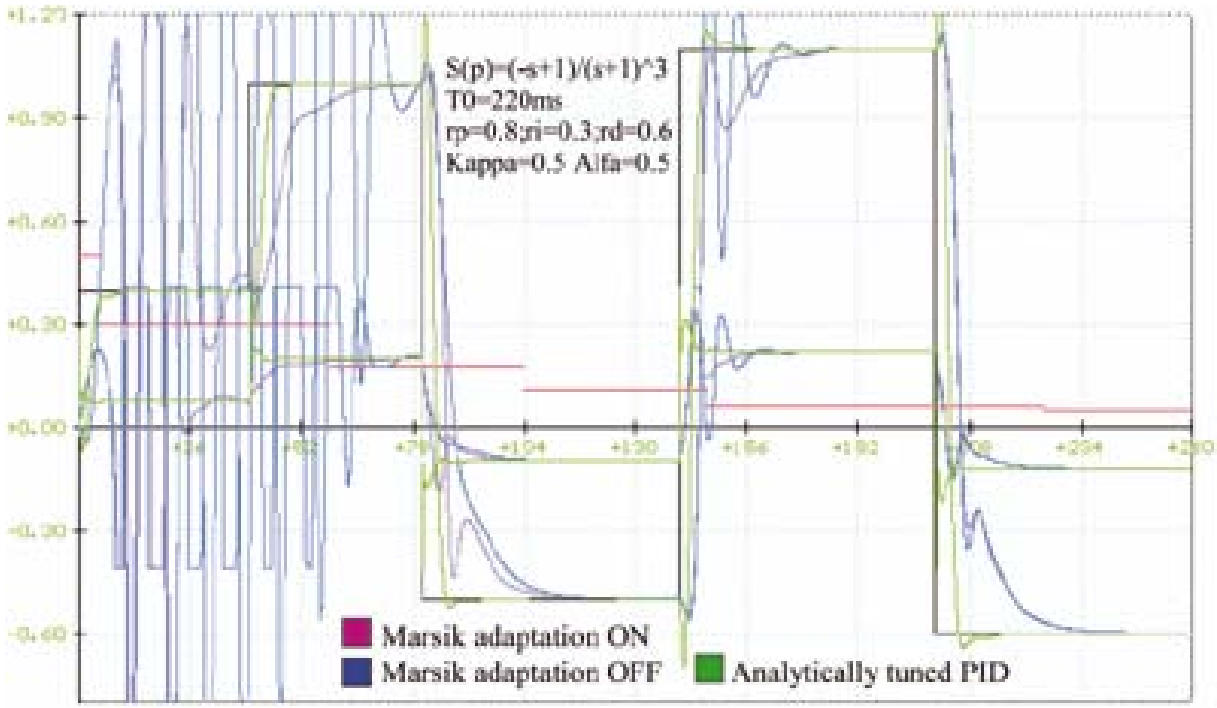


Fig. 4. Non-minimal Phase system

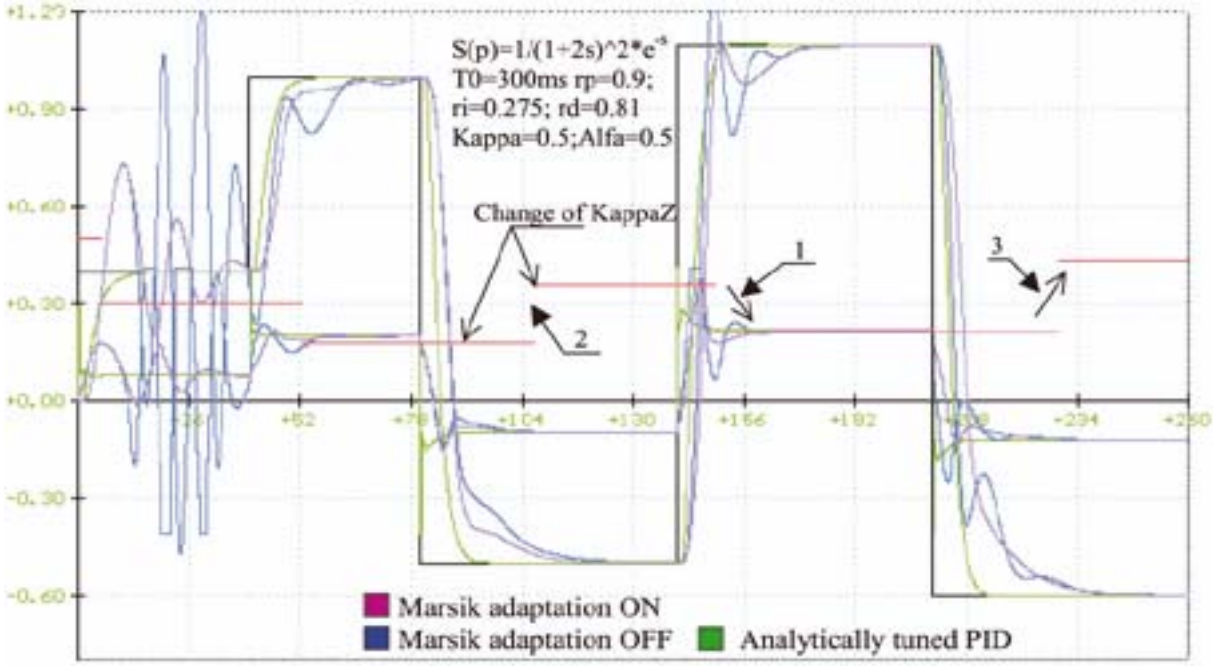


Fig. 5. System with Dead Time

to decrease it. Great oscillations at the chart beginning are caused by a huge gain compensated later as the adaptation goes on. Last, Fig. 5 shows that systems with non-minimal phase are harder to control by Marsik algorithm. Here, the influence of *KappaZ* to a controlled process can be seen clearly, esp. at points 1, 2, and 3. *KappaZ* changes the system dynamics, PID proves to perform best in the analytical PID control.

IAE criterion Table 1

	Fig. 1	Fig. 2	Fig. 3
PID	22.813	20.058	24.456
Marsik without adaptation	27.825	95.43	55.420
Marsik with adaptation	31.69	58.44	48.435

Table 1 shows that control process with adaptation has value of IAE smaller than process without adaptation in most cases. However smaller value of IAE criterion does not need to be always corespondent to better quality of control. For instance number of oscillations, their magnitude, number of overshoots and moreover, are also important.

4. Conclusion

It can be claimed that the KappaZ adaptation leads to a better performance at benchmarked systems but its adaptation itself has to be improved to adapt continually at every sample time period. The adaptation under disturbancies can be added as an improvement as well. Marsik algorithm is sensitive to great large changes of the sampling period. Algorithm tests on different systems and carrying out tests with disturbancies is planned and adding a filter for KappaZ adaptation as well. All the improvements may lead to a more robust algorithm insensitive to sampling period values.

Acknowledgment

This work has been supported by the institutional grant FRI 28/2003.

References

[1] PETERKA, V. a kol.: *Algorithms for adaptive microprocesor conroll of technological process,* Institute of Information Theory and Automation of the Academy of Science of the Czech Republic, 1982

[2] GÖRNER, V., MARŠIK, J.: *Across automation controll. Software newspaper 6/1990,* SWS Slušovice

[3] ÅSTRÖM, K. J., HÄGGLUND, T.: *Benchmark Systems for PID Control'* In IFAC Workshop on Digital Control – Past, present, and future of PID Control, Terrassa, Spain, 2000, pp. 181-182

[4] ALEXÍK, M.: *Adaptive Self-Tuning PID Algorithm Based on Continuous Synthesis.* A Proceeding from the 2nd IFAC Workshop, *New Trends in Design of Control Systems.* Smolenice, Slovak Republic, 7-10 September 1997. pp.481-486. Published for IFAC by Pergamon an Imprint of Elsevier Science, 1998

Juraj Smieško *

EXPONENTIAL MODEL OF TOKEN BUCKET SYSTEM

In the presented paper we use the Theory of Markovov Chains for simulation of a simple model of the Token Bucket System (TBS). In the ***first section*** *we deal with a steady-state analysis of the TBS with Exponential model of On-Off Source of human speech. We develop recurrent formulas for state probabilities of the TBS. In the* ***second section*** *we use real parameters of VoIP and we calculate characteristic of the TBS. In the* ***last section*** *the values of probability of a loosing packet are approximated by more types of regression functions. It is shown that the cubic approximation is the most efficient. Using this function we can directly compute values of probability of a loosing packet.*

1. Steady-state Analysis

The Token Bucket System is related to VoIP problems. We have formed the steady-state analysis of VoIP under the Token Bucket Control. Our main problems will be to compute the probability characteristics of TBS and to find relation between probability of a loosing packet and the bucket depth. At first we will analyze the work of TBS in general (Fig. 1).

ON: $T_1 \sim f_1(t) = \alpha e^{-\alpha t}$

$$ET_1 = \frac{1}{\alpha} = 227\ ms \quad \alpha = 0.00441 ms^{-1}$$

OFF: $T_2 \sim f_2(t) = \mu e^{-\mu t}$

$$ET_1 = \frac{1}{\mu} = 596\ ms \quad \alpha = 0.00168 ms^{-1}$$

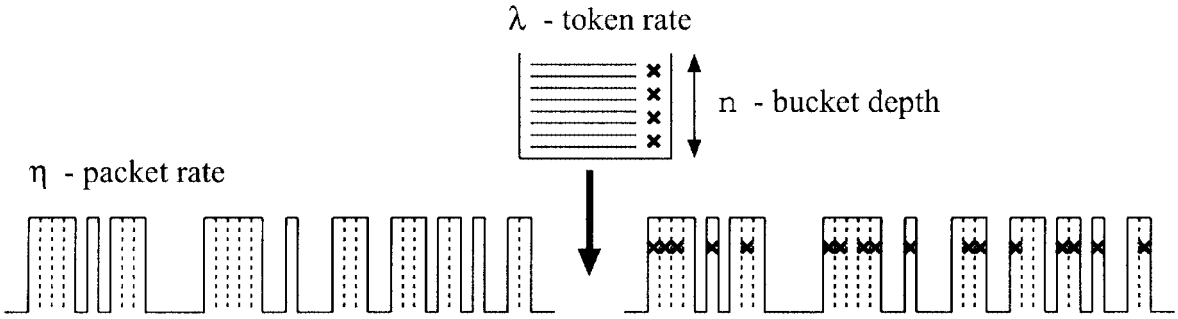


Fig. 1

There is a flow of packets entering the TBS. The Token Bucket System is generating tokens (marks) and then it marks each packet with one of them. Only marked packets will go through the network. For practical reasons we can assume a limited bucket with depth "n". In case the bucket is empty (there are no tokens), TBS cannot mark any packet, which is lost then. Let P_{lst} be probabilities of this random phenomenon and we will call it "probability of loosing packet".

We will deal with simple Exponential model of On-Off Source of human speech. This Source has two states. The state On is period of "speech" and the Off state is period of "silence". Between these states, the source switches randomly. Talk-spurt duration is modelled by variant T_1 and pause duration is modelled by variant T_2. The values of distribution parameters were gained from [1].

When the Source is in On state it starts generating a flow of packets, which represents human speech. The packet rate in usual VoIP systems (using G.729A) is 50 p/s. We will assume that the flow of packets is modelled by Poisson process $N_P(t)$ with rate $\eta = 0.05$p/ms .

In general there can be any token rate. Usually it is the same as an average number of packets entering the TBS. Let P_{ON} and P_{OFF} be probabilities that the Source is in state On or Off. The flow of tokens will be modelled by Poisson process $N_T(t)$ with rate $\lambda = \eta \,.\, P_{ON} + 0 \,.\, P_{OFF}$.

We will construct a transition diagram of Markov model of On-Off Source (Fig. 2):

* **Juraj Smieško**
Department of InfoCom Networks, Faculty of Management Science and Informatics, University of Žilina

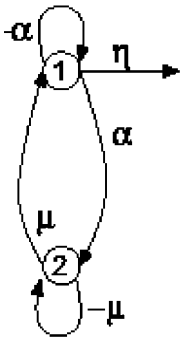


Fig. 2

Let Q_0 be a rate matrix for this chain. We compute state probabilities π of a steady-state chain by balance equations:

$$Q_0 = \begin{pmatrix} -\alpha & \alpha \\ \mu & -\mu \end{pmatrix} \quad \pi = (\pi_1, \pi_2),$$

$$\pi \cdot Q_0 = 0 \;\Rightarrow\; \pi = \left(\frac{\mu}{\alpha+\mu}, \frac{\alpha}{\alpha+\mu}\right)$$

We compute probabilities and token rate for real parameters of VoIP:

$\pi = (0.27586, 0.72414)$

$\mu = 0.05p/ms$. $P_{ON} = 0.05p/ms$. $\pi_1 = 0.01379\ p/ms$

In the Token Bucket System we have three elementary random phenomena, entering the packet, generating token and switching between On-Off states in Source. We have assumed that each of them is Poisson process, and because of that we can model this TBS by Markov chain.

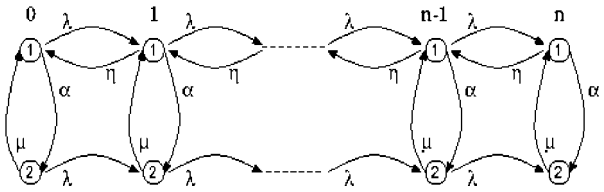


Fig. 3

Probability of an increasing number of tokens in the bucket about one during "short" time Δt is

$$P(N_T(\Delta t) = 1) = \lambda\Delta t \cdot e^{-\lambda\Delta t} = \lambda\Delta t + o(\Delta t)$$

If the bucket contains k tokens, probability of a decreasing number of tokens in the bucket about one during "short" time Δt is

$$P(N_P(\Delta t) = 1) = \eta\Delta t \cdot e^{-\eta\Delta t} = \eta\Delta t + o(\Delta t)$$

If there is an arriving packet and the bucket is empty, we cannot mark it and this packet is lost. Probability of loosing packet was named P_{lst}.

We will construct a transition diagram of the whole system (Fig. 3). Component columns of diagram refers to a number of tokens in the token bucket. We have assigned this as "k-level" for $k = 0, ..., n$. For easy reading loops in the transition diagram are omitted:

Let $X_k(Y_k)$ be a probability that in the TBS are k tokens and Source is in state On (Off). Let P_k be a probability of k-level. We see that $P_k = X_k + Y_k$ and $P_{lst} = X_0$. Taking from the Theory of Markov Chains [2] we can write the following equations for state probabilities of the steady-state chain:

$$\begin{aligned}
\text{for } k = 1 \quad & 0 = -(\lambda+\alpha)X_0 + \mu Y_0 + \eta X_1 \\
& 0 = \alpha X_0 - (\lambda+\alpha)Y_0 \\
& \vdots \\
\text{for } k = 2, ..., n-1 \quad & 0 = \lambda X_{k-1} - (\lambda+\alpha+\eta)X_k + \mu Y_k + \eta X_{k+1} \\
& 0 = \lambda Y_{k-1}\, \alpha X_k + \mu Y_k - (\lambda+\mu)Y_k \\
& \vdots \\
\text{for } k = n \quad & 0 = \lambda X_{n-1} - (\alpha+\eta)X_n + \mu Y_n \\
& 0 = \lambda X_{n-1} + \alpha X_n - \mu Y_n
\end{aligned}$$

We will solve the steady-state equations $Q'p = 0$, where $p = (Y_n, X_n, ..., Y_k, X_k, ... Y_0, X_0)$ and matrix Q' is formed as:

$$Q' = \begin{pmatrix}
-\mu & \alpha & \lambda & 0 & ... & 0 & 0 & 0 & 0 \\
\mu & -(\alpha+\eta) & 0 & \lambda & ... & 0 & 0 & 0 & 0 \\
0 & 0 & -(\lambda+\mu) & \alpha & ... & 0 & 0 & 0 & 0 \\
0 & \eta & \mu & -(\lambda+\alpha+\eta) & ... & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & ... & -(\lambda+\mu) & \alpha & \lambda & 0 \\
0 & 0 & 0 & 0 & ... & \mu & -(\lambda+\alpha+\eta) & 0 & \lambda \\
0 & 0 & 0 & 0 & ... & 0 & 0 & -(\lambda+\mu) & \alpha \\
0 & 0 & 0 & 0 & ... & 0 & \eta & \mu & -(\lambda+\alpha)
\end{pmatrix}$$

We modified the rate matrix Q' to the equivalent triangular matrix and we used a normalization condition for state probabilities

$$\sum_{k=0}^{n}(X_k + Y_k) = 1$$

$$\begin{pmatrix}
-\mu & \alpha & \lambda & 0 & ... & 0 & 0 & 0 & 0 \\
0 & -\eta & \lambda & \lambda & ... & 0 & 0 & 0 & 0 \\
0 & 0 & -(\lambda+\mu) & \alpha & ... & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\eta & ... & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & ... & -(\lambda+\mu) & \alpha & \lambda & 0 \\
0 & 0 & 0 & 0 & ... & 0 & -\eta & \lambda & \lambda \\
0 & 0 & 0 & 0 & ... & 0 & 0 & -(\lambda+\mu) & \alpha \\
1 & 1 & 1 & 1 & ... & 1 & 1 & 1 & 1
\end{pmatrix} \cdot \begin{pmatrix} Y_n \\ X_n \\ Y_{n-1} \\ X_{n-1} \\ \vdots \\ Y_1 \\ X_1 \\ Y_0 \\ X_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

We gained the recurrent formulas for state probabilities:

$$\boxed{Y_0 = \frac{\alpha}{\lambda+\mu} X_0}, \quad \boxed{X_k = \frac{\lambda}{\eta}[Y_{k-1} + X_{k-1}]} \quad \text{for } k = 1, ..., n$$

$$Y_k = \frac{\alpha}{\lambda+\mu} X_k + \frac{\lambda}{\lambda+\mu} Y_{k-1} \quad \text{for } k = 1, ..., n,$$

$$Y_n = \frac{\alpha}{\mu} X_n + \frac{\lambda}{\mu} Y_{n-1}$$

We can reduce formulas for Y_k and Y_n to:

$$\boxed{X_k = \frac{\lambda}{\lambda+\mu}\left[\frac{\alpha+\eta}{\eta} Y_{k-1} + \frac{\alpha}{\eta} X_{k-1}\right]} \quad \text{for } k = 1, ... n,$$

$$\boxed{Y_n = \frac{\lambda}{\mu}\left[\frac{\alpha+\eta}{\eta} Y_{n-1} + \frac{\alpha}{\eta} X_{n-1}\right]}$$

2. TBS with real parameters of VoIP

To deduce explicit formula for probability of loosing packet $P_{lst} = P_{lst}(n) = X_0$ is difficult and impracticle, but there is no problem to compute the values of P_{lst} for real parameters of our TBS with Exponential On-Off Source:

$\alpha = 0.00441\ m.s^{-1}$ $\mu = 0.00168\ m.s^{-1}$

$\lambda = 0.01379\ p/ms$ $\eta = 0.05 p/ms$

Than recurrent formulas for state probabilities are formed as:

$Y_0 = 0.28507 \cdot X_0$ $X_k = 0.27580 \cdot [Y_{k-1} + X_{k-1}]$

for $k = 1, ..., n$,

$Y_k = 0.97002 \cdot Y_{k-1} + 0.07862 \cdot X_{k-1}$ for $k = 1, ..., n$,

$Y_n = 8.93231 \cdot Y_{n-1} + 0.72397 \cdot X_{n-1}$

For example, the reader can see the difference between $n = 4$ and $n = 5$:

n	X_k	Y_k	P_k
0	0.14880	0.04242	0.19122
1	0.05274	0.05285	0.10559
2	0.02912	0.05541	0.08453
3	0.02331	0.05604	0.07935
4	0.02188	0.51742	0.53931

n	X_k	Y_k	P_k
0	0.13803	0.03935	0.17738
1	0.04892	0.04902	0.09794
2	0.02701	0.05140	0.07841
3	0.02136	0.05198	0.07361
4	0.02030	0.05212	0.07242
5	0.01997	0.48027	0.50024

For real use its enough to have the bucket depth $n = 1, ..., 10$. Now we will increase the bucket depth n and calculate characteristics of models:

$P_{lst}(n)$ – probability of loosing packet or probability of empty bucket in time of arriving packet

$\lambda \,.\, P_{lst}(n)$ – average number of lost packets

P_n – probability of full bucket (bucket will refuse tokens)

EK – average bucket depth $\aleph = \frac{EK}{n}$ – token bucket usage

n	$P_{lst}(n)$	$\lambda \cdot P_{lst}(n)$	P_n	EK	$\aleph$
1	0.20367	2.809 p/s	0.73826	0.73826	73.8%
2	0.17796	2.454 p/s	0.64503	1.41634	70.8%
3	0.16163	2.229 p/s	0.58580	2.05572	68.5%
4	0.14880	2.052 p/s	0.53931	2.66994	66.7%
5	0.13803	1.903 p/s	0.50024	3.26649	65.3%
6	0.12874	1.775 p/s	0.46658	3.84967	64.2%
7	0.12064	1.664 p/s	0.43719	4.42225	63.2%
8	0.11349	1.565 p/s	0.41129	4.98619	62.3%
9	0.10715	1.478 p/s	0.38828	5.54295	61.6%
10	0.10148	1.399 p/s	0.36771	6.09369	60.9%

3. Relation between Probability of loosing packet and the Token Bucket depth

The most interesting characteristic is probability of loosing packet $P_{lst}(n)$. Its values will be approximated by regression functions $Y_I(n)$:

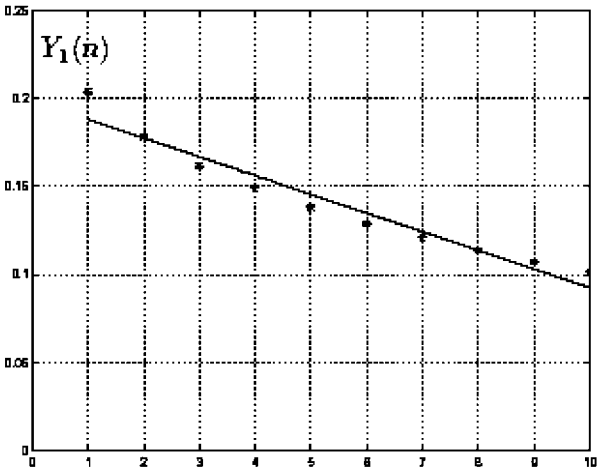
$Y_1(n) = an + b$ $Y_2(n) = an^2 + bn + c$

$Y_3(n) = an^3 + bn^2 + cn + d$ $Y_e(n) = ae^{bn}$

by the least squares method. We will measure the quality of approximation by square of sum residuals:

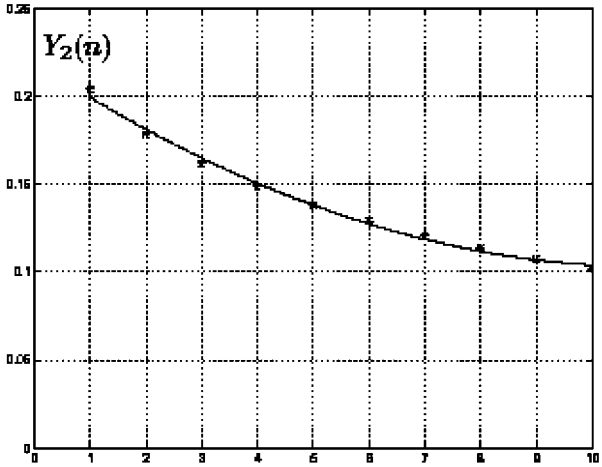
$$\mathfrak{E}_I = \sqrt{\sum_{n=0}^{10} [P_{lst}(n) - Y_I(n)]^2} \qquad I = 1, 2, 3, e$$

Approximation by the linear function: $Y_1(n) = -0.01060 \,.\, n + 0.19849$ with $\mathfrak{E}_1 = 0.02307$



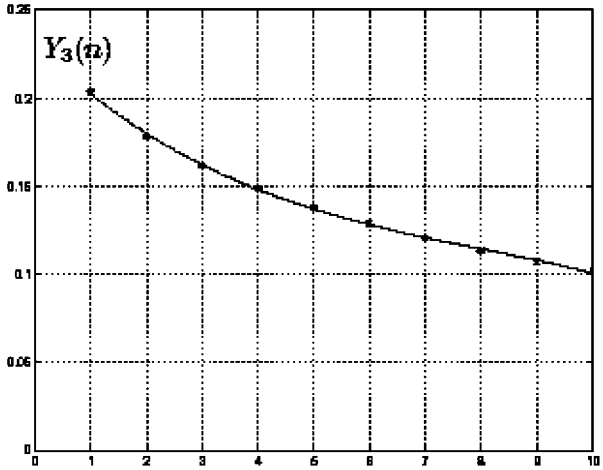
n	$Y_1(n)$	$P_{lst}(n) - Y_1(n)$
1	0.18788	0.01579
2	0.17728	0.00068
3	0.16667	0.00505
4	0.15607	0.00726
5	0.14546	0.00743
6	0.13486	0.00611
7	0.12425	0.00361
8	0.11365	0.00015
9	0.10304	0.00411
10	0.09244	0.00904

Quadratic regression: $Y_2(n) = 0.00095 \,.\, n^2 - 0.02105 \,.\, n + 0.21937$ with $\mathfrak{E}_2 = 0.00753$



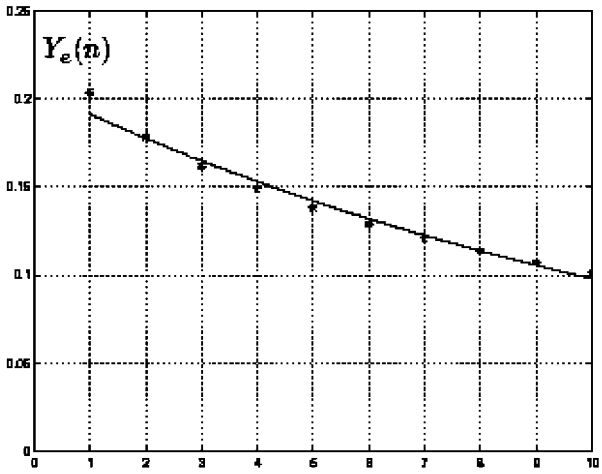
n	$Y_2(n)$	$P_{lst}(n) - Y_2(n)$
1	0.19927	0.00440
2	0.18108	-0.00311
3	0.16477	-0.00315
4	0.15037	-0.00157
5	0.13787	0.00016
6	0.12726	0.00148
7	0.11856	0.00208
8	0.11175	0.00175
9	0.10684	0.00031
10	0.10383	-0.00235

Cubic regression: $Y_3(n) = -0.00012 \,.\, n^3 + 0.00297 \,.\, n^2 - 0.03039 \,.\, n + 0.22990$ with $\mathfrak{E}_3 = 0.00318$



n	$Y_3(n)$	$P_{lst}(n) - Y_3(n)$
1	0.20237	0.00131
2	0.18004	-0.00208
3	0.16220	-0.00057
4	0.14809	0.00071
5	0.13697	0.00104
6.	0.12815	0.00060
7.	0.12084	-0.00020
8	0.11432.	-0.00083
9	0.10787	-0.00072
10	0.10073	0.00074

Approximation by the exponential function: $Y_e(n) = 0.20615 \,.\, e^{-0.07447}$ with $\mathcal{E}_e = 0.01501$



n	$Y_e(n)$	$P_{lst}(n)$ - $Y_e(n)$
1	0.19135	0.01232
2	0.17762	0.00034
3	0.16487	-0.00325
4	0.15304	-0.00424
5	0.14206	-0.00403
6	0.13186	-0.00312
7	0.12240	-0.00176
8	0.11361	0.00012
9	0.10546	0.00169
10	0.09789	0.00359

We can see the cubic regression function is a very "good" approximation with maximum error $2{,}1 \cdot 10^{-3}$. If we are satisfied with this precision, we can exchange $P_{lst}(n)$ for $Y_3(n)$:

$$P_{lst}(n) \doteq -0.00012 \cdot n^3 + 0.00297 \cdot n^2 - 0.03039 \cdot n + 0.22990$$

Because of its practical use the exponential approximation is "better" (with maximum error $1{,}2 \cdot 10^{-2}$).

References

[1] ITV-T Recommendation P.59: *Articial Conversational Speech,* 1993
[2] PEŠKO & SMIEŠKO*: Stochastic Models of Operation Research (Stochastické Modely Operačnej Analýzy)*, Žilinská univerzita, 1999.

Miroslav Plevný *

A NOTE ON "MORE FOR LESS" PARADOX IN RELATION TO ECONOMIC PROBLEMS

The More for Less Paradox is not only an interesting theoretic construction. An optimization problem, which has this interesting property, can also be found in the real economic world. This contribution was stimulated by the paper [4], which results from the contribution [1] and [2] discussing one interesting property of the linear programming models behaviour, and comments on it and discusses it further. In the paper we attempt to show on a concrete case of a real practical problem [5] that similar property does not need to be a purely theoretical plaything, and it can also be spotted when analysing different variants of a real problem solution.

1. Introduction

The contribution discusses one interesting property of some linear programming problems known in literature as "More for Less Paradox". In the opening chapters the substance of this paradox and some further connection related to this phenomenon are described briefly. The following parts of the contribution demonstrate that it is also possible to find some characteristics analogous to "More for Less" paradox in individual practical problems. The purpose is to point out that even rather extensive practical problems can show an analogue with simple theoretical model properties.

2. Substance of the "More for Less Paradox"

Suppose the firm uses n different manufacturing processes T_1, T_2, ... T_n (it is technologies, production programs, cutting schedules, etc.) to produce the required amounts of m different products. Each manufacturing process j is described by the output coefficient a_{ij}, indicating the amount of product i by using just one unit of the manufacturing process T_j. Further the required output numbers b_i of product i, and costs c_j per unit of the manufacturing process T_j are known. The decision-making problem for a firm is, what manufacturing processes to apply and in what frequency in order to produce the required amount of products at minimal costs.

This decision-making problem can be formulated as following linear model, where x_j means the number of units of the manufacturing process T_j used:

$$\text{minimize} \quad z(\boldsymbol{x}) = \sum_{j=1}^{n} c_j x_j \tag{1}$$

$$\text{subject to} \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, 2, \ldots m \tag{2}$$

$$x_j \geq 0\,, \quad j = 1, 2, \ldots n \tag{3}$$

The above paradox consists in the fact that for a number of the concrete queries for the some values of the matrix $\boldsymbol{A} = \{a_{ij}\}$ and the cost vector $\boldsymbol{c} = \{c_j\}$ is possible to find (depending on the values of the two different required output vectors $\boldsymbol{b}^1 = \{b_i^1\}$ and $\boldsymbol{b}^2 = \{b_i^2\}$) such two optimal solutions $\boldsymbol{x}_{opt}^1$ and $\boldsymbol{x}_{opt}^2$ of the problem (1) – (3), and so the following applies:

$$b_i^1 \geq b_i^2 \qquad \text{for all} \quad i = 1, 2, \ldots m,$$

and simultaneously

$$z(\boldsymbol{x}_{opt}^1) < z(\boldsymbol{x}_{opt}^2).$$

In other words, it is possible to achieve higher (it is "MORE") or even level of production for all products at lower total cost ("FOR LESS").

We can outline the above-mentioned situation using the specific example presented in [4]:

Suppose model (1) – (3) with $m = 2$ and $n = 6$, where the output coefficient matrix $\boldsymbol{A}$ is given:

$$\boldsymbol{A} = \begin{pmatrix} 3 & 2 & 1 & 2 & 4 \\ 1 & 1 & 2 & 2 & 5 \end{pmatrix}.$$

The required output numbers of the products are given by the vector $\boldsymbol{b}^1 = (15, 5)$, and the expenses per unit of the manufacturing process T_j are given by the vector $\boldsymbol{c} = (5, 3, 3, 4, 6)$. Then the optimal solution of the problem (1) – (3) leads to $\boldsymbol{x}_{opt}^{\boldsymbol{1}} = (5, 0, 0, 0, 0)$ with the objective function value $z(\boldsymbol{x}_{opt}^1) = 25$. However if the required output numbers of the products increase from $\boldsymbol{b}^1 = (15, 5)$ to the values $\boldsymbol{b}^2 = (16, 20)$, the optimal production programme of the firm will be the application of the visibly most expensive manufacturing process T_6 at solution $\boldsymbol{x}_{opt}^2 = (0, 0, 0, 0, 4)$ whereas total cost value decreases form 25 to the value $z(\boldsymbol{x}_{opt}^2) = 24$. This means that with the production increase of both the first product by 6,7% and the second product by 300%, the total cost will decrease by 4%.

* **Miroslav Plevný**
Department of Statistics and Operations Research, Faculty of Economics, University of West Bohemia in Pilsen, Hradební 22, 350 11 Cheb, Czech Republic; E-mail: miroslav.plevny@fek.zcu.cz

3. Some further comments to the solution showing the above paradox

The explanation of the paradox occurrence described in the previous chapter is trivial. The principal point is that we have to choose the expensive manufacturing processes at the expense of keeping the required combination of output products number. Such combinations of the output product number may exist that seen from the production cost point of view they are cheaper even with a higher number of manufactured products (compare required outputs $\boldsymbol{b}^1$ and $\boldsymbol{b}^2$ with the characteristics of the particular manufacturing processes $T_1, T_2, \ldots T_6$).

Further on it is possible to state that it is possible to prevent an occurrence of the presented paradox in the model (1) – (3) simply if we substitute the constraints (2) by the constraints:

$$\sum_{j=1}^{n} a_{ij}\, x_j \geq b_i, \quad i = 1, 2, \ldots m. \tag{4}$$

The model:

minimize (1)
subject to (4), (3)

allows all the combinations of the output product numbers with an even or higher production level than the limit numerical values b_i for all products $i = 1, 2, \ldots m$. So the found optimal solution shows from the point of the objective function value the features of the global minimum for all these combinations of the output product numbers, and thus the occurrence of the mentioned paradox is not possible.

4. Connection with shadow prices

The fact that the problem (1) – (3) shows the "more for less" property is possible to identify simply by means of the shadow prices of this problem. It is obvious that if the given problem solution has to show the characteristics of the above paradox, the increase of at least one right-hand-side value b_k of the constraints (2) has to be possible with simultaneous non-decreasing of other values b_i, $(i = 1, 2, \ldots m; i \neq k)$ so that the objective function value (1) decreases at the same time.

From the point of view of the definition of the shadow price value (see e.g. [3, pgs. 74-77]) for the above k^{th} constraint of the model (1) – (3) it means that this shadow price is negative.

It is obvious at the same time that in case of the model (1), (4), (3) such an instance cannot arise. This statement results from the feature of this model (the minimization problem, all constraints are of the type „ ≥ ", and all variables are non-negative), for which all the optimal solution variable values of the corresponding dual problem – it is shadow prices for primary problem – necessarily have to be non-negative (if there is any optimal solution at all).

5. Characteristics of the analyzed concrete example

The property of the optimization problem solution presented in the previous chapters need not be a purely theoretical plaything. Analysing the different solution variations of the chosen concrete practical problems, we can notice some analogical features in the interrelationship of some solutions of the given problem.

We will try to demonstrate it on a concrete problem, which we recently solved by means of linear programming [5]. It was a case of a wood-processing company consisting of two main workplaces: a sawmill and a shop. Besides the finished lumber for direct sale the sawmill supplies wooden lamellae to be processed in the shop. It is possible to buy further lamellae from other subjects if necessary for the manufacturing in the shop. In principle, both workplaces can operate independently from each other in the form of one to three-shift modes. The objective of the study was to pass a judgment on the company production schedule in terms of general efficiency at different alternatives of shift working.

The constructed linear model is described in [5] in detail. Here we present just the resulting objective function values (= the total profit) for the optimal solution of the constructed model, which were gained based on the concrete shift-working mode (see Table

Optimal solution of given problem depending on shift working — Table 1

Combina-tion No.	Number of shift		Profit	Combina-tion No.	Number of shift		Profit
	Saw-mill	Shop	[CZK]		Saw-mill		Saw-mill
1	0	0	-200 000,-	9	1	2	+916 521,-
2	1	0	-311 320,-	10	1	3	+954 921,-
3	2	0	-333 433,-	11	2	1	+1 217 559,-
4	3	0	-467 745,-	12	2	2	+1 380 085,-
5	0	1	-241 000,-	13	2	3	+1 421 085,-
6	0	2	-99 600,-	14	3	1	+1 461 027,-
7	0	3	-269 700,-	15	3	2	+1 645 169,-
8	1	1	+765 521,-	16	3	3	+1 686 169,-

1 – the results for in total 16 variations of shift-working in both workplaces taken into account).

6. Economic analysis of the resulting solution and connection with the above paradox

As the introduced concrete solved problem is a maximization model, we can by means of analogy to the presented model (1) – (3) consider the paradox occurrence, if higher aggregate gains are achieved at lower shift number. It is evident in the Table 1 that this situation occurs for all cases of the isolated saw-mill operation, when with the shift number increasing the gain goes down, i.e. the loss increases (see combination Nos. 2 – 4). Similar effect can be seen for the isolated hall operation (except the two-shift operation - combination No. 6) when the three-shift operation gain is worse than the gain from the one- or two-shift operation (see combination Nos. 5 - 7).

According to the obtained solution it is possible to achieve the highest total profit at 16^{th} shift working combination, i.e. at three-shift operation in both workplaces.

It could be misleading and even shortsighted to close the solution of the above problem by this rigorous statement of facts. It is well known that the application of the three-shift operation in companies, which cannot afford to have a reserve production facility, and usually do not operate in parallel running production lines, can be highly hazardous. It is impossible to compensate for an accidental production line breakdown, which can threaten the whole production process including holdbacks of other connected production lines. Using the three-shift operation we usually have a limited time left for prevention and diagnostics, whereby the breakdown risk increases. A possible operation failure cannot be caught up with a short-term productions increase by using additional shift. This is the case of the shop operation for the analyzed company in this study, where the high cost equipment is meant to glue the prisms.

Just in the case of the three-shift operation a realistic estimation of the anticipated profit for the company is necessary as well as considering a number of the above-mentioned hard to quantifiable limitations also, which wasn't included in the solved mathematical model.

Comparing the combination No. 13 and 14 it becomes evident that the use of one additional sawmill shift compensates for the two-shift decrease in the shop (see combination No. 10 and 11). This fact is very important in the context of the above-discussed problem of the three-shift shop operation, which is highly hazardous in case of any production disturbances. In case of disturbances and subsequent shop shutdown we come instantly to loss-making production schedules No. 1 – 4.

On the basis of the previous point it is possible to state that despite the found optimal solution for combination No. 16, the appropriate recommendation for the company operation can be the shift combination No. 15 (it is sawmill – 3 shift, shop – 2 shift). The total production profit calculated by the mathematical model can be lower only by 2.5%.

If it is necessary to reduce production (distribution difficulties, lack of input etc.) the shift combination No. 14 seems relatively profitable, where the profit is lower of about less than 13.5% at production volume decrease (in m^3) of 25.5%.

These findings did not directly reflect the "more for less paradox" properties discussed in this paper already, nevertheless some analogies can certainly be found here. We tried to point out a possible connection between the solution of rather extensive existing problems in economic reality, and some properties of the simple theoretical models, which seem to be artificial and impractical at the first glance.

References

[1] CHARNES, A., DUFFUAA, S. and RYAN, M.: *The more for less paradox in linear programming.* European Journal of Operational Research 31 (1987), pp. 194-197. ISSN 0377-2217.

[2] CHOBOT, M., TURNOVEC, F.: *One economic paradox in linear programming models.* Ekonomicko-matematický obzor 4, 1974, pp. 374-386. ISSN 0013-3027 (in Slovak language).

[3] JABLONSKÝ, J.: *Operations Research - Quantitative Models for Economics Decision Making (Operační výzkum - Kvantitativní modely pro ekonomické rozhodování).* Praha: Professional Publishing, 2002, p. 323. ISBN 80-86419-23-1 (in Czech language).

[4] LUPTACIK, M.: *A Note on the "More-for-Less" Paradox.* In: KISCHKA, P. ... (ed.) *Models, Methods and Decision Support for Management.* Heidelberg, New York: Physica-Verlag, 2001, pp. 103-109. ISBN 3-7908-1373-7.

[5] PLEVNÝ, M.: *Small Decision Problems in Logistics-Management and their Effective Solution.* In: *Mathematical Methods in Economics 2002 – Proceedings of the 20th International Conference,* Sept. 3-5, 2002, Ostrava, pp. 215-222. ISBN 80-248-0153-1.

komunikácie
COMMUNICATIONS

Michal Žarnay *

DEADLOCK SOLVING IN TRANSPORT SYSTEM WITH METHODS FROM COMPUTER OPERATING SYSTEM

In its first part, the article compares control principles in computer operating systems and in transport systems, and outlines similarities. One part of the control in both systems is handling of deadlock situations. When replacing human control by computer control in transport systems, handling of deadlock situations must be tackled as well. In its second part, the article outlines algorithms used in the operating systems field for that and discusses their potential application in the transport systems field. It comes to a conclusion that avoidance of deadlock by dynamic analysis of requested and assigned resources to processes looks to be the most perspective way generally, although in specific systems, rules for prevention or detection and recovery from deadlock can be also applied.

1. Motivation

As informatics and automatic control are being implemented in the area of transport, the control part is included as well. There are attempts to replace human control by computer control.

In some parts of transport systems control, computers successfully replaced human operators. It relates especially to control of automatic operations that were defined as algorithms and that help now to produce control of higher quality.

However, in a complex transport system with complex control tasks that require complex decision-making, the human supervision is necessary. Even if computers do partial operations, some situations usually occur that cannot be solved by the computer support. This problem could be transformed into the problem of modelling of human decision-making with help of the computer.

When looking for algorithms to make models of human behaviour in transport system control, we may use several approaches. In this article, I would like to compare control in computer operating systems and in transport systems.

One part of control functions in both systems represents handling of deadlock situations. In operating systems, it is usually the machine that solves the situations automatically, without human interference. In complex transport systems it is usually a human operator who handles these situations. In the effort of replacing (at least in part) the human by a computer control, handling of deadlock situations must be created. That is why another goal of this article is to discuss application of algorithms for solving deadlock situations from the operating systems field in the transport systems field.

2. Control in Operating Systems

A goal of an operating system (OS) is to carry out a given set of jobs with use of a given set of resources in a given time period according to a prescribed algorithm. Both sets may change their contents in course of time.

When a job arrives to an OS, it is inserted into a queue of jobs waiting for processing. When all resources needed for a start of job processing are available, the job is removed from the waiting queue and it gets ready to be processed. The job processing can consist of one or more process threads. The process or its threads individually get assigned resources that they need. This can be at the beginning or in course of process – according to a prescribed algorithm. When the process or its thread is finished, the assigned resources that have not yet been released, are given back to the OS. After all the process threads related to a job are done, the job is removed from the system.

As an example, the resources in an OS can be a processor, internal and external memory (RAM, hard disk, floppy disk, CD-ROM), input and output devices for communication with the user (keyboard, mouse, display, printer) or with other computers (network equipment).

Among jobs, as an example, we can imagine calculation of a mathematical problem, printing of a text document, editing of a table in a table processor, scanning of a picture, copying of data from a hard disk to a floppy disk and receiving of data from the computer network.

All of these jobs process data according to a prescribed algorithm. The term data in the given examples, as it is evident, covers numerical data, a text document, tables, pictures, etc. In some processes, the data are transformed from input to output, in other cases, they are moved between two places, or both, moved and transformed.

3. Analogy in Control of Transportation System and Operating System

As a transportation system (TS) we can consider a network or a part of it on macroscopic or microscopic level, serving one or

* **Michal Žarnay**
Department of Transportation Networks, Faculty of Management Science and Informatics, University of Žilina, Moyzesova 20, 010 26 Žilina, Slovak Republic, Tel: +421-41-5134224, E-mail: Michal.Zarnay@fri.utc.sk

more transportation modes – for instance a passenger train station, a road crossing, air traffic network in a country, inland waterway network in a region or a terminal of combined transport.

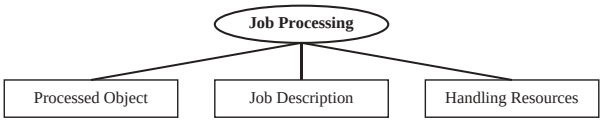


Fig. 1. Process in Transport System

If we take the passenger train station, "processing" of a job on a train in the station can represent a process (Fig. 1). The train enters the station, arrives to a platform, people get off and on the train, the train crew and train locomotives may exchange and the train leaves out. The handling resources in this case include infrastructure of the station, namely the tracks and the platform used, and station crew involved in the "processing" of the train. For other jobs connected with a change of train cars composition, required resources may include other types of tracks (e.g. a hump, sorting siding, depot) and workers as well as shunting locomotives.

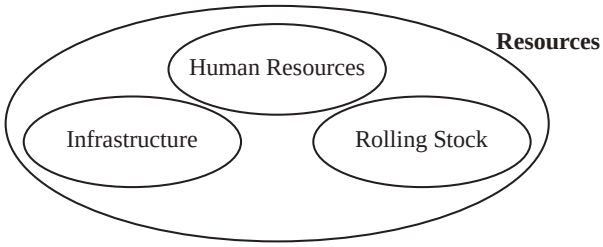


Fig. 2. Resources in Transport System

In other TS-s, it is similar. Jobs are represented by movement of vehicles, passengers or goods from one point to another in the system, with or without additional operations performed. This responds to the different kinds of processes including transformation or movement or both.

Resources can be in principle divided into technical equipment and human resources. Technical equipment can be further structured as infrastructure (routes and adjacent equipment) and rolling stock (moving vehicles). See Fig. 2.

The processing algorithms in TS are represented by job descriptions. They can be represented in different ways. One of them is a flow chart (Fig. 3).

4. Deadlock theory

Processes in OS as well as TS (the word "system" will be used further, since it may apply both OS and TS) can run in principle in two ways: successively or concurrently. When running concurrently, the system must take care of synchronization of their access to the resources.

If two or more processes use disjunctive sets of resources during the whole time period, when they run concurrently, or they use common resources with a non-exclusive assignment, there is no problem of synchronisation.

However, if they use at least one common resource that must be assigned exclusively, it may happen that one of the processes needs the common resource, while another process is using it. Thus the first process must wait for the second one to release the conflict resource. The need of the resource may arise by more processes and they get into a set of waiting processes for the conflict resource. After the resource is made available, it will be assigned to one of the waiting processes that can thus continue. After this process is finished, the conflict resource can satisfy next process from the waiting set, etc. until the set is empty.

If there are two processes P_1 and P_2 (Fig. 4) that use two or more common resources – let's assume 2 resources R_A and R_B for this case – and the resources are to be assigned in different order in the time period of concurrent running, the two processes can come to a situation called deadlock. It is a situation, when the P_1

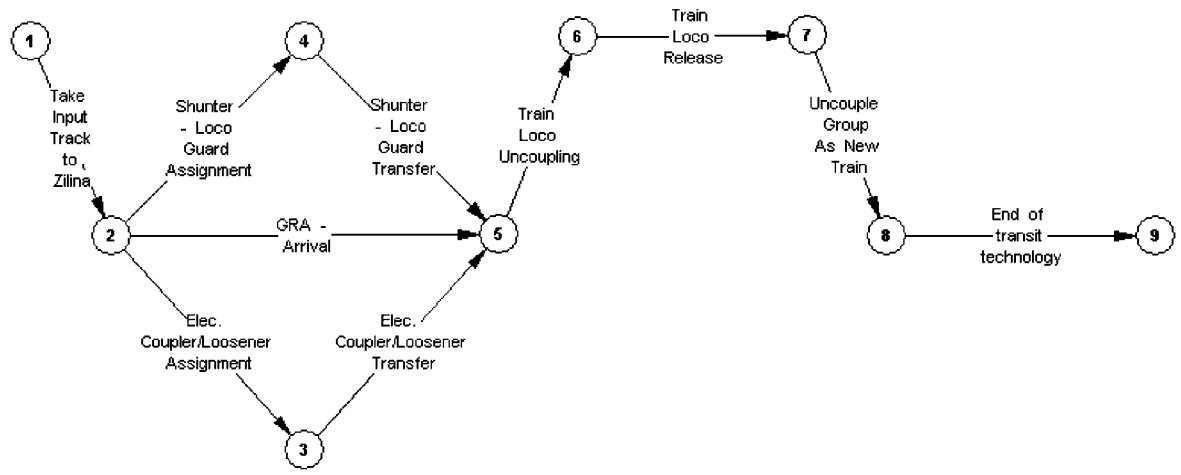


Fig. 3. Flow chart of technological job description

process has got assigned the R_A resource and needs the R_B resource, and the P_2 process has got assigned the R_B resource and waits for the R_A resource.

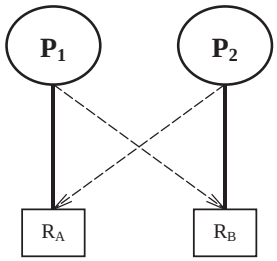


Fig. 4. Deadlock

There are four necessary Coffman conditions for a deadlock to occur:

- Mutual exclusion – there is more than one resource in the system that is either currently assigned to exactly one process or is available,
- Hold and wait – a process currently holding resources granted earlier can request other resources,
- No pre-emption – resources previously granted cannot be taken away from a process, but they must be explicitly released by the process holding them,
- Circular wait – there must be a circular chain of two or more processes, each of which is waiting for a resource held by next member of the chain.

All four of these conditions must be fulfilled for a deadlock to occur. If one of them is not fulfilled, no deadlock is possible.

In general, the following strategies are used for dealing with deadlocks:

- Detection and recovery – let deadlocks occur, detect them, and take a remedy action.
- Prevention – by structurally preventing one of the four conditions necessary to cause a deadlock.
- Dynamic deadlock avoidance – by a careful resource allocation.

5. Deadlock detection and recovery

The first approach requires the system to have:

- Algorithm for detection – to examine system state and to determine, whether the deadlock has occurred,
- Algorithm for recovery of the system from the deadlock.

The recovery from deadlock is possible in principle in two ways:

a) To notify a system operator that may handle the problem.
b) To recover automatically either by killing one or more processes in a deadlock chain or by removing one or more resources involved in deadlock from their current owner. Both methods aim to break the circular wait condition.

In computer OS, it is evident that both automatic ways (part b) may cause damage of some data that must be repaired and corresponding processes in most cases must be rerun. In TS, the recovery can have also large harmful consequences on running processes, plus the recovery itself is in most situations a very complex task to be handled automatically. Thus we can expect the recovery interventions in TS to be done by a system operator in most cases. This depends on specific TS and specific situation that occurred.

6. Deadlock prevention

As prevention of a deadlock, it must be assured that at least one of the four Coffman conditions is broken. Let's see, if this is possible in TS.

To prevent the Mutual exclusion condition, there must be maximum one resource that cannot be shared by more processes at the same time. All other resources must have the attribute to be shared by more processes. In TS this would mean, for instance, that the same section of route could be occupied by more than one vehicle at the same time. In general, we can say that the prevention cannot be achieved by breaking this condition.

To prevent the Hold and wait condition means to force a process to have no resources at the moment, when it requests resources. This can be achieved in two ways:

- The process uses some resources and before asking for new, it must release all, and ask for them again,
- The process asks for all resources needed at the beginning and holds each of them until there is no need for them anymore.

The first way is practically impossible in TS. Once a vehicle occupies a section of route (e.g. road, track, water or air route), it cannot disappear before asking for additional resources.

The second way leads to loss of effectiveness. The more complex is the TS, the larger may be this loss. For instance, a train coming to a station would get assigned the whole route until the departure point from the station, even if it stands in the station for a long time, while other trains could come and leave the station – some of the tracks would be assigned with no utilization for too long time.

To break the No pre-emption condition means to allow taking away previously granted resources, regardless the situation, in which the process currently is. Two principal kinds of situation could be distinguished here: whether the resource that is to be revoked is currently in use or it is only reserved to the process. In case it is only reserved, it could be taken away. However, this is only a partial solution and we cannot ensure breaking of the condition for any situation when a resource is assigned to a process. In TS, vehicles usually occupy a section of route, and this, as a resource, cannot be practically taken away.

The way to prevent the Circular wait condition is to arrange all types of resources into an order and to demand processes to request resources according to ascending order of resource numbers. The order could respond to costs of assigning and release of resources from a process.

As an example, in a marshalling yard, the most expensive is most probably assigning and release of tracks (because this operation requires moving a train set that is processed between the tracks), after that, there are mobile resources (locomotives) and the least expensive are usually movements of members of crew. However, fulfilling this algorithm would imply taking the crew members away from the process in any case of a moving resource request or a track request, as well as taking the moving resource away from the process in the case of building a track route for the processed train set to move.

We can easily understand that the outlined way of breaking of the last condition results in reduced effectiveness of TS control.

7. Deadlock avoidance

Dynamic avoidance is an alternative to the structural prevention from deadlock. It is based on careful allocation of resources to processes after analysing information about requested and available resources. A system grants requested resources, only if it will result in a safe state of the system, i.e. the new state must not lead, under any circumstances, into circular waiting – the last of the Coffman conditions. If this is not guaranteed, the requesting process must wait until any of the assigned resources get released.

This analysis is to be carried out by each resource request. Available algorithms differ in what information they analyse. The simplest ones require from each process to state the maximum number of resources of each type needed for its execution. One example of these algorithms is the Banker's algorithm.

This method of dealing with deadlock situations is applicable in TS. The application depends on a job description and the way how the job description is represented and used in the control system. Anyhow, the control system will be obliged to extract the information about requested resources during the whole job processing.

If all the resources requested during the whole process will be taken into consideration in the beginning, it may happen that absence of a resource needed in the processing later than in its beginning will cause the waiting of the process before its start until the resource is available. That is why it looks to be a good idea not to consider resource requests for the whole process in the beginning but to divide the process into parts that are independent, and to run the analysis on each part separately before its turn comes. For instance, technological operations carried out on an airplane after its arrival and the ones on the same airplane before its departure from an airport could make up 2 parts of the airplane processing during its visit to the airport.

It could be even possible to separate two branches of the same process running in parallel and having no common resources. It could be done at the same time but not necessary. For instance, technical and traffic inspection of a cargo airplane.

Another way to make the control more effective would be finding a time estimate when a resource is supposed to be used in a given process. Duration of individual activities in a process can be estimated to a certain extent following experience and information about the processed object. This estimate can serve for creation of a time plan of resources use with possible utilization by optimisation of the resource assignment to processes as well. The time plan can be updated on-line as the information about running processes and expected jobs is received.

One of disadvantages of the Banker's algorithm may be its expensive application every time, when a request for a resource occurs. This strongly depends on a number of processes, resources and resource requests in a given time in the specific system. Higher complexity of system will result in a higher number of processes, in a higher number of resources and in more frequent resource requests, thus the algorithm will be more time demanding. TS have an advantage against computer OS in that the time needed for the algorithm to bring results should be relatively small compared to a duration of processes.

8. Conclusion

In this article I tried to outline analogy in control between an operating system in computers and transport systems. I focused on the deadlock theory and handling of deadlocks in the operating systems. We have found that handling methods used there can be also applied in automatic control in transport systems. From the analysis we can see that

- Detection and recovery from deadlock may be applicable, but recovery cost and complexity strongly depends on a concrete TS and a concrete deadlock situation.
- Prevention from deadlock by structural prevention of one or more conditions of deadlock is not applicable in most cases. Some of the conditions can be broken only for an unacceptable cost and loss in effectiveness of a system.
- Dynamic avoidance algorithms like the Banker's algorithm can be applied. They cost some time of a control subsystem but their effect should satisfy the need. The cost depends on dispositions of the concrete TS.

When comparing the conclusions we may understand that the most perspective way is probably the dynamic avoidance of deadlocks. However, the conclusions discussed in this article are made rather on a general level. They may vary from one transport system to another. It is also possible to apply rules that combine all three approaches to handling deadlocks - specifically for each system.

References

[1] CENEK, P.: *Operating Systems.* VŠDS, Žilina, 1989 (In Slovak)
[2] MARTINCOVÁ, P.: *Operating Systems.* ŽU, Žilina, 1997 (In Slovak)
[3] ŽARNAY, M.: *Analysis of Control Methods in Transport Systems.* Paper for academic dissertation examination, Žilina, 2001 (In Slovak)

komunikácie
COMMUNICATIONS

Katarína Bachratá *

EFFECTIVE BANDWIDTH FOR DETERMINISTIC NETWORKS

In present paper are given exact definitions of the studied notions. Some elementary properties of an effective bandwidth are deduced. Some terms and results are illustrated with examples for finite as well as for infinite packet flows in the internet network.

1. Introduction

Packets in packet switched communication networks create a random point process which can be described as it is known from the queueing theory. This randomness of packet inputs has an important impact to the quality of service that is offered by packet switched networks for real time applications. Therefore, it is necessary have a policy mechanism which controls a committed rate of incoming packets. The concept of the committed rate needs another description of the packet stream, which is known as an effective bandwidth. To use this concept in a proper way it is necessary to understand deeply stream behavior and consequently delay and packet loss introduced by an effective bandwidth as a link capacity. This paper explains a determinist case of the packet flow as a first step in understanding its random behavior.

These problems can be found in more abstract form in [1], [2].

2. Basic terms

The flow of packets will be modeled by the sequence $\{X(t_i)\}$ whose terms represent the number of packets arriving the communications system in discrete times

$t_0, t_1, t_2, t_3, \ldots$

In this paper the following terms will be used:

The channel is that part of Internet network, which transmits the packets.

Analogical term in the queueing theory is *the serving link*.

The capacity of a channel is the maximum amount of packets transmitted in the unit of time.

Analogical term in the queueing theory is *the service rate*.

Buffer is the place where the packets are stored in the case that they have to wait for transmission.

Analogical term in the queueing theory is *the queue*.

The buffer size (backlog) is the maximum number of packets stored in buffer and waiting for transmission.

Analogical term in the queueing theory is *the queue length*.

The communication system is the system consisting of a buffer and a channel.

Analogical term in the queueing theory is the *single server system with queue*.

The virtual delay at time t is the gap between the arrival of a packet into the communication system at the given time *t* and the start of packet transmission through the channel. Because at given time *t* there may be no real arrival of packet, we talk about virtual delay of fictive packet.

Analogical term in the queueing theory is *the queueing time plus service time* of a customer arriving the system at a given time *t*.

The flow of packets is a sequence of packets transmitted through the communication system.

Analogical term in the queueing theory is the *arrival process of customers*.

The effective bandwidth (of a given input flow) is the minimum capacity of the channel (or transmission speed, packet rate) with which communication system is able to transmit the given flow under a given delay or queue characteristics (precise definition is given below).

The effective bandwidth is measured in the packets per second [p/s] units.

In the queueing theory for the given flow in the *stable single-server system with the fixed (finite or infinite) queue length* the *number of served customers* per unit of time is computed. In other words, this is the *minimum service rate,* which guarantees sustainable servicing without overloading the system.

* **Katarína Bachratá**
Department of InfoCom Networks, Faculty of Management Science and Informatics, University of Žilina,
E-mail: Katarina.Bachrata@fri.utc.sk

3. Calculation of an effective bandwidth

Next we will assume the flow of packets represented by the sequence $\{X(t_i)\}$.

The number of packets arriving by flow X at times $t_0, t_1, t_2, t_3, ..., t_i, ...$ will be

$$X(t_0), X(t_1), X(t_2), X(t_3), ..., X(t_i), ...$$

To simplify the situation we assume that for the arriving times, $t_0, t_1, t_2, t_3, ...$, holds $t_i - t_{i-1} = 1$ second. At the beginning the flow is empty, so in time t_0 is $X(t_0) = 0$.

The arrival flow of packets may be *finite* (bounded in time), or *infinite* (unbounded in time).

We will present the computation method of the effective bandwidth for both cases separately.

1. Finite (time-bounded) flow of packets

The cumulative function $R_X(t)$ of the flow X counts the number of packets (seen on the flow in time interval $[0, t]$) arriving by the flow X from the time 0 until the time t:

$$R_X(t) = X(t_0) + X(t_1) + X(t_2) + ... + X(t)$$

Computation $efb_B(X)$ *of the effective bandwidth* of a *finite* flow of packets

$$X = \{ X(t_0), X(t_1), X(t_2), X(t_3), ..., X(t_k)\}$$

and for a fixed *buffer* size B:

$$efb_B(X) = \sup_{0 \le s \le t \le t_k} \left\{ \frac{R_X(t) - R_X(s) - B}{t - s} \right\} \qquad (1)$$

Computation $efb_D(X)$ *of the effective bandwidth* of a *finite* flow of packets

$$X = \{ X(t_0), X(t_1), X(t_2), X(t_3), ..., X(t_k)\}$$

and for a given *maximum virtual delay* D:

$$efb_D(X) = \sup_{0 \le s \le t \le t_k} \left\{ \frac{R_X(t) - R_X(s)}{t - s + D} \right\} \qquad (2)$$

Example 1:

Consider a flow where in times t_i $X(t_i)$ packets arrive.

i	0	1	2	3
$X(t_i)$	0	1	1	1

We perform the computation of the effective bandwidth of a flow X for a fixed buffer size B according to formula (1).

B	0	1	2	3	4	5
$efb_B(X)$	0	2/3	1/3	0	0	0

Commentary:

The minimum transmission speed (minimum capacity of channel) which is able to transmit the flow X in a system without buffer is

$$efb_{B=0}(X) = 1[p/s]$$

The requirement of zero buffer size (the system has no buffer) can be replaced by the requirement of zero delay time (packets mustn't wait).

Then we have:

$$efb_{B=0}(X) = efb_{D=0}(X)$$

The minimum transmission speed (minimum capacity of channel) which is able to transmit the flow X for the buffer size equal to 1 is

$$efb_{B=1}(X) = \frac{2}{3}\,[p/s]$$

It means when 1 packet can wait in the buffer then with a speed $2/3[p/s]$ in 3 seconds we transmit 2 packets with 1 packet waiting.

The request of a buffer size 1 means that 1 packet can wait.

The delays (waiting times) of packets can be computed from the transmission speed. According to the table the transmission speed is $2/3[p/s]$. In the buffer there is at most 1 packet waiting. The waiting time (the time to empty the full buffer) is 1.5 second.

The packets are arriving by the flow X for 3 seconds and afterwards 1.5 seconds are necessary to transmit the rest of the flow.

We have:

$$efb_{B=1}(X) = efb_{D=1.5}(X)$$

If the buffer size is 3 or more, we can let all the 3 packets of a flow X wait. In the extreme case there is no need for transmission, and the "transmission" of flow X can be realized by buffering all the packets. For this case the transmission speed 0 $[p/s]$ is sufficient.

It is clear that if we can use a buffer with sufficient size (and the packets in buffer can wait for an arbitrary time) for transmission of the finite flow a very small transmission speed is suitable. In an extreme case all packets are waiting in the buffer and no transmission is realized.

So it will be useful to restrict the delay for a packet instead of restricting the buffer size.

We compute the effective bandwidth values for the same flow X, for a fixed maximum delay D. According to the formula (2).

D	0	1	1.5	2	3	4
$efb_D(X)$	1	3/4	2/3	3/5	3/6	3/7

Commentary:

The minimum transmission speed (minimum capacity of channel) which is able to transmit the flow X when there are no delays for packets allowed is

$$efb_{D=0}(X) = 1[p/s]$$

The minimum transmission speed which is able to transmit the flow X if 1 second delays of packets are allowed is

$$efb_{D=1}(X) = \frac{3}{4}[p/s]$$

In 3 seconds from 3 packets arriving with flow X, 3 times 3/4 of packets are transmitted, it is 9/4 of packets on the whole. The remaining 3/4 of the packet will wait 1 second until it will be transmitted with speed 3/4 $[p/s]$.

According to the table it is not necessary to use only integer values of delays. If the packets can wait 1.5 second, the minimum transmission speed can be 2/3 $[p/s]$. Then 2 packets are transmitted in 3 seconds and then the last packet is transmitted 1.5 second (with speed 2/3 $[p/s]$) .

Example 2:

Consider a flow where in times t_i $X(t_i)$ packets arrive:

i	0	1	2	3	4	5	6	7
$X(t_i)$	0	1	5	0	0	2	0	1

The values of the effective bandwidth of flow X, for a fixed buffer size B.

B	0	1	2	3	4	5
$efb_B(X)$	5	4	3	2	1	0

The values of the effective bandwidth of flow X, for a fixed maximum delay D:

D	0	0.5	1	1.5	2	2.5
$efb_D(X)$	5	3.33	2.5	2	1.667	1.4286

If the effective bandwidth of flow X is smaller than the capacity of the channel c,

$$efb_D(X) \leq c$$

then it is possible to transmit the flow X through this channel.

For finite flows with zero sized buffer holds:

If we aggregate the flows $X_1, X_2, ..., X_k$ and compute the effective bandwidth of this sum, we obtain a value which is less than or equal to the sum of the effective bandwidths of these flows.

$$efb_B\left(\sum_{j=1}^{k} X_j\right) < \sum_{j=1}^{k} efb_B(X_j) \qquad (4)$$

When computing the effective bandwidth using the maximum delay, then the formula holds without constraints:

$$efb_D\left(\sum_{j=1}^{k} X_j\right) < \sum_{j=1}^{k} efb_D(X_j) \qquad (5)$$

This property can be used by transmission of more flows through the same channel. Although the sum of the effective bandwidths of some flows overreaches the capacity of some channel, still these flows can be transmitted through this channel. The effective bandwidth of the sum of these flows can be less than the capacity of the channel.

Example 3:

Consider the flows X and Y with the following number of packets in times t:

t	0	1	2	3	4	5	6	7	8	9	10
$X(t_i)$	0	2	0	2	0	2	0	2	0	2	0
$Y(t)$	0	1	3	0	2	0	4	0	0	1	3
$[X+Y](t)$	0	3	3	2	2	2	4	2	0	3	3

B	0	1	2	3	4	5
$efb_B(X)$	2	1	0.8889	0.7778	0.6667	0.5556
$efb_B(Y)$	4	3	2	1.2	1	0.9
$efb_B(X) + efb_B(Y)$	6	4	2.8889	1.9778	1.6667	1.4556
$efb_B(X + Y)$	4	3	2.3333	2.1667	2	1.9

D	0	1	2	3	4	5
$efb_D(X)$	2	1	0.9091	0.8333	0.7692	0.7143
$efb_D(Y)$	4	2	1.3333	1.125	1	0.9333
$efb_D(X) + efb_D(Y)$	6	3	2.2424	1.9583	1.7692	1.6476
$efb_D(X + Y)$	4	2.2857	2	1.8462	1.7143	1.6

Commentary:

In this example we can see that the relation (4) holds for zero value of B (buffer size). (That relation holds for such a small buffer that not too many packets will be stored and transmitted after arrival of the whole flow into the system.)

The relation (5) can be proved for an arbitrary finite delay D.

2. The computation of the effective bandwidth *efb(X)* for an infinite flow of packets

The flow of packets represented by sequence $\{X(t_i)\}$ will be transmitted through some channel. This channel will be represented by sequence $\{\hat{A}(n)\}$, which we call the envelope of the flow. We will search for the smallest envelope $\{\hat{C}(n)\}$ with sufficient properties. (We try to find the (minimum) smallest channel, which is able to transmit the given flow.)

We call the envelope of a sequence $\{X(t_i)\}$ such sequence $[\hat{A}(n)]$, which is the upper bound of the sum of arbitrary n consecutive terms of sequence $\{X(t_i)\}$:

$$\sup_{\forall i} [X(t_i) + X(t_{i+1}) + X(t_{i+2}) + ... + X(t_{i+n-1})] \leq \hat{A}(n) \quad (6)$$

Because the condition (6) is satisfied, the flow $\{X(t_i)\}$ can be transmitted through the channel $\{\hat{A}(n)\}$ ("it fits"). The sequence $\{\hat{A}(n)\}$ can be replaced by a n*ondecreasing and sub-additive* sequence $\{\hat{C}(n)\}$ which is also the envelope of the sequence $\{X(t_i)\}$.

We will explain why we require these properties and how we can assure them:

Because the sequence $\{X(t_i)\}$ represents the flow of packets, its terms are nonnegative:

$$X(t_i) > 0$$

The nondecreasing property of the envelope

$$\hat{B}(n) < \hat{B}(n+1) \quad \forall n \in N \tag{7}$$

means that the number of packets transmitted by the channel during a longer period $(n+1)$ is equal or greater than the number of packets transmitted by the channel during a shorter period (n).

We have:

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1})\} \leq$$

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1}) + X(t_{i+n})\} \leq \hat{A}(n+1)$$

and also

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1})\} \leq \hat{A}(n+1)$$

It follows: there exists a number β, which holds

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1})\} \leq \beta < \hat{A}(n+1)$$

Define now $\hat{B}(n) = \beta$.

$\{\hat{B}(n)\}$ is the envelope of the sequence $\{X(t_i)\}$ and in addition we assure that the envelope $\{\hat{B}(n)\}$ is nondecreasing and "smaller" than $\{\hat{A}(n)\}$:

$$\hat{B}(n) < \hat{A}(n) \quad \forall n \in N$$

The sub-additivity of envelope, it means the property

$$\hat{C}(n + m) + \hat{C}(n) + \hat{C}(m) \quad \forall n, m \in N \tag{8}$$

represents the fact that in the time period of length $m + n$, the channel can transmit at most such an amount of packets as in the time period m and n (these periods can start in an arbitrary moment).

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+m+n-1})\} =$$

$$\sup_{\forall i} \{X(t_i) + ... + X(t_{i+n-1}) + X(t_{i+n-1+1}) + ... +$$

$$+ X(t_{i+n-1+m})\} \leq \hat{B}(n) + \hat{B}(m)$$

For $\hat{B}(n) + \hat{B}(m)$ is the upper boundary of the value sup

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+m+n-1})\}$$

there exists a number γ, which holds

$$\sup_{\forall i} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+m+n-1})\} \leq \gamma \leq \hat{B}(n) + \hat{B}(m)$$

Let $\hat{C}(n + m) = \gamma$.

The sequence $\{\hat{C}(n)\}$ is the envelope of the sequence $\{X(t_i)\}$ and moreover we assure that the envelope $\{\hat{C}(n)\}$ is non-decreasing, sub-additive and

$$\hat{C}(n) < \hat{A}(n) \quad \forall n \in N$$

If the sequence $\{X(t_i)\}$ *is known*, from all of nondecreasing and sub-additive envelopes $\{\hat{C}(n)\}$ we can choose the smallest one $\{C^*(n)\}$.

$$\{C^*(n)\} = \min_{\forall \hat{C}} \{\{\hat{C}(n)\}\} \tag{9}$$

The terms of sequence $\{C^*(n)\}$ (the minimum envelope) can be also estimated as supreme of all the sums of n consecutive terms of the sequence

$$\{C^*(n)\} = \sup_{i \leq 0} \{X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1})\} \tag{10}$$

The value

$$c^* = \lim_{n \to \infty} \frac{C^*(n)}{n} \tag{11}$$

is the mathematical analogue of *the effective bandwidth*, (this is the *intensity* of the minimum envelope of flow X).

$$efb(X) = c^* \tag{12}$$

If the sequence $\{X(t_i)\}$ *is not known*, we can use the knowledge of any arbitrary envelope (any channel which was able to transmit the flow). We can minimize this envelope and transfer it into nondecreasing and sub-additive sequence, which still remains an envelope.

Example 4:

Consider the given envelope $\{\hat{A}(n)\}$ represented a channel which was able to transmit some flow.

We assume, that the flow represented by the sequence $\{X(t_i)\}$ was transmitted through channel $\{\hat{A}(n)\}$, whose capacity is:

n	1	2	3	4	5	6	7
$\hat{A}(n)$	8	6	7	21	14	25	19

Our goal is to minimize the envelope $\{\hat{A}(n)\}$ and preserve the non-decreasing property of the envelope. We change the values of the envelope setting:

$$\hat{B}(n) = \inf_{m>n} \hat{A}(m)$$

(Thus if for transmission of the flow X in $n = 2$ time units the channel with capacity $\hat{A}(2) = 6$ is sufficient, then for transmission of the flow X in $n = 1$ time unit the channel of capacity $\hat{B}(1) = 6$ is sufficient.)

n	1	2	3	4	5	6	7
$\hat{B}(n)$	6	6	7	21	14	25	19

In the same way we change the values $\hat{A}(4)$ and $\hat{A}(6)$.

n	1	2	3	4	5	6	7
$\hat{B}(n)$	6	6	7	14	14	19	19

$[\hat{B}(n)]$ is the envelope of the sequence $\{X(t_i)\}$ and we ensure that the envelope $[\hat{B}(n)]$ is non-decreasing and

$$\hat{B}(n) < \hat{A}(n) \quad \forall n \in N$$

Next we ensure the sub-additivity by setting:

$$\hat{C}(n + m) = \min[\hat{B}(n + m); \hat{B}(n) + \hat{B}(m)]$$

(So if for transmission in $n = 2$ time units the channel with capacity $\hat{B}(2) = 6$ is sufficient, then for transmission in $n = 2 + 2 = 4$ time units the channel with intensity $\hat{C}(4) = 6 + 6 = 12$ packets is sufficient.)

n	1	2	3	4	5	6	7
$\hat{C}(n)$	6	6	7	12	14	19	19

Using the same advance we recalculate the values $\hat{C}(5)$ and $\hat{C}(6)$.

n	1	2	3	4	5	6	7
$\hat{C}(n)$	6	6	7	12	13	14	19

The envelope $\{\hat{C}(n)\}$, which we obtain using this method is wide enough to transmit the flow, is non-decreasing and sub-additive.

If we know all of the possible nondecreasing and sub-additive envelopes of the given (and unknown) flow, we can write:

$$C^*(n) = \min_{\forall \hat{C}} [\hat{C}(n)] =$$

$$= \sup_{i \leq 0} [X(t_i) + X(t_{i+1}) + ... + X(t_{i+n-1})]$$

For sub-additive functions $f(t)$ holds

$$\lim_{n\to\infty} \frac{f(t)}{t} = \inf_{t \geq 1} \frac{f(t)}{t}$$

For the computation of c^* we can use formula:

$$c^* = \lim_{n\to\infty} \frac{C^*(n)}{n} = \inf_{t \geq 1} \frac{C^*(n)}{n}$$

For the infinite flow X can be proved:
If $efb(X) < \infty$, then there exists such $D \in R_0^+$, that the packets of the flow are waiting in the buffer for maximum delay D.

Moreover, the inequality (5) holds in the formSummary:

$$efb\left(\sum_{j=1}^{k} X_j\right) \leq \sum_{j=1}^{k} efb_B(X_j)$$

We have shown some examples of the effective bandwidths in the deterministic case.

The given examples illustrated some important properties of the characteristic of the flow of packets called effective bandwidth. The given formulas can by used for computing minimal envelopes.

References

[1] LE BOUDEC, J.-Y., THIRAN, P.: *Network Calculus, A Theory of Deterministic Queueing Systems for the Internet,* Springer-Verlag Berlin, Heidelberg 2001

[2] CHANG, CH.-SH: *Stability, Queue Length and Delay of Deterministic and Stochastic Queueing Networks,* IEEE Trans. Automatic Control, Vol. 39, pp. 913-931 (1994).

Juraj Hanuliak *

TO A PERFORMANCE EVALUATION OF PARALLEL ALGORITHMS IN NOW

A recent trend in high performance computing (HPC) is to use networks of workstations (NOW) as a cheaper alternative to massively parallel multiprocessors or supercomputers. In such parallel systems (NOW's) individual workstations are connected through widely used communication standard networks and co-operate to solve one large problem. Every workstation is treated similarly as a processing element in a conventional multiprocessor system. To make the whole system appear to the applications as a single parallel computing engine (a virtual parallel system), run-time environments such as PVM (Parallel virtual machine), MPI (Message passing interfaces) are often used to provide an extra layer of abstraction. In this paper, we discuss a new performance evaluation method on the example of multidimensional DFFT (Discrete Fast Fourier Transform) in a NOW's based on Intel's personal computers.

1. Introduction

There has been an increasing interest in the use of networks of workstations (cluster) connected together by high – speed networks for solving large computation-intensive problems [1, 6, 14, 15, 19]. This trend is mainly driven by the cost effectiveness of such systems as compared to large multiprocessor systems with tightly coupled processors and memories. Parallel computing on a cluster of workstations connected together by high – speed networks has given rise to a range of hardware and network related issues on any given platform. Performance prediction and evaluation, load balancing, inter-processor communication, and transport protocol for such machines are being widely studied. With the availability of cheap personal computers, workstations and networking devises, the recent trend is to connect a number of such workstations to solve computation-intensive tasks in parallel on such clusters.

The workstations can be connected using different network technologies such as off the shelf devices like Ethernet to specialised networks. Such networks and the associated software and protocols introduce latency and throughput limitations thereby increasing the execution time of cluster – based computation. Researchers are engaged in designing algorithms and protocols to minimise the effect of this latency [16, 18].

Network of workstations (Fig. 1.) has become a widely accepted form of high-performance parallel computing [1, 6, 14, 15, 19]. As in conventional multicomputers, parallel programs running on such a platform are often written in an SPMD form (Single – program – multiple data) to exploit data parallelism. Each workstation in a NOW is treated similarly to a processing element in a multicomputer system. However, workstations are far more powerful and flexible than the processing elements in conventional multicomputers. We can also use the advantages of the new SIMD (Single instruction Multiple data) instructions in the modern personal computers.

2. Effective parallel algorithms

The duty of a programmer is to develop an effective parallel algorithm for the given parallel system and for the given application problem. This task is more complicated in those cases, in which we have to create conditions for a parallel activity, that is through dividing the sequential algorithm to many mutual independent parts, which are named processes (decomposition strategy). Principally the development of the parallel algorithms includes the following activities [7, 16]:

- Decomposition – the division of the application into a set of parallel processes and data
- Mapping – the way in which processes and data are distributed among the computer elements of a used parallel system
- Inter-process communication – the way in which individual processes are cooperated and synchronized
- Tuning – performance optimisation of a developed parallel algorithm.

The most important step is to choose the best decomposition method for a given application [7, 16]. To do this it is necessary to understand the concrete application problem, the data domain, the used algorithm and the functional flow of activities in a given application.

3. Decomposition strategies

When designing a parallel program the description of the high-level algorithm must include, in addition to design, a sequential program the method you intend to use to break the application into processes and distribute data to different nodes – the decomposition strategy. The chosen decomposition method drives the rest of program development. This is true in case of developing a new

* **Juraj Hanuliak**
University of Žilina, Faculty of Management and Informatics, Veľký Diel, Unimo 2, 010 26, Žilina, Slovakia

application as porting serial code. The decomposition method tells you how to structure the code and data and defines the communication topology.

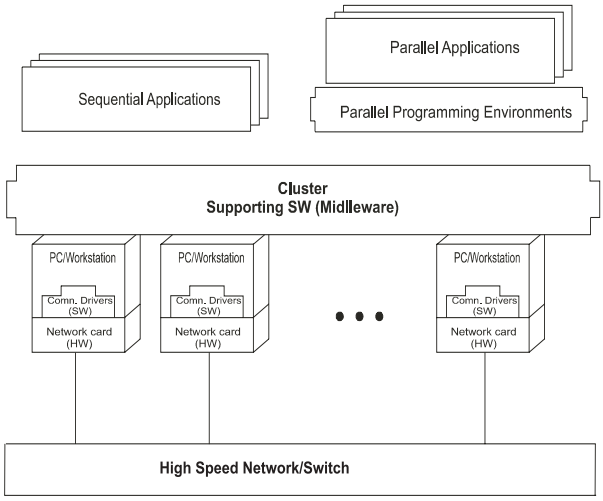


Fig. 1 Network of workstations- a principal structure.

To choose the best decomposition method for these applications, it is necessary to understand the concrete application problem, the data domain, the used algorithm and the flow of control in given application. According to a concrete application we can use the following decomposition models:

- Perfectly parallel decomposition
- Domain decomposition
- Control decomposition
- Object-oriented programming OOP (the latest modern programming technology)

3.1. Perfect parallel decomposition

Certain applications fall naturally into the category perfectly parallel. Perfectly parallel applications can be divided into a set of processes that require little or no communication with one another. Applications of this kind are usually the easiest to decompose. To this class belong, for example, all numerical integration algorithms. Obvious way to implement perfect parallelism is simply to run equivalent sequential programs on the various nodes, each with different data set. On a single processor system, this would require running each case sequentially.

3.2. Domain decomposition

Another decomposition technique is called domain decomposition. Problem subjects to domain decomposition are usually characterized by a large, discrete, static data structure. Decomposition "the domain" of computation, the fundamental data structures, provides the road map for writing the program.

3.3. Control decomposition

Another major decomposition strategy is called control decomposition. When there is no static structure or fixed determination of the numbers of objects or calculations to be performed, domain decomposition is not appropriate. Instead you can focus on the flow of control in the application. As the development progresses you will also distribute the data structures but the guideline to development remains the flow of control.

3.4. Object-oriented programming

Object-oriented programmers view applications as a set of abstract data structures or objects. Associated with these objects are tasks so that they are in no confusion about the parts of the code and data that affect other parts.

4. The theoretical part - The discrete Fourier Transform

The discrete Fourier transform (DFT) has played an important role in the evolution of digital signal processing techniques. It has opened new signal processing techniques in the frequency domain, which are not easily realisable in the analogue domain. The discrete Fourier transform (DFT) is defined as [3, 8, 17]:

$$X_n = \sum_{m=0}^{N-1} x_m \cdot w^{m,n}, \quad n = 0, 1, ..., N-1$$

and the inverse discrete Fourier transform (IDFT) as:

$$X_m = \frac{1}{N} \sum_{m=0}^{N-1} x_n \cdot w^{-m,n}, \quad m = 0, 1, ..., N-1,$$

in which w is N - root of unity that is $w = e^{-i(2p/N)}$ for generally complex numbers. In principle the mentioned equations are the linear transforms. Direct computations of the DFT or the IDFT, according to definitions require N^2 complex arithmetic operations. In such a way we could take into account only the calculation times and not also the overhead times caused through a parallel way of an algorithm implementation.

Cooley and Tukey [3, 7] developed a fast DFT algorithm which requires only $O(N. \log 2(N))$ operations. The difference in execution time between a direct computation of the DFT and the new DFFT (Discrete Fast Fourier Transform) algorithm is very large for large N. Direct computations of the DFT or the IDFT, according to the following program, requires N^2 complex arithmetic operations.

```
Program Direct_DFT;
var
   x, Y: array[0..Nminus1] of complex;
begin
   for k:=0 to N-1 do
      begin
         Y[k] :=x[0];
         for n:=1 to N-1 do
```

```
            Y[k] := Y[k] + Wnk * x[n];
        end;
end.
```

For example the time required for just the complex multiplication in a 1024-point FFT is $T_{mult} = 0.5.N \log_2(N).4.T_{real} = = 0.5.1024. \log_2(1024).4. T_{real}$, where the complex multiplication corresponds approximately to four real multiplication. The basic idea of Cooley and Tukey algorithm, which uses a divide-and-conquer strategy, is illustrated in Fig. 2. Several variations of the Cooley-Tukey algorithm have since been derived. These algorithms are collectively referred to as the DFFT algorithms.

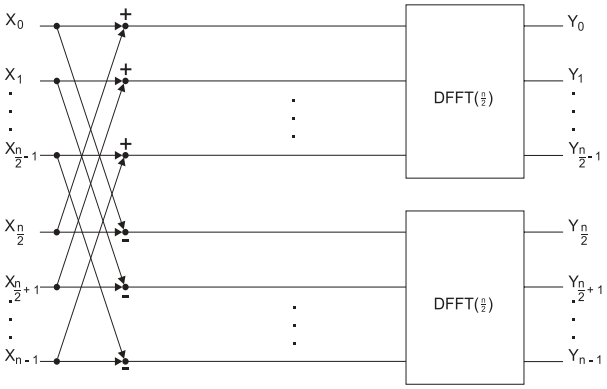


Fig. 2 Divide - and - conquer strategy for DFFT.

The basic form of parallel DFFT is the one-dimensional (1D), unordered, radix-2 (a use of divide and conquer strategy according to the principle in Fig. 2). The effective parallel computing of DFFT tends to computing one - dimensional FFT's with radix equals and greater than two and computing multidimensional FFT's by using the polynomial transfer methods. In practical part of this article we computed 2DFFT (two-dimensional DFFT). In general a radix-q DFFT is computed by splitting the input sequence of size s into a q sequences of size n/q each, computing faster the q smaller DFFT's, and then combining the result. For example, in a radix-4 FFT's, each step computes four outputs from four inputs, and the total number of iterations is $\log_4$ s rather than $\log_2 s$. The input length should, of course, be a power of four. Parallel formulations of higher -radix strategies (e. g. radix-3 and 5) 1-D or multidimensional DFFT's are similar to the basic form because the underlying ideas behind all sequential DFFT are the same. An ordered DFFT is obtained by performing bit reversal (permutation) on the output sequence of an unordered DFFT. Bit reversal does not affect the overall complexity of a parallel implementation.

5. Performance evaluation

To the performance evaluation of parallel algorithms we can use analytical approaches to get under given constraints some relations such as the known theorem of Munro - Paterson [7], Amdahl's law [14, 15], Gustafson's law [14, 15] etc. But all these relations have been derived in an idealised way without considering architecture and communication complexity. That means a complexity C_p is a function only of parallel algorithm calculation. Such assumption could be real in some centralised multiprocessor systems but not in NOW's (network of workstations based on personal computers).

In such a parallel system we have to take into account all complexity elements according to the relation $C_p = f$ (architecture, communication, calculation). In such a case we can use the following solution methods to get a complexity:

- Direct measurement - real experimental measure of P_p and its components for a concrete developed parallel algorithm on the concrete parallel system [7, 8]
- Analytical (to find C_p on basis of some closed analytical expressions or statistical distributions for overheads) [9, 10, 11, 12, 13]
- Simulation [2, 4, 5] (real experimental measure of C_p and its components for a concrete developed parallel algorithm) on NOW's.

6. The results

For direct measuring of complex performance evaluation in a NOW we used the structure according to Fig. 3.

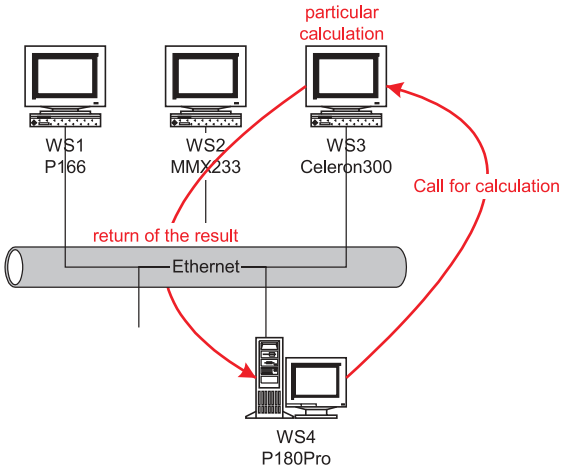


Fig.3 Illustration of the measure in NOW (Ethernet network)

The developed parallel algorithms were divided into the two logical parts - manager and servers. All programs were written on the WNT (Windows New Technology) platform. The manager controls the computer with starting services, makes connections and starts remote functions in a parallel way. At the end it sums the particular results. Every server waits for calculation starting and then calculates the particular results. At the end of calculation it returns to the manager the calculated results and the calculation time. The results are not only the computed results, but also the computation, communication and synchronization times (all overhead components for a given parallel algorithm). To measure these times we used the function "Query Performance Counter", which measures calculation times in ms.

Calibration power results in our experiments for 2DFFT are in Fig. 4. The achieved results for 2DFFT algorithm document increasing of both computation and communication parts in a geometricall way with the quotient value nearly four (increasing matrix dimension mean to do twice more computation on columns and twice more on rows). Therefore for a better illustration we used dependencies on relative input load. At these experiments we used computers according Table 1.

Parameters of the used personal computers Table 1.

Label	Processor	RAM [MB]	Operation system
WS1	Pentium I 233 MHz	64	Windows 2000
WS2	Pentium II 450 MHz	128	Windows 2000
WS3	Pentium III 933 MHz	256	Windows 2000
WS4	Pentium IV 1Ghz	512	Windows 2000
WS5	Pentium IV 1.4 Ghz	512	Windows 2000
WS6	Pentium IV 2.26 Ghz	1000	Windows 2000
WS7	Pentium IV Xeon – 2 proc., 2.2 GHz	1000	Windows 2000 Server

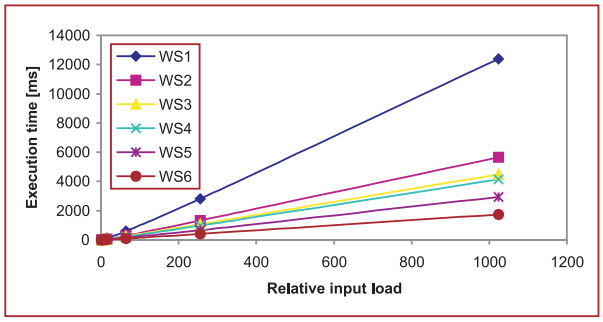


Fig. 4 Calibration power results for 2DFFT.

The results in Ethernet NOW are graphically illustrated for 2DFFT in Fig. 5. For a better graphical illustration we limited the measured values for WS1 network node. The achieved results for 2DFFT-algorithm document increasing of both computation and communication parts in a geometrical way with the quotient value nearly four. The influence of matrix dimension to the network load illustrates the Fig. 6.

Percentile amounts of the individual parts (computation, overheads – network load, initialisation) at 2DFFT execution time for the matrix 1024 × 1024 illustrate Fig. 7. The high network loads are involved through the needed matrix transpositions during 2DFFT computation.

7. Conclusions

Distributed computing was reborn as a kind of "lazy parallelism". A network of computers could team up to solve many problems at once, rather than one problem in higher speed. To get the most out of a distributed parallel system, designers and soft-

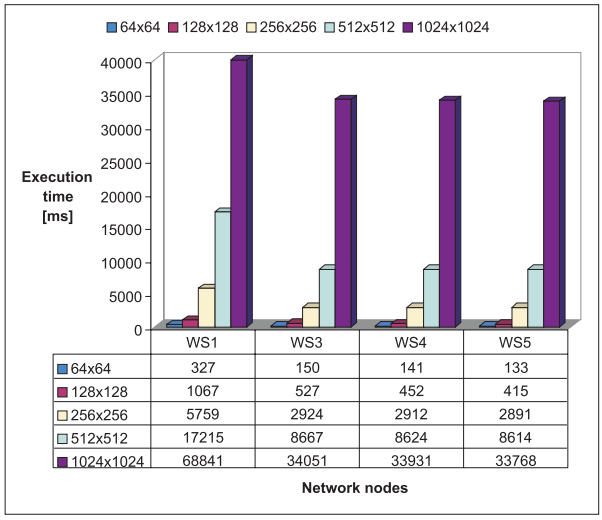


Fig. 5 Results in NOW for 2DFFT calculation (Ethernet network).

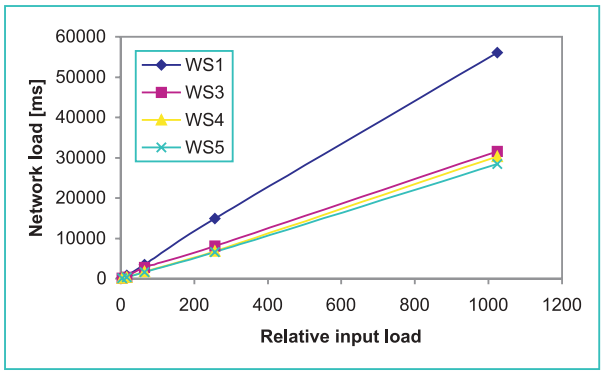
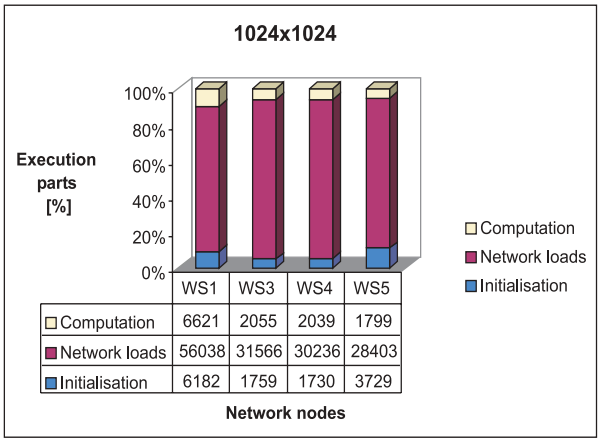


Fig. 6 The influence of matrix dimension to network load

ware developers must understand the interaction between hardware and software parts of the system. It is obvious that the use of a computer network based on personal computers would be principally less effective than the used typical massively parallel architectures in the world, because of higher communication overheads, but a network of workstations based on powerful personal computers, belongs to very cheap, flexible and perspective asynchronous parallel systems. This trend we can see in recent dynamic growth in the parallel architectures based on the networks of workstations as a cheaper and more flexible architecture in comparison to conventional multiprocessors and supercomputers. Second the used principles in world – realised multiprocessors are implemented in recent time in new symmetric multiprocessor systems (SMP), which are implemented on a single motherboard. Unifying of both mentioned approaches open the new possibilities in HPC computing in our country.

The next steps in the evolution of distributed parallel computing will take place on both fronts: inside and outside the box. Inside, parallelism will continue to be used by hardware designers to increase performance. Intel's new SIMD (Single instruction Multiple data) or MMX (Multimedia extensions) instructions tech-

nology, which implements a small/scale form of data parallelism, is one example. Out-of-order instruction execution by super-scalar processors in all latest powerful processors is another example of internal parallelism.



	WS1	WS3	WS4	WS5
Computation	6621	2055	2039	1799
Network loads	56038	31566	30236	28403
Initialisation	6182	1759	1730	3729

Fig. 7 The individual parts of the 2DFFT execution time (1024 × 1024)

In relation to achieved results at our faculty (Faculty of Control and Informatics, Zilina) we are able to do better load balancing among network nodes (performance optimisation of parallel algorithm). For these purposes we can use calibration results of network nodes in order to apply the input load according the measured performance power of used network nodes. Second we can do load balancing among network nodes based on modern SMP parallel systems and on network nodes or with only single processors. Generally we can say that the parallel algorithms with more communication overheads (similar to analyzed 2DFFT algorithm) will have better speed-up values for modern SMP parallel system as in its parallel implementation in NOW. For the algorithms with small or constant communication overheads (similar to π computation [7, 14, 15]) we can prefer to use the other network nodes based on single processors.

Queueing networks and Petri nets models, simulation, experimental measurements, and hybrid modelling have been successfully used for the evaluation of system components. Via the form of experimental measurement we illustrated the use of this technique for the complex performance evaluation of parallel algorithms. In this context I presented the first part of achieved results. We would like to continue in these experiments in order to derive more precise and general formulae (generalisation of the used Amdahl's and Gustafson's laws) and to develop suitable synthetic parallel tests (SMP, NOW) to predict performance in NOW for some typical parallel algorithms from linear algebra and other application oriented algorithms. In the future we will report about these results.

References

[1] ANDREWS, G. R., F*oundations of Multithreaded, Parallel,* and Distributed Programming, Addison Wesley Longman, Inc., 664 pp., 2000, USA
[2] BANKS, J., DAI, J. G.: *Simulation studies of multiclass queueing networks,* IEEE Transactions, Volume 29, 1997, pp. 213-219
[3] BASOGLU, CH., LEE, W., KIM, Y.: *An efficient FFT algorithm for Super-scalar and VLIW Processor Architectures,* Real Time Imaging 3, pp. 441-453, 1997, USA
[4] FODOR, G., BLAABJERG, S., ANDERSEN, *A.: Modelling and simulation of mixed queueing and loss systems,* Wireless Personal Communication, N. 8, 1998, pp. 253-276
[5] GREENBERG, D. S., PARK, J. K., SCHVABE, E. J.: *The cost of complex communication on simple networks,* Journal of Parallel and Distributed Computing 35, pp. 133-141, 1996
[6] HANULIAK, I.: *Parallel architectures - multiprocessors, computer networks* (in Slovak), 187 pages, July 1997, Book centre, Žilina, Slovakia
[7] HANULIAK, I.: *Parallel computers and algorithms* (in Slovak), Košice, Slovakia, ELFA Press 1999, 327 pp.
[8] HANULIAK, J.: *To a complexity of parallel algorithms,* In Proceedings: TRANSCOM 2001 (4-th European Conference in Transport and Telecommunications), 25-27 June 2001, pp. 51-54, Žilina, Slovakia
[9] HANULIAK, I.: *On the analysis and modelling of computer communication systems*, Kybernetes, The International Journal of Systems & Cybernetics, West Yorkshire, United Kingdom, Vol. 31, No. 5, pp., 715-730, 2002
[10] HANULIAK, I.: *Buffer management control in data transport network node,* The International Journal of Systems Architecture, Volume 47, Elsevier Science, Netherlands, pp. 529-541, 2001
[11] HANULIAK, M.: *To the behaviour analysis of mobile data networks,* in Proceedings of 7th Scientific conference, November 9-10, pp. 170-175, 2001, TÂRGU – JIU, Romania
[12] HARRISON, P. G., PATEL N.: *Performance modelling of communication networks and computer architectures,* Addison – Wesley Publishers 1993, 480 pp.
[13] HSU, W. T., PEN-CHUNG, Y.: *Performance Evaluation of Wire-Limited Hierarchical Networks,* Parallel and Distributed Computing 41, 1997, pp. 156 - 172
[14] HESHAM, E. L., REWINI, TED. G. LEWIS: *Distributed and parallel computing,* 467 pp., Manning Publications Co., 1997, USA

[15] HWANG, K. X., SCALABLE, Z.: *Parallel Computing:* Technology, Architecture, Programming, Mc Graw-Hill Companies, 802 pp., 1998, USA
[16] KUMAR, V., GRAMA, A., GUPTA, A., KARYPIS. G.: *Introduction to Parallel Computing* (Second Edition), Addison Wesley, 856 pp., 2001
[17] MARINESCU, D. C., RICE, J. R.: *On the scalability of asynchronous parallel computations,* Parallel and Distributed Computing 31, pp. 88-97, 1995
[18] NANCY, A. L.: *Distributed Algorithms*, 872 pp., 1996, Morgan Kaufmann Publishers, Inc., USA
[19] WILLIAMS, R.: C*omputer Systems Architecture - A networking approach,* Addison Wesley, 660 pp., 2001, England.

Jan Pelikán *

THE AIRCRAFT LANDING PROBLEM

The problem studied in the paper is an air traffic problem on the airport runway. The goal is finding an aircraft landing sequence that meets the time window for the particular aircraft and at the same time the separation times between two aircraft, which is necessary for the security of landings. The integer programming formulations and the relationship to the traveling salesman problem with cumulative costs are shown.

Keywords: *mathematical programming, integer programming, scheduling*

1. The scheduling aircraft landings problem

The problem is decision of the landing time for the set of planes, which are in the radar horizon of an air traffic controller, which involves the decision of sequencing of planes. There are two basic conditions for this time: the landing time has to lie within a specified time window and the landing times should follow a separation condition.

The lower bound of the time window for the particular plane depends on the distance of the plane from the airport and the speed of the plane, the upper bound of the time window depends on the amount of fuel. An economic speed for each plane determines the preferred landing time so called target-landing time.

The second main important constraint is the separation time between two planes. Each plane generates an air turbulence that can be dangerous for successive planes. The intensity of the turbulence depends on the type and weight of the plane. It must be specified certain time distance, separation time, between planes. There are two separation conditions:

a) complete separation conditions, if we have to ensure separation to all previous landing planes,
b) successive separation conditions, which ensure only separation to directly previous landing plane.

It can be proved that if the triangular condition for separation times is satisfied the successive separation conditions ensure the complete separation conditions. In other case the successive separation conditions are weaker than the complete separation conditions.

The goal is either a maximum number of planes scheduled in the time period or minimal mean landing time of all planes or minimal deviation of the landing times from appropriate target landing times.

The problem can be formulated for one or more runways, for landings only, or the plane take offs only or for both landings and take offs.

2. Mathematical model of the aircraft-landing problem with completes separation.

In [1] the following mixed integer zero-one formulation of the problem is presented.

Notation:
P number of planes
E_i the earliest landing time for plane i ($i = 1, 2, ..., P$)
L_i the latest landing time for plane i ($i = 1, 2, ..., P$)
T_i the target (preferred) landing time for plane i ($i = 1, 2, ..., P$)
S_{ij} the required separation time between plane i and plane j if plane i lands before plane j
$g_i > 0$ the penalty cost per time unit for landing before the target time T_i for plane i ($i = 1, 2, ..., P$)
$h_i > 0$ the penalty cost per time unit for landing after the target time T_i for plane i ($i = 1, 2, ..., P$)

It's supposed $E_i \leq T_i \leq L_i$, $i = 1, 2, ..., P$ and for all i, j.

The variables in the model are:
t_i the landing time for the plane i ($i = 1, 2, ..., P$)
$t_i^+ = \max\{0, T_i - t_i\}$ the landing time before target time
$t_i^- = \max\{0, T_i - t_i\}$ the landing time after target time

$x_{ij} = 1$ if plane i lands before (not necessarily directly) plane j, $i, j = 1, 2, ..., P$, $i \neq j$
$x_{ij} = 0$ otherwise.

Mathematical model of the aircraft landing problem with the complete separation:

$$\min \sum_{i=1}^{P} g_i t_i^+ + h_i t_i^- \qquad (1)$$

subject to

$$x_{ij} + x_{ji} = 1,\ i, j = 1, 2, ..., P,\ i \neq j \qquad (2)$$

$$t_i + S_{ij} - (L_i + S_{ij} - E_j)x_{ji} \leq t_j,\ i, j = 1, 2, ..., P,\ i \neq j \qquad (3)$$

* **Jan Pelikán**
University of Economics, Prague, E-mail: pelikan@vse.cz

$$E_i \le t_i \le L_i, \ \ i = 1, 2, ..., P \tag{4}$$

$$t_i + t_i^+ - t_i^- = T_i, \ \ i = 1, 2, ..., P \tag{5}$$

$$t_i, t_i^+, t_i^- \ge 0, i = 1, 2, ..., P, \ \ x_{ij} \in \{0,1\},$$

$$i, j = 1, 2, ..., P, i \ne j \tag{6}$$

The equation (2) means that either plane i lands before plane j or plane j lands before plane i. The landing time t_j should be greater than t_i with a difference which is the separation time S_{ij} (inequality (3) for $x_{ji} = 0$). If $x_{ji} = 1$, then inequality (3) is in the form $t_i - L_i + E_j \le t_j$ and it is always satisfied due to the inequality (4).

The landing time t_i should lie in the time window E_i, L_i (the inequality (4)).

The equation (5) defines variables t_i^+ and t_i^- which are the differences of t_i from the target time T_i. The variables t_i^+ and t_i^- are not defined by (5) uniquely, nevertheless the uniqueness of t_i^+ and t_i^- is guaranteed by the fact that $g_i > 0$ and $h_i > 0$ in the objective function (1).

Alternative objective function is the average landing time $(1/P)\sum_{i=1}^{P} t_i$. In this case the inequality (5) and the variables t_i^+ and t_i^- can be dropped.

Comment

For a given sequence of planes the determination of the optimal landing times is a linear programming problem. We can obtain this model by putting all variables x_{ij} to the appropriate values into the model (1)–(6).

3. The heuristic method

Because of NP hardness of the ALP (Aircraft Landing Problem) heuristic methods were proposed for the problem. One of them is the greedy approach [3] based on priorities numbers p_j^k, in which k-th plane is picked according to the lowest priority number p_j^k. The priority numbers are calculated as $p_j^k = \delta T_j + \epsilon EE_j^k + \alpha_j$, where δ, ϵ are priority weights and α_j is a perturbation of the priority.

EE_j^k is defined as the earliest time in which the plane j can land given by the previous sequence of planes, that is, if the partial sequence $s_1, s_2, ,s_{k-1}$ of planes is constructed already, then $EE_j^k = \max\{E_j, \max_{i<k}\{EE_{i,s} + S_{s,j}\}\}$. The next plane to land is

$$sk = \underset{j \notin \{s_1, s_2, ..., s_{k-1}\}}{\operatorname{argmin}}, p_j^k.$$

The earliest possible landing time for plane s_k is $EE_{s_k}^k$.

This heuristic will not necessary find a feasible landing sequence (it is possible that $EE_{s_k}^k > L_{s_k}$), in this case we can change the parameters α_j and try it again.

4. Mathematical model of the aircraft landing problem with successive separation.

In this section we will solve the problem in which only successive separation is enforced. If the triangular inequality $s_{ik} \le \le s_{ij} + s_{jk}$ for all $i \ne j \ne k$ holds, the successive separation is sufficient to ensure complete separation.

The aircraft-landing problem with successive separation can be viewed as an open traveling salesman problem with time windows, where nodes in this problem are the planes. The objective function is cumulative, so the special formulation, called traveling salesman problem with cumulative costs (or the deliveryman problem) should be used [4].

The following formulation of the aircraft-landing problem with successive separation is proposed.

Let $x_{ij} = 1$ if plane i lands directly before plane j, $i, j = 1, 2, ..., P, i \ne j$, $x_{ij} = 0$ otherwise.

$$\min \sum_{i=1}^{P} g_i t_i^+ + h_i t_i^- \tag{7}$$

$$\sum_{i=1}^{P} x_{ij} = 1, j = 0, 1, ..., P \tag{8}$$

$$\sum_{j=1}^{P} x_{ij} = 1, i = 0, 1, ..., P \tag{9}$$

$$t_i + S_{ij} - (L_i + S_{ij} - E_j)(1 - x_{ij}) \le t_j,$$

$$i, j = 0, 1, ..., P, i \ne j, j > 0, t_0 = 0 \tag{10}$$

$$E_i \le t_i \le L_i, \ \ i = 1, 2, ..., P \tag{11}$$

$$t_i + t_i^+ - t_i^- = T_i, \ \ i = 1, 2, ..., P \tag{12}$$

$$t_i, t_i^+, t_i^- \ge 0, i = 1, 2, ..., P, \ \ x_{ij} \in \{0,1\},$$

$$i, j = 1, 2, ..., P, i \ne j \tag{13}$$

The equations (8) and (9) assure that only one plane precedes and only one plane follows each plane. Plane 0 is artificial, so that $S_{oi} = S_{i0} = 0$ for all i.

5. Aircraft landing problem for multiple runways

There are two or more runways on large international airports. The plane-landing problem solves the question on which runway the plane will land and at which landing time. There are two different separation times:

a) separation times for two planes landing on the same runway S_{ij},
b) separation times for two planes landing on different runways s_{ij}.

It is assumed that $0 \leq s_{ij} \leq S_{ij}$.

The mathematical model (1)-(6) has to be modified so that the equation (3) for determination of landing times should be replaced by

$$t_i + S_{ij}z_{ij} + s_{ij}(1 - z_{ij}) - (L_i + S_{ij} - E_j)x_{ji} \leq t_j,$$

$$i, j = 1, 2, ..., P, i \neq j \qquad (14)$$

where variable z_{ij} equals 1 if planes i and j land on the same runway and equals 0 otherwise (the variable z_{ij} does not assign the planes to runways). For $z_{ij} = 1$ the constraint (14) corresponds the inequality (3), for $z_{ij} = 0$ it is exchanged Sij for sij in (3).

6. Computational results

Many computational results with using the models shown above [1] have been published.

The problem and models are tested by the author on the data sets provided by the OR problem library maintained by Beasley (http//mscmga.ms.ic.ac.uk/info.html). There are 8 problems in this data set, all of them were solved as the same model [1] and the results were compared. The computer system for the branch and bound method LINGO ver.7 was used and run on Pentium II. The results are presented in the table 1, where the original runtimes of the computation from [1] are written in the brackets (if they differ significantly).

Acknowledgement
The research was supported by the grant No. 402/03/1283 of the Grant Agency of the Czech Republic and research project CEZ:J18/98:311401001.

Numerical experiments Table 1.

No. data set	No. planes	No. Runways	Cost Heuristic m.	Opt. cost LINGO	Runtime (sec.)
1	10	1	1210	700	1
		2	120	90	1
		3	0	0	1
2	15	1	2030	1480	6
		2	210	210	3
		3	0	0	1
3	20	1	2870	820	4
		2	60	60	2
		3	0	0	2
4	20	1	4480	2520	548 (220)
		2	680	640	2754 (1919)
		3	130	130	75 (2299)
		4	0	0	2
5	20	1	7120	3100	1379 (920)
		2	1220	650	(11510)
		3	240	170	(1655)
		4	0	0	
6	30	1	24442	24442	2 (33)
		2	882	554	2482 (1568)
		3	0	0	3
7	44	1	3974	1550	37 (10)
		2	0	0	5
8	50	1	4390	1950	77 (111)
		2	260	135	301 (3450)
		3	0	0	9

References

[1] BEASLEY, J. E., KRISHNAMOORTHY, M ., SHARAIHA, Y.M., ABRAMSON, D.: *Scheduling aircraft landings - the static case.* Transportation Science, vol. 34 No. 2 (2000)
[2] BIANCO, L., MINGOZZI, A., RICCIARDELI, S.: *The Traveling Salesman Problem with Cumulative costs.* Networks, vol. 23 (1993)
[3] ERNST, A. T., KRISHNAMOOTHY, M., STORER, R. H.: *Heuristic and exact algorithms for scheduling aircraft landings.* Networks 34 (1999)
[4] FISCHETTI, M., LAPORTE, G., MARTELLO, S.: *The delivery man problem and cumulative matroids.* Operations Research vol. 41, no. 6 (1993)
[5] MĚLNÍČEK, S.: *Diplomová práce* VŠE 2002

Genia Ortis *

DESIGN OF A TRAFFIC MICROSIMULATION IN UML

From the programming point of view traffic microsimulations consist of concurrent, interacting processes. Several representations of the same type (i.e. a car) are simulated over time. This structure highly corresponds to the concept of object oriented programming, which is therefore the first choice for the implementation of a traffic microsimulation. However, object oriented software has to be designed througly before the first line of program code is writen. Otherwise, with the expansion of the software and the increase of the number of developers a serious danger of incompatibilities and incorrect model behaviour arises. There exists a standard to describe object oriented models, the Unified Modelling Language UML. It can be used to describe a software from different points of view and therefore facilitates acquaintance and coordination processes and even code generation. In this paper I give an introduction into UML especially for developers of traffic microsimulations.

1. Introduction

Within the framework of the EU project CETRA (CEnter for TRAnsportation research) I had the opportunity to work half a year at the Institute for Transportation Networks at Zilina University. Since several years, the institute has been doing research in the field of simulation. One of the resulting products which is successfully commercialised is Villon, a software to simulate marshalling yards.

Several years ago the idea arose to use the experience, concepts and possibly some tools of railway and truck simulation for a microsimulation of car traffic. Some components of it are already designed or even implemented, e.g. parts of the graphic representation and the data structure of the network graph.

The development and implementation of this model is partly done by students, who are interested in a certain feature of it. I was invited to work at the microsimulation development as well. In order to get acquainted with the subject and to provide a comprehensive introduction for future developers of the model, I decided to create a structured top down description to form a powerful traffic simulation software suite.

If this description is carefully kept up-to-date, all work will be preserved and new modules can be designed to be compatible with the existing ones.

The UML (the Unified Modelling Language) is a widespread standard to describe object oriented software like a traffic simulation. In this article I show how to use the UML to describe a traffic simulation model. It is not my intention to provide a full description of the model but to illustrate with several examples how the UML is applied.

2. Components of a Microsimulation

2.1 Overview of existing traffic microsimulation tools

In order to get an overview of traffic microsimulations, I made a literature and internet review. A detailed description was found about the following models:

Traffic Microsimulations (not complete) Table 1

aaSIDRA	Akcelik & Associates Pty Ltd	http://www.aatraffic.com
AIMSUN 2	TSS – Transport Simulation Systems	http://www.tss-bcn.com/aimsun.html
CORSIM	Trafficware Corporation	http://www.trafficware.com
DRACULA	ITS, University of Leeds	http://www.its.leeds.ac.uk/projects/smartest
HUTSIM	Helsinki University of Technology	http://www.hut.fi/Units/Transportation/Research/hutsim.html
NEMIS	MIZAR Automazione S.p.a. Torino	–
SimTraffic	Trafficware Corporation	http://www.trafficware.com
SITRA B+	ONERA-CERT	http://www.cert.fr/fr/dcsd/CD/CDPUB/FINALITES/trafic.html
TRANSIMS	TSA-4, Los Alamos National Laboratory	http://www.transims.tsasa.lanl.gov
VISSIM	PTV	http://www.english.ptv.de/produkte/vissim.pl

* **Genia Ortis**
Department of Transportation Networks, Faculty of Management Science and Informatics, University of Žilina,
Tel.: +44-7952-641161, E-mail: genia.ortis@i-one.at

Object structure of the traffic microsimulation Table 2

basic item	representation in prototype	attributes	options for further development
Vehicle	car	kinematics, acceleration/deceleration rates, max. speed, driver aggressivity	split into other means of transport and pedestrians, characteristics of man/driver (i.e. aggressiveness)
network infrastructure	single net		several networks: public transport, car, pedestrian, bycicle, ...
	node	coordinates	
	edge	lane width and direction, allowed speed, length	
control infrastructure	traffic light	predefined signal control program	adaptive (to traffic conditions), coordinated (groups of traffic lights), optimisation of signal control program
	traffic panel	fixed content	variable message sign
traffic generation	single matrix	vehicles/hour foreach OD-pair	several matrices for several modes of transport
Time	timestamp		

As a result, objects, functions and results of the investigated traffic microsimulations were listed. This list forms the base for the UML model.

Table 2 shows the basic objects occurring in the traffic microsimulation. To ensure extensibility, known options for further development should be considered from the very beginning of the project.

This is the suggested structure of objects and attributes of the microsimulation to be designed. The initial UML-description of the software will be based on this structure.

It is necessary to define functions (i.e. car following, dynamic route guidance) and output parameters (i.e. emissions, travel time) of the microsimulation as well. They will be used to refine the initial UML structure.

3. Design of the Object Oriented Model in UML

UML is a standard defined by the Object Management Group (OMG) and is a widely spread toolset to define object oriented processes. "The Unified Modeling Language UML is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components." [OMG Unified Modeling Language Specification, Version 1.4, September 2001, see http://www.omg.org/cgi-bin/doc?formal/01-09-67, p. xix (Foreword)]

There exist many free and commercial applications, which support modeling in UML. The decision for one of them can be made according to the technical environment (operating system, programming language) and the requirements of the project (use as a project management tool as well, project size, team size). The following alternatives were considered for the traffic microsimulation prototype:

UML tools (not complete) Table 3

Free software:	"ArgoUML", a CASE-tool in/for Java
	"Ideogramic", a lightweight, intuitive UML-tool
	"UML modeller", a sourceforge project for Linux only
	"QuickUML"
Commercial software:	"Select Enterprise 6.2": a powerful project management tool,
	"Model Maker": a Module of the Delphi Enterprise edition
	"Poseidon": the commercial version of ArgoUML with additional OLE-functions
	"MagicDraw"
	"Rational Rose"
	"Together"

There exists an XML-based standard to exchange metadata like UML-models, named XMI. With XMI software specifications can be exchanged between different versions of the same software or even between different software tools.

UML documentation used to define the traffic microsimulation Table 4

Link	description	language
http://www.omg.org/cgi-bin/doc?formal/01-09-67	the formal UML specification	English
http://argouml.tigris.org/documentation/pdf/manual/argomanual.pdf	ArgoUML documentation	English
http://www.embarcadero.com/products/describe/umltutorials.asp	UML Tutorial	English
http://www.geocities.com/SiliconValley/Network/1582/uml-example.htm	UML by examples	English
http://www.rittershofer.de/info/uml/	UML Tutorial	German
http://ivs.cs.uni-magdeburg.de/%7Edumke/UML/index.htm	UML Tutorial	German

The internet provides a wide range of documentation about UML for different purposes and levels of knowledge. Table 4 contains a list of the UML descriptions and tutorials I referred to.

UML defines 9 types of diagrams, 4 describing the structure and 5 the behaviour of a software. Not all of them have to be used for the particular application. 5 different types of diagrams are sufficient to describe a traffic microsimulation. The diagrams can be drawn in any order, the sequence of the diagrams described in this paper is recommended to initially define the top-level description of the traffic microsimulation. Then, according to the progress of the software development, particular diagrams have to be refined down to the level of implementation.

3.1 Use Case Diagram

Starting point of UML modeling is the transfer of the user defined software behaviour (in human language) into a so called "Use Case Diagram". A Use Case Diagram contains the interactions (="Use Cases", drawn as ellipses) between Actors (User, Store) and the software.

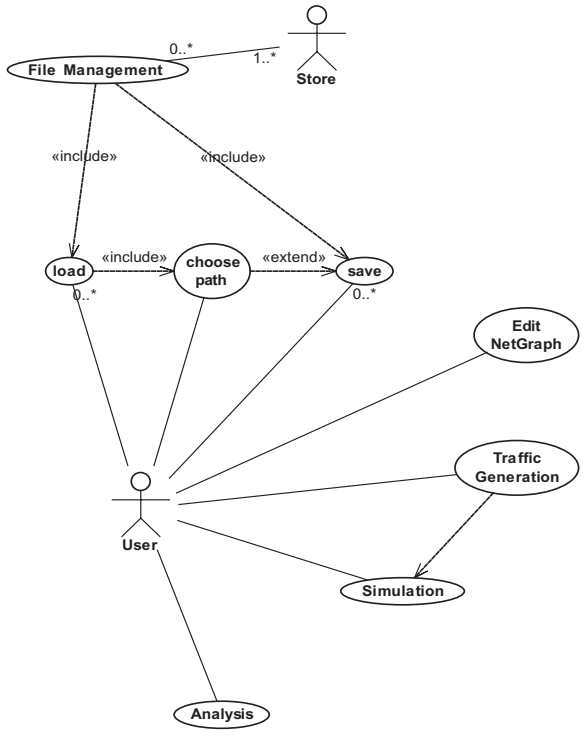


Fig. 1 Use Case Diagram

The association lines between actors and use cases represent communication paths. The association ends can be labeled with a multiplicity description.

Between use cases can exist different dependencies which are drawn as dashed arrows. A change to the target element may require a change to the source element in the dependency. The use case "load" "includes" the use case "choose path" because a filename and a path always have to be chosen. The use case "choose path" may "extend" the use case "save" subject to specific conditions ("save" vs. "save as").

3.2 Class Diagram

The object structure of the traffic microsimulation listed in table 2 is the base for the next step of UML modeling: The software structure is designed in a "Class Diagram". Classes, their methods and attributes as well as relations, (e. g, generalization) multiplicities and access paths between classes are visualized in the Class Diagram (see Fig. 2). Generalizations (i.e. inheritance) are visualized by a hollow diamond at the end of the more general object.

The result is a framework for the implementation. If there is a code-generator available for the chosen programming language, the skeletal source-code can be generated automatically.

3.3 Describing the Model Behaviour

To further specify the structure several behavioural diagrams are available in the UML. "Sequence Diagrams" and "Collaboration Diagrams" are used to describe the interaction between objects.

The Sequence Diagram has a timeline from top to bottom and shows sequential and concurrent flows within a process. The lifetime of an object instance is indicated by a bar along the dashed timeline. A solid arrow is a procedure call, while a dashed arrow is the return from a procedure.

The Collaboration Diagram focuses on the relationships among the objects and is a tool to define the procedures and functions in detail. The numbers correspond to the sequence of the procedure calls, nested numbers indicate nested procedure calls.

3.4 Describing the Behaviour of Single Objects

"State chart Diagrams" and "Activity Diagrams" focus on single objects and serve as a further specification of them.

Activity Diagrams are appropriate where procedures are triggered by the completion of previous procedures and define the sequence of actions within a procedure.

Alternatively in a State Chart Diagram states and transitions of an object triggered by asynchronous events during runtime are visualized.

4. Summary

The introduced technique is very well suitable to the development of a traffic microsimulation. Because of its object oriented

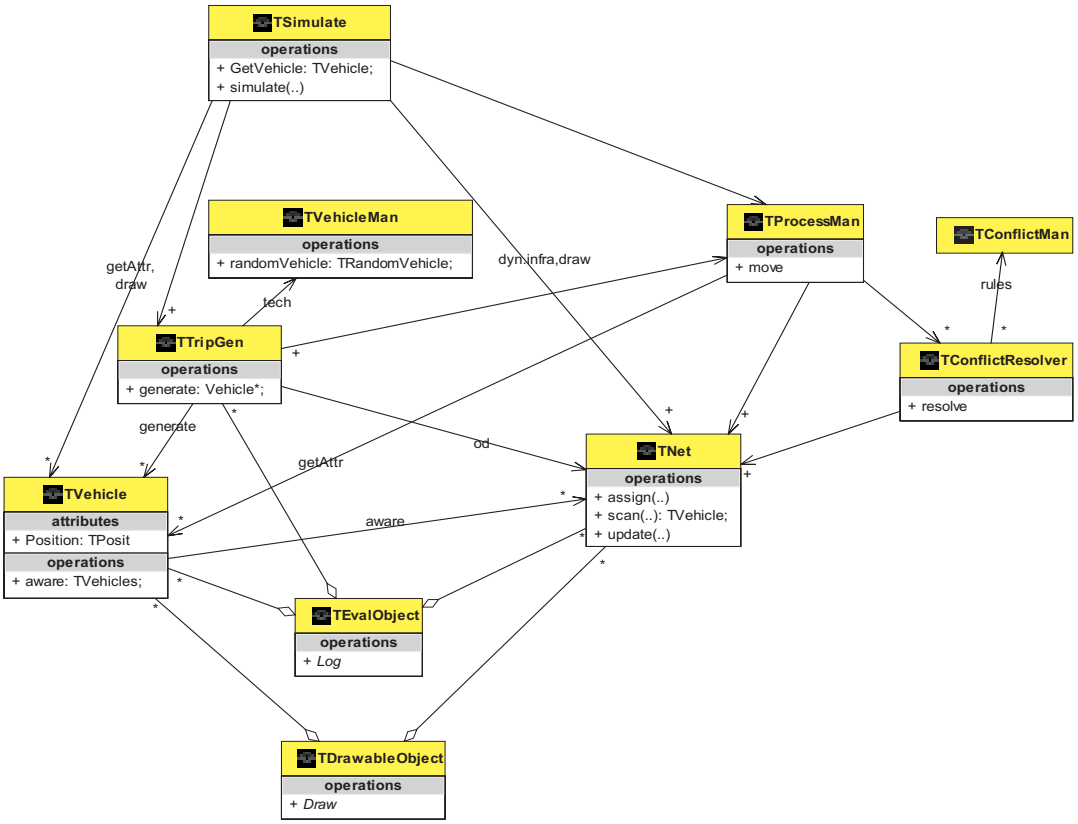


Fig. 2 Class Diagram of the Simulation Process

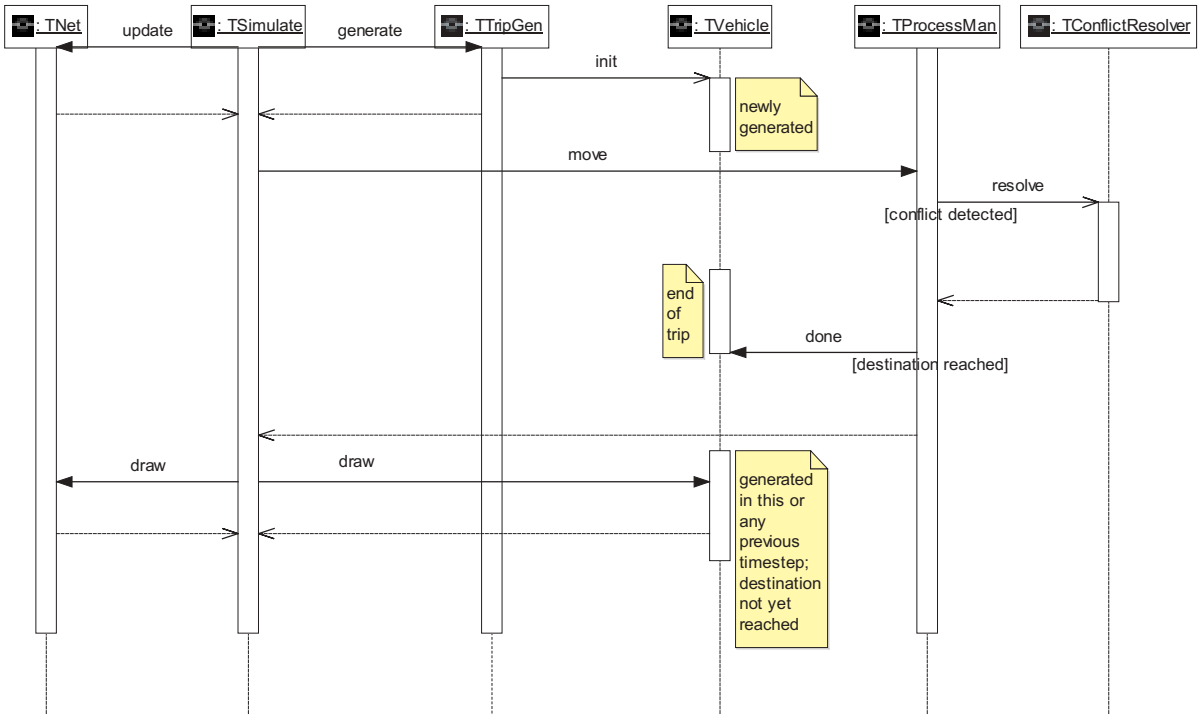


Fig. 3 Sequence Diagram of one simulation timestep

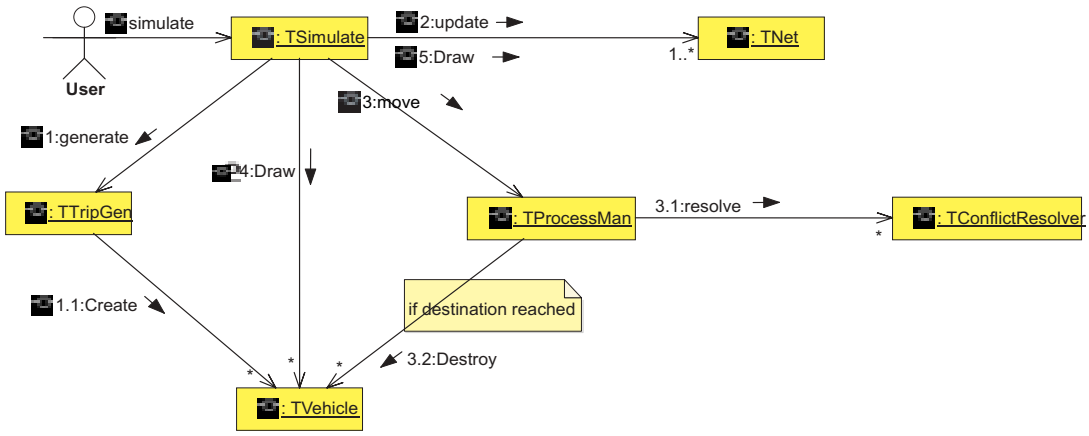


Fig. 4 Collaboration Diagram of one simulation timestep

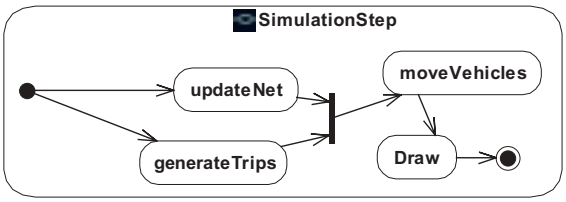
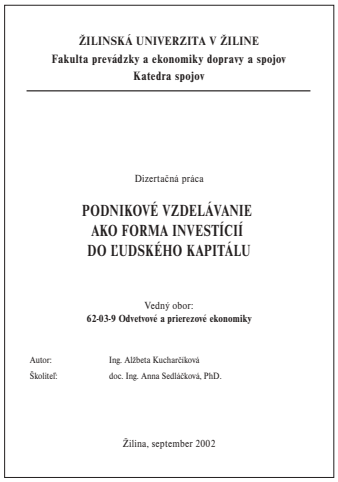


Figure 5 Activity Diagram of one simulation timestep

character the development of a simulation software requires a careful design process, before the first line of code can be written. The top-down approach with the UML enables an intuitive understanding of the necessary objects within the complex software structure. A consequent usage of the UML assures a consistent and flexible development-environment for a traffic simulation software suite. The reduced cost of software coordination and reengineering will many times pay the initial effort of learning and applying the UML.

Further information about the UML can be found in the documents listed in table 2.

Názov doktorandskej dizertačnej práce: **Podnikové vzdelávanie ako forma investícií do ľudského kapitálu**
Autor: **Ing. Alžbeta Kucharčíková**
Vedný odbor: **62-03-9 odvetvové a prierezové ekonomiky**
Školiace pracovisko: **Fakulta prevádzky a ekonomiky dopravy a spojov Žilinskej univerzity**
Školiteľ: **doc. Ing. Anna Sedláčková, PhD.**



ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta prevádzky a ekonomiky dopravy a spojov
Katedra spojov

Dizertačná práca

PODNIKOVÉ VZDELÁVANIE
AKO FORMA INVESTÍCIÍ
DO ĽUDSKÉHO KAPITÁLU

Vedný obor:
62-03-9 Odvetvové a prierezové ekonomiky

Autor: Ing. Alžbeta Kucharčíková
Školiteľ: doc. Ing. Anna Sedláčková, PhD.

Žilina, september 2002

Resumé:

Vzdelávanie je jednou z významných foriem investícií do ľudského kapitálu. Tento význam sa zvyšuje práve v súčasnom období, ktoré je charakteristické globalizačnými procesmi, technickými a technologickými inováciami, tvorbou a zavádzaním vyspelých informačných technológií, zavádzaním zmien vo vnútropodnikových procesoch atď.

Dizertačná práca sa zaoberá problematikou ľudského kapitálu, ktorý je v súčasnosti považovaný za najvýznamnejší zdroj ekonomického rastu a to v zmysle rozpracovanej teórie. Definícia ľudského kapitálu je rozšírená s ohľadom na súčasné svetové ekonomicko-politické trendy a objasnený je vzťah medzi kategóriami ľudský kapitál – ľudský potenciál a ľudský kapitál – ľudské zdroje.

Následne je uskutočnená analýza vzdelávania ako formy investícií do ľudského kapitálu z makro i mikropohľadu. Nakoľko vzdelávanie predstavuje veľmi širokú ekonomicko-sociálnu kategóriu, v dizertačnej práci sú rozobrané jednotlivé fázy a prvky procesu vzdelávania realizovaného podnikmi. Z dôvodu efektívnejšej realizácie jednotlivých fáz vzdelávania v praxi je analýza doplnená vlastnými návrhmi a odporúčaniami, ktoré napomôžu efektívnejšie dosiahnuť ciele, plynulo a bezproblémovo zrealizovať vzdelávací program a zvýšiť motiváciu účastníkov k učeniu sa.

Na to, aby vzdelávanie prinášalo očakávané pozitívne efekty, je nevyhnutné uskutočňovať vyhodnocovanie vzdelávania. Je to proces náročný, nakoľko nie všetky náklady, ale hlavne úžitky, ktoré vzdelanie prináša, sa dajú presne kvantifikovať. Navrhnutý je preto nepriamy spôsob vyčíslenia úžitkov vzdelávania pomocou prevodovej schémy a tiež konkrétne prístupy k hodnoteniu efektívnosti investícií do podnikového vzdelávania. Pri realizácii vyhodnocovania podnikového vzdelávania sa odporúča použiť navrhnutý 6-úrovňový model vyhodnocovania vzdelávania a pre každú jeho úroveň súbor metód a odporúčaní i postupov pre jeho efektívnejšiu realizáciu v praxi.

Praktickým prínosom doktorandskej dizertačnej práce je návrh aplikačného modelu prepojenia vzdelávania, rozvoja a úspešnosti podniku na trhu. Model ukazuje, či vzdelávanie pomáha podniku dosahovať ciele, prípadne riešiť problémy zistené počas fázy identifikácie a analýzy vzdelávacích potrieb, čím zároveň umožňuje podniku úspešne napredovať.

COMMUNICATIONS – Scientific Letters of the University of Žilina
Writer's Guidelines

1. Submissions for publication must be unpublished and not be a multiple submission.
2. Manuscripts written in English language must include abstract also written in English. The submission should not exceed 7 pages (format A4, Times Roman size 12). The abstract should not exceed 10 lines.
3. Submissions should be sent: **by e-mail** (as attachment in system Microsoft WORD) to one of the following addresses: *holesa@nic.utc.sk* or *vrablova@nic.utc.sk* or *polednak@fsi.utc.sk* **with a hard copy** (to be assessed by the editorial board) **or on a 3.5″ diskette** with a hard copy to the following address: Zilinska univerzita, OVaV, Moyzesova 20, SK-10 26 Zilina, Slovakia.
4. Abbreviations, which are not common, must be used in full when mentioned for the first time.
5. Figures, graphs and diagrams, if not processed by Microsoft WORD, must be sent in electronic form (as GIF, JPG, TIFF, BMP files) or drawn in contrast on white paper, one copy enclosed. Photographs for publication must be either contrastive or on a slide.
6. References are to be marked either in the text or as footnotes numbered respectively. Numbers must be in square brackets. The list of references should follow the paper (according to **ISO 690**).
7. The author's exact mailing address, **full names, e-mail address, telephone or fax number, and the address of the organisation where the author works** and contact information must be enclosed.
8. The editorial board will assess the submission in its following session. In the case that the article is accepted for future volumes, the board submits the manuscript to the editors for review and language correction. After reviewing and incorporating the editor's remarks, the final draft (before printing) will be sent to authors for final review and adjustment.
9. The deadlines for submissions are as follows: September 30, December 31, March 31 and Juny 30.

POKYNY PRE AUTOROV PRÍSPEVKOV DO ČASOPISU KOMUNIKÁCIE – vedecké listy Žilinskej univerzity

1. Redakcia prijíma iba príspevky doteraz nepublikované alebo inde nezaslané na uverejnenie.
2. Rukopis musí byť v jazyku anglickom. Príspevok by nemal prekročiť 7 strán (formát A4, písmo Times Roman 12 bodové). K článku dodá autor **resumé** v rozsahu maximálne 10 riadkov (v anglickom jazyku).
3. Príspevok prosíme poslať: **e-mailom**, ako prílohu spracovanú vo Word-e, na adresu: *holesa@nic.utc.sk* alebo *polednak@fsi.utc.sk* príp. *vrablova@nic.utc.sk* (alebo doručiť na diskete 3,5″) **a jeden výtlačok** článku na adresu Žilinská univerzita, OVaV, Moyzesova 20, 010 26 Žilina.
4. Skratky, ktoré nie sú bežné, je nutné pri ich prvom použití rozpísať v plnom znení.
5. Obrázky, grafy a schémy, pokiaľ nie sú spracované v Microsoft WORD, je potrebné doručiť buď v digitálnej forme (ako GIF, JPG, TIFF, BMP súbory), prípadne nakresliť kontrastne na bielom papieri a predložiť v jednom exemplári. Pri požiadavke na uverejnenie fotografie priložiť ako podklad kontrastnú fotografiu alebo diapozitív.
6. Odvolania na literatúru sa označujú v texte alebo v poznámkach pod čiarou príslušným poradovým číslom v hranatej zátvorke. **Zoznam použitej literatúry** je uvedený za príspevkom. Citovanie literatúry musí byť **podľa záväznej STN 01 0197 (ISO 690)** „Bibliografické odkazy".
7. K rukopisu treba pripojiť **plné meno a priezvisko autora a adresu inštitúcie v ktorej pracuje, e-mail adresu** a číslo telefónu alebo faxu.
8. Príspevok posúdi redakčná rada na svojom najbližšom zasadnutí a v prípade jeho zaradenia do niektorého z budúcich čísiel podrobí rukopis recenziám a jazykovej korektúre. Pred tlačou bude poslaný autorovi na definitívnu kontrolu.
9. Termíny na dodanie príspevkov do čísiel v roku sú: 30. september, 31. december, 31. marec a 30. jún.

KOMUNIKÁCIE
COMMUNICATIONS

VEDECKÉ LISTY ŽILINSKEJ UNIVERZITY
SCIENTIFIC LETTERS OF THE UNIVERSITY OF ŽILINA
5. ROČNÍK – VOLUME 5

Adresa redakcie:
Address of the editorial office:
Žilinská univerzita
Oddelenie pre vedu a výskum
Office for Science and Research
Moyzesova 20, Slovakia
SK 010 26 Žilina
Tel.: +421/41/5620 392
Fax: +421/41/7247 702
E-mail: polednak@fsi.utc.sk, holesa@nic.utc.sk

Každý článok bol oponovaný dvoma oponentmi.
Each paper was reviewed by two reviewers.

Časopis je excerpovaný v Compendexe
Journal is excerpted in Compendex

Vydáva Žilinská univerzita
v EDIS – vydavateľstve ŽU
J. M. Hurbana 15, 010 26 Žilina
pod registračným číslom 1989/98
ISSN 1335-4205

It is published by the University of Žilina in
EDIS - Publishing Institution of Žilina University
Registered No: 1989/98
ISSN 1335-4205

Objednávky na predplatné prijíma redakcia
Vychádza štvrťročne
Ročné predplatné na rok 2004 je 500,– Sk

Order forms should be returned to the editorial office
Published quarterly
The subscription rate for year 2004 is 500 SKK.

Jednotlivé čísla časopisu sú uverejnené tiež na:
Single issues of the journal can be found on:
http://www.utc.sk/komunikacie