Int. J. Appl. Math. Comput. Sci., 2010, Vol. 20, No. 2, 391–400
DOI: 10.2478/v10006-010-0029-z

amcs

MINIMIZATION OF THE TOTAL COMPLETION TIME FOR ASYNCHRONOUS TRANSMISSION IN A PACKET DATA–TRANSMISSION SYSTEM

ADAM PIÓRKOWSKI *, JAN WEREWKA **

* Department of Geoinformatics and Applied Computer Science
AGH University of Science and Technology, al. Mickiewicza 30, 30–059 Cracow, Poland
e-mail: pioro@agh.edu.pl

**Computer Science Laboratory, Institute of Automatics
AGH University of Science and Technology, al. Mickiewicza 30, 30–059 Cracow, Poland
e-mail: werewka@ia.agh.edu.pl

The minimization of the total completion time for asynchronous transmission in distributed systems is discussed. Attention is focused on the problem of message scheduling on part of the sender. Messages to be sent form a queue, and the order in which they are to be sent has to be first established. The methods of scheduling messages, which minimize the factor of the total completion time, are presented herein. The message-scheduling problem becomes considerably complicated when the stream of data transmitted between the sender and the receiver is organized into packets. A scheduling rule, according to which the shortest messages (SPT—Shortest Processing Time) are selected as the first to be sent, has been proven to be appropriate for the proposed model. A heuristic algorithm for scheduling messages with real-time constraints is proposed. The performance of the scheduling algorithm is experimentally evaluated. The results of the study show the possibility of improving the total completion time from a few to ten percent, depending on the characteristics of the sender. Thus, the practicability of the method has been proved.

Keywords: message scheduling, message queuing, distributed systems, real-time systems.

1. Introduction

Asynchronous communication is the most popular method of exchanging data in distributed systems. This is because of the numerous advantages provided by this type of communication and due to the fact that the sender is not blocked during the sending of the message; in addition, the ascribed transmission bandwidth between the communicating nodes can be completely used. Network transmission services are frequently recognized by Message Queuing Systems (MQS), which are part of middleware. The data transmitted through such a system are formatted to a message of a specific length and priority. Messages that cannot be sent at any given moment are deposited in queues, waiting to be sent later on when the transmission channel becomes free.

The scheduling of messages to be sent (by the sender) has a significant influence on the quality of data transmission. Message-scheduling methods are usually worked out on the basis of solutions proposed for task scheduling. There are numerous analogies between multitask or production systems and asynchronous communication. Data-transmission systems also have their peculiarities, making them different from other systems. One such specific property is stream communication, which represents a continuous data transmission between the nodes (stream). The stream is divided into packets of uniform size. Some messages contain data that can be fully organized in one packet; other messages that contain data in excess of the size of the packet are fragmented and sent in many packets. Scheduling with this specific property is the focus of the discussion.

Scheduling algorithms to be applied for real systems should have low computational complexity. This requirement results from the fact that delays in sending data caused by the scheduling of messages should be minimized. Moreover, if the transmission system is ready to send another packet and the data to be sent are available, then the packet should be sent regardless of the computation stage of the scheduling algorithm optimization. Hence, an algorithm that provides better or equally good

solutions for every successive computation step is investigated.

Furthermore, it is assumed in the article that network communication is the bottleneck of asynchronous communication. Therefore, the issue of data-transmission optimization is analyzed only from the view point of the sender. The presented solutions are directly applicable assuming an "impatient" receiver, who waits for a successive message and processes it immediately.

2. Asynchronous communication

In distributed systems, oriented toward sending messages, synchronous and asynchronous types of communication can be distinguished.

In the case of synchronous communication, after sending a message, the sender is in a standby mode of passive waiting because the receiver should complete the data receipt and the sender should possibly obtain the results of processing. Such a communication has two basic drawbacks: it blocks the sender (the sender does not process further until the receiver acknowledges the receipt of the message) and hinders using the full transmission bandwidth allocated for the connection.

Asynchronous communication is an alternative to synchronous communication (Fig.1).

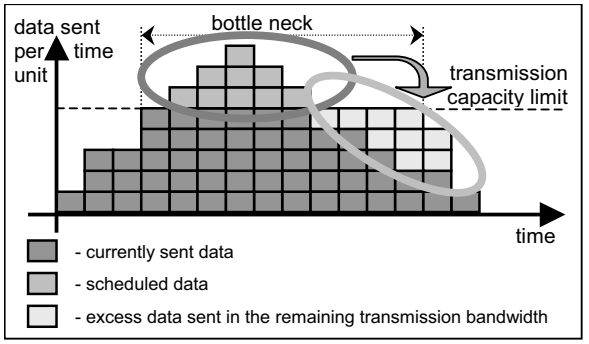

Fig. 1. Building of a transmission bottleneck and methods of problem solving.

In this model, the sender is not blocked during the process of sending messages and thus can send any number of messages without waiting for acknowledgment. Unfortunately, the message-acknowledging procedure is not simple. However, this method enables better use of the allocated transmission bandwidth, although the messages that cannot be sent at a given moment have to be queued.

A complex solution of asynchronous communication is offered by Message Queuing Systems (MQSs). These are the links between the senders and receivers of messages. Software worked out for such systems is called *Message-Oriented Middleware (MOM)*.

The main task of an MQS is delivering messages from the sender to the receiver. The message-sending process is accomplished by MOM brokers and covers the following stages:

- reception of the data to be sent by the sender as a message to the receiver,
- queuing the data of messages to be sent,
- transmission of queued messages,
- reception of data messages by the receiver and the queuing of messages,
- acknowledging the receipt and waiting for a query to receive new data.

Attention should be paid to the queuing of messages by a broker on the side of the sender. There are situations where the number of scheduled data exceeds the transmission capacity. Under these circumstances, a part of the messages is withheld by the broker (Fig.1) until the moment the data can be transmitted.

The broker joins the messages in one stream of data to completely use the allotted transmission bandwidth. Then the stream is divided by the transport layer into network packets and sent to the broker on the side of the receiver.

3. Research on asynchronous packet communication

One of the basic questions discussed regarding asynchronous communication is the scheduling of messages to be sent. The scheduling has a significant influence on the quality of the basic factors controlling the operation of the system. Message-scheduling problems are similar to task-scheduling issues. Therefore selected solutions developed for task scheduling have been applied to solving data-transmission problems. Below are discussed some interesting solution examples for scheduling algorithms proposed for message scheduling along with their computational complexity.

Original message-scheduling point-to-point algorithms are presented in (Ramanthan and Rupnick, 1991). The proposed Minimum-Cost Scheduling (MCS) is based on penalty imposed on undue deliveries. The schedule is determined for a given node with every arrival of a new message. In each such cycle, a new message is queued so that the factor value is minimized. The computational complexity of the algorithm is low and equals $O(n)$. The principle and conditions of the algorithm (system with a known due delivery time) resemble the algorithms EDF (Earliest Deadline First) and ELF (Earliest Laxity First). The experimental methods presented therein showed better results (lower cost) with the MCS algorithm compared with the EDF and FIFO models.

The scheduling of messages sent through the network having a line topology is presented in the report by Adler, *et al.* (1998). The optimization problem was analyzed for messages with a known time of generation and a due time of delivery. It is additionally assumed that exactly one packet is used for sending one message. Optimum message scheduling, with or without the sender buffer, has been proved to be an NP-hard task. Similar conclusions for a graph (tree, network) topology are presented in the article by Adler *et al.* (1999).

In the study (Lui and Zaks, 1997), the authors propose a "greedy" message-scheduling algorithm to be used for synchronous communication. The possibility of keeping the due-time limitation is analyzed for various topologies (lines, circle, and tree). The NP-completeness of the optimum-scheduling problem has been proved.

A solution of the message-scheduling problem with a due time of delivery for distributed systems with a complex structure is proposed in the paper presentation by Tsai and Shin (1996). The authors also present the experimental results of the proposed algorithm as compared to other known scheduling algorithms, such as Longest First (LF), Shortest First (SF), Farthest First (FF), Nearest First (NF), Largest Remaining Bandwidth Requirement (RBR) First (LBF), and Smallest RBR First (SBF). The latter two algorithms of LBF and SBF account for the RBR in the data-transmission system.

An interesting optimization method is proposed in the study by Dobrin and Fohler (2001). A method for grouping messages for a controller area network bus was presented, wherein the loading of the data-transmission system can be reduced.

The optimization of the network buffer emptying procedure is presented in the reports by Harchol-Balteret *et al.* (2000) as well as Bansal and Harchol-Balter (2000). An experiment was conducted in which the server control of websites was modified. In this approach, the static query was serviced through the Shortest Remaining Processing Time (SRPT). Such scheduling was proven to considerably reduce the average and variation response time of the server.

The concept of dividing a data packet is provided in the report by Zhu *et al.* (2001). The packet is a document imparted by the WWW server. Depending on needs, the document is sent in parts, as, e.g., by pages. The optimization of their scheduling depends on the tasks set before the server and network loading.

The optimization of asynchronous communication can be applicable to agent systems (Yang *et al.*, 2002) or computation environments (Kielmann *et al.*, 1999).

The above discussion shows that investigations devoted to the optimization of asynchronous communication do not follow a single pathway; they are oriented to solving various specific problems.

4. Model of asynchronous communication

The data-transmission model presented in the article considers the message queue on the side of the sender. The model will be used in further analyses for developing message-scheduling methods and algorithms on the side of the sender.

For the sake of precision, the following general assumptions on data transmission have been made:

- Some transmission bandwidth, which is guaranteed, is expressed by minimum (or constant) transmission velocity (size of data sent in a time unit).
- The most effective use of the medium lies in sending data in an uninterrupted stream of successive scheduled messages.
- Only one stream of messages can be transmitted at a time.
- When the transmitted stream can be divided into packets, the data at the receiver end can be processed only after receiving the entire packet.
- If data to be sent are available at all times and in an unlimited mode, they are formatted into packets of uniform length and sent in constant time intervals, regardless of the size of the transmitted data.
- It is assumed that, in the analyzed systems, the message-scheduling time is negligible compared with the transmission time of messages.
- The time consumed for processing the received packets (identifying the message and servicing it) is negligible compared with the transmission time. An assumption is made that the receiver is continuously waiting for new messages all the time ("impatient receiver"). In other words, no delays are assumed as caused by the reception and processing of data on the side of the receiver.

4.1. Model of a message. A message is a coherent sequence of data, having a definite value to the receiver. No interpretation of the content of the sent messages is made in the analyzed system. It is assumed that a message m is characterized by two parts as follows:

$$m = [l, w],$$

where l is the length of the message, and w is the weight (priority) of the message.

The length of a message is the number of elementary data units. In this article, the size of the message (length) is defined in bytes.

The weight is a natural number from a given interval and can be analyzed under two categories as follows:

- weight—**importance** of a given message; the message must be delivered,
- priority—**urgency** of a message when compared with other messages and connected with the priority of sending; priority is mainly applicable in real-time systems.

4.2. Message-scheduling model (sender). The arriving and unsent messages create a queue Q in the system:

$$Q = (m_1, \ldots, m_n) \ . \tag{1}$$

The index i in a message m_i denotes its order in Q. The number of messages n changes with time and depends on the system state. After the arrival of a new set of data, a new order in the sending queue Q is established.

Messages are sent according to the new order, meeting the time requirements or priority (order) of delivery. The determination of a new order in the queue is called scheduling. New scheduling of a queue Q_n is a permutation of the queue with n elements.

4.3. Linear transmission model. A linear dependence of data transmission, which can be adapted to the transmission system with a message queue, is presented in the (Coulouris *et al.*, 2001). The basic factor describing the quality of service is the completion time C_j of a queued message m_j. The completion time is a sum of the times required for the transmission of this message and the remaining preceding messages.

$$C_j = d + \left(\sum_{i=1}^{j-1} l_i + l_j \right) \frac{1}{V}, \tag{2}$$

where C_j is the completion time of a message m_j, l_j is the length of a message m_j, $\sum_{i=1}^{j-1} l_i$ is the sum of the lengths of messages preceding m_j, d stands for a delay (initial content of the adapter's buffer), V is the velocity of data transmission.

4.4. Nonlinear transmission model. In the case of data transmission by division into network packets, data transmission is not achieved on a continuous basis. The transmission time cannot be treated as linear due to a distinct division of the data-transmission stream into network packets. This phenomenon mainly manifests itself in low-capacity lines, e.g., for dial-ups, for which successive packets are received in time intervals of hundred miliseconds and the processing of the received data is ten to a hundred times faster.

For queued and scheduled messages, the time within which they will be delivered to the receiver should be determined. The nonlinear completion time C'_j of a message m_j counts from the moment it is queued until the moment the receiver obtains the message. The completion time requires a consideration of the initial filling up of the buffer q and other messages preceding the message m_j. The value C'_j is obtained by determining the number of packets into which the data stream of messages $(m_1, m_2, \ldots, m_j)$ is divided and then multiplying it by the transmission time of a single packet (2).

$$C'_j = \left\lceil \frac{q + \sum_{i=1}^{j-1} l_i + l_j}{PS} \right\rceil TTP, \tag{3}$$

where C'_j is the completion time of a message m_j (nonlinear model), $\lceil \cdot \rceil$ is the ceiling function, which rounds the argument up to the next highest integer value, l_j denotes the length of a message m_j, $\sum_{i=1}^{j-1} l_i$ is the sum of the lengths of messages preceding m_j, q stands for the initial filling of the buffer (in bytes), PS is the size of the packet, TTP is the transmission time of a single network packet.

The comparison of completion times for the linear and nonlinear (packet) transmission models presented in Figs. 2 and 3 illustrates the features of the transmission models.

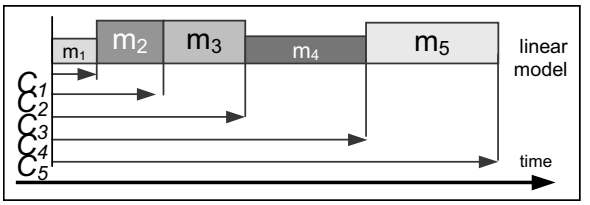

Fig. 2. Linear model of network transmission.

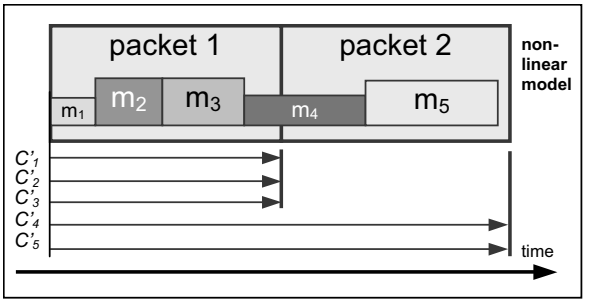

Fig. 3. Nonlinear model of network transmission.

5. Message scheduling

For packet communication, optimization criteria that are significant for defining the quality of services of such systems should be determined. The quality factor of the total completion time has been selected for analysis in this article.

Total completion time. The sum of the completion time $\sum_{j=1}^{n} C_j$ is the sum of receiver waiting times to obtain all messages C_j queued on the side of the sender. The factor is elaborated with the formula

$$\sum_{j=1}^{n} C_j = C_1 + C_2 + \cdots + C_n \,. \tag{4}$$

5.1. Minimizing the factor of the total completion time. The completion of data transmission expressed using the formula (4) is defined for a continuous transmission. The same result should be achieved based on the analogy between the problem of message scheduling and task scheduling in a multitask operating system for a defined case. For minimizing the factor $\sum C_j$, the Shortest Processing Time (SPT) rule can be applied, i.e., tasks are performed from the shortest to the longest ones. A theorem is presented below based on this rule, adapted to the needs of asynchronous communication.

Theorem 1. (SPT (Smith, 1956)) *Scheduling according to the nondecreasing length of messages minimizes the sum of completion times.*

In the packet transmission mode, a nonlinear dependence of the data-transmission time is expressed by (3). Hence, the question of whether or not the SPT rule is valid has to be resolved also for a nonlinear transmission model.

Theorem 2. (SPT for a nonlinear model) *Message scheduling according to the nondecreasing message size (SPT) when the data stream is divided into packets minimizes the sum of completion times* $\sum C'_j$.

Proof. (Indirect) For the sake of simplicity, without losing the general character, a unit time of packet transmission is assumed (i.e., $TTP = 1$). When scheduling n messages according to their nondecreasing length, the initial scheduling is assumed as in the expression below:

$$m_1, \ldots, m_{x-1}, m_x, \ldots, m_y, m_{y+1}, \ldots, m_n,$$

where $1 \le x < y \le n$. For sending the first messages m_1, the number of needed packets can be calculated by $\left\lceil l_1/PS \right\rceil$. Due to the assumption $TTP = 1$, the m_1 message completion time is given by

$$C_1 = \left\lceil \frac{l_1}{PS} \right\rceil TTP = \left\lceil \frac{l_1}{PS} \right\rceil \,.$$

Accordingly, the sum of completion times of all messages for this sequence (σ) is given by the following formula:

$$\begin{aligned}
&\sum_{j=1}^{n} C'_j \\
&= \left\lceil \frac{l_1}{PS} \right\rceil + \cdots + \left\lceil \frac{l_1 + \ldots + l_{x-1}}{PS} \right\rceil \\
&+ \left\lceil \frac{l_1 + \cdots + l_{x-1} + l_x}{PS} \right\rceil + \cdots + \left\lceil \frac{l_1 + \cdots + l_y}{PS} \right\rceil \\
&+ \left\lceil \frac{l_1 + \cdots + l_y + l_{y+1}}{PS} \right\rceil + \cdots + \left\lceil \frac{l_1 + \ldots + l_n}{PS} \right\rceil .
\end{aligned} \tag{5}$$

The analysis of other modes of scheduling σ' performed by an exchange of the order of the messages m_x and m_y is carried out as follows:

$$m_1, \ldots, m_{x-1}, m_y, \ldots, m_{y-1}, m_x, \ldots, m_n.$$

Then the sum of completion times is given by

$$\begin{aligned}
&\sum_{j=1}^{n} C'_j \\
&= \left\lceil \frac{l_1}{PS} \right\rceil + \cdots + \left\lceil \frac{l_1 + \cdots + l_{x-1}}{PS} \right\rceil \\
&+ \left\lceil \frac{l_1 + \cdots + l_{x-1} + l_y}{PS} \right\rceil + \ldots + \left\lceil \frac{l_1 + \cdots + l_y + l_x}{PS} \right\rceil \\
&+ \left\lceil \frac{l_1 + \cdots + l_y}{PS} \right\rceil + \cdots + \left\lceil \frac{l_1 + \ldots + l_n}{PS} \right\rceil .
\end{aligned} \tag{6}$$

The analysis of the sums of completion times shows that, for both scheduling cases, the C'_j values for the messages $[1, x-1]$ and $[y+1, n]$ are identical, whereas the C'_j values for $[x, y]$ for the second scheduling (σ') are equal to or greater than the corresponding values from the first scheduling. Therefore, the sum of completion times of messages with packet division according to their nondecreasing length has a minimum value. ■

This issue is also generally discussed by Janiak and Krysiak (2007).

6. Asynchronous transmission in real-time systems

The criteria for message delivery are to be determined for real-time message-transmission systems. In such systems, the coefficient w denotes the priority of the message to be sent. In real-time systems, messages with higher priority are sent before those with lower priority.

For messages waiting in the priority queue, the time D_j after which they will be delivered to the receiver, can

be determined by the following equation:

$$D_j = \left\lceil \frac{q + \sum_{i=1}^{n} l_i \cdot v_{ij}}{PS} \right\rceil TTP, \tag{7}$$

where D_j is the maximum time of the transmission delay of the message m_j, l_j is the length of the message m_j, n denotes the number of messages in the queue, v_{ij} determines whether the priority of the message m_j is greater than or equal to the priority of the message m_j,

$$v_{ij} = \begin{cases} 1 & \text{if} \quad w_i \geq w_j, \\ 0 & \text{if} \quad w_i < w_j, \end{cases}$$

q stands for preliminary filling of the buffer (in bytes), PS denotes packet size, TTP is the time of transmission of a single network packet.

When constructing real-time systems, a time analysis is performed using additionally defined properties of the modeled system. An assumption that the system generates periodic messages is usually made.

6.1. Idea of optimizing the total completion time in real-time systems with packet transmission. Suppose that there are two messages m_1 and m_2 in a system and that m_1 has higher priority, with the message length greater than the packet size, whereas message m_2 has lower priority, with a smaller message length (less than one network packet).

Example 1. Consider

$$\begin{aligned} m_1: &\quad l_1 = 4.5\ PS, \quad w_1 = 10, \\ m_2: &\quad l_2 = 0.5\ PS, \quad w_2 = 8. \end{aligned}$$

The priority-queuing algorithm selects m_1 as the first to be sent and m_2 as the second (Q):

$$Q = (m_1, m_2).$$

To send these messages, the system must send five network packets. This situation is illustrated in Fig. 4.

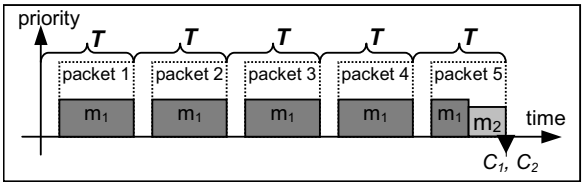

Fig. 4. Example of the priority-queuing schedule (Q).

The completion times C_1 and C_2 of the transmission of messages m_1 and m_2 is the time of receiving the fifth packet ($C_1 = C_2 = 5T$). The total completion time equals in this case $\sum C_j = 5T + 5T = 10T$.

The order can be changed, and message m_2 can be sent first and the message m_1 as the second:

$$Q' = (m_2, m_1).$$

In this case (Q'), the completion time of the transmission of message m_1 will be the same ($C_1^* = C_1 = 5T$), but the transmisson of the message m_2 will be completed when the receiver receives the first network packet, so the completion time of the message m_2 will be shorter ($C_2^* = T < C_2$). In that case, the total completion time is $\sum C_j = 5T + 1T = 6T$. This situation is illustrated in Fig. 5.

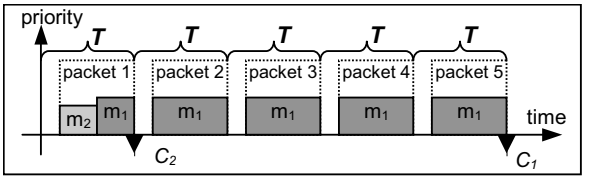

Fig. 5. Example of the proposed schedule (Q').

This example proves that the minimization of the total completion time for message-queuing systems cannot be achieved with the SPT algorithm, hence there should be a new algorithm developed for a real-time based message system. This problem, described by Graham *et al.* (1979), is known as a problem $1|sp - graph|\sum f(C_j)$, where $f(C_j)$ is a stepwise function.

6.2. Minimization of the total completion time in real-time systems with packet transmission. An algorithm is proposed herein by the authors, which minimizes the sum of completion times in real-time systems with packet transmission.

Algorithm 1

The proposed algorithm consists of two phases:

1: construction, and

2: improvement.

The first phase is devoted to sorting messages with the following sorting rule: higher priority messages first; for messages with the same priority, the shorter message first. Thus, a permissible solution is generated and deadlines are calculated by the way. This solution is optimal ($\min \sum C_j$) for the linear model of transmission.

The second phase is to improve the solution. The algorithm is similar to the bubble-sort one. Two adjacent messages m_i and m_{i+1} are swapped when m_{i+1} is shorter than m_i and the deadlines of m_i and m_{i+1} are not exceeded:

$$\begin{cases} C'_i \leq D_i, \\ C'_{i+1} \leq D_{i+1}, \end{cases} \tag{8}$$

where C'_i, C'_{i+1} denote the completion times of the same messages m_i and m_{i+1} after swapping, D_j, D_{j+1} stands for the maximum time of the transmission delay of messages m_i and m_{i+1}.

The direction of the bubbles is backward from the start of the queue.

Experimental tests with brute force algorithms demonstrated that Algorithm 1 gives an optimal solution for 100.000 sets of message queues with a maximum of fourteen messages in each set. In the case of the performed experiments, the proposed algorithm has always given the optimal value, but a formal proof of the optimality of the algorithm has not been made.

6.3. Performance of the proposed algorithm. The performance of the proposed algorithm was assessed experimentally. There were six series of tests; each one had 1000 random sets of messages.

The parameters of these series are enumerated in Table 1, where N is the number of messages, $P_{\max}$ denotes

Table 1. Parameters of the series

Series	N	$P_{\max}$	$L_{\min}$	$L_{\max}$	PS
1	10	5	4	100	50
2	10	20	4	100	50
3	10	200	4	100	50
4	10	5	4	400	50
5	10	20	4	400	50
6	10	200	4	400	50

the number of priority levels, $L_{\min}$ is the minimal length of messages, $L_{\max}$ denotes the maximal length of messages, PS stands for packet size.

Results of experiments are shown in Fig. 6. For each test series a CTR (Completion Time Ratio) was determined:

$$CTR = \frac{\text{average} \sum C_j \text{ for Algorithm 1}}{\text{average} \sum C_j \text{ for Algorithm RT}}. \qquad (9)$$

As a reference to the proposed Algorithm 1, a standard RT algorithm based on message priorities and an FIFO queue was chosen. The standard RT algorithm is based on the RMS rule (without preemption) and was published by Liu and Layland (1973).

The completion time ratio depends on message parameters. For small messages (approximately equal to the size of the packet or smaller), the completion time reaches the lowest level, about 10% less. The small diversity of priorities (e.g., five levels for ten messages) is also the cause of the lower cost of optimization.

7. Real implementations

The problems considered are initiated by real applications.

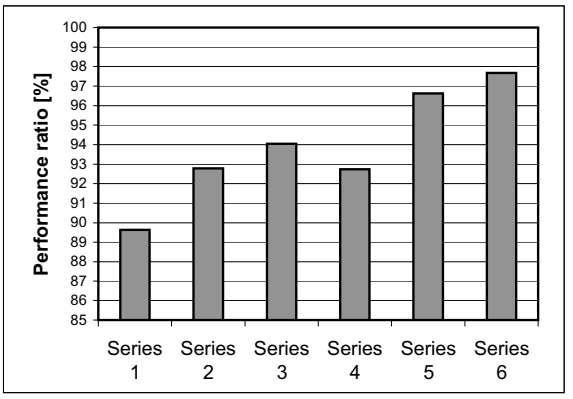

Fig. 6. CTR for a nonlinear model of network transmission.

The original target may be a sport betting system (Osman *et al.*, 2000), especially a real time sport betting subsystem, which uses wireless slow speed networks (Hamalainen *et al.*, 2003; Xu, 2008). In such a system it is important for the bookmaker to know the current state of the system. For crucial time moments, a decrease in the message-transmission completion time by a few percent may be an essential case of slow transmission networks. Another example of a slow network is that which uses a star-topology network on slow DSL connections, which has to keep the real-time constraint—the time of reaction no longer than 500 ms (Fig. 7). It is important to put bets as quickly as possible to the main server, and so that minimizing the total completion time factor is desirable.

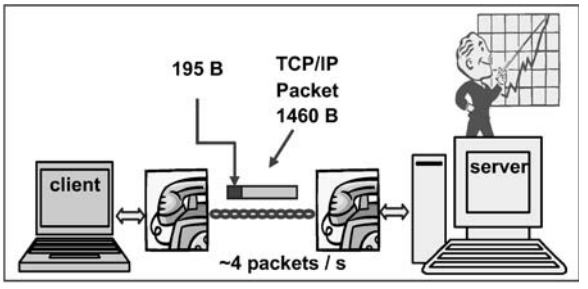

Fig. 7. Schema of the mobile betting system.

Other systems which can optimize the total completion time factor are medical monitoring applications (Gouaux and Simon-Chautemps, 2002; Khoor *et al.*, 2003). The time of delivery in some heart diagnostics should be as short as possible.

This consideration was investigated by experiments in a real environment—there were two computers connected by a serial cable (RS 232) and a direct connection was set. Special software, which was transmitting a stream of messages over TCP/IP, emulated the described system. Packet size (PS) was 1460 B, typical for TCP/IP, the initial content of the adapter's buffer, q = 1460 B.

The set of four messages was sent (length in bytes):

$$\begin{aligned} m_1 &: l_1 = 5000\text{B}(+12\text{B msg. header}),\ w_1 = 10,\\ m_2 &: l_2 = 100\text{B}(+12\text{B msg. header}),\ w_2 = 1,\\ m_3 &: l_3 = 1000\text{B}(+12\text{B msg. header}),\ w_3 = 5,\\ m_4 &: l_4 = 200\text{B}(+12\text{B msg. header}),\ w_4 = 2. \end{aligned}$$

For this example,

$$\begin{aligned} D_1 &= \lceil (1460+5012)/PS \rceil \cdot TTP = 5 \cdot TTP,\\ D_2 &= \lceil (1460+5012+1012+212+112)/PS \rceil \cdot TTP\\ &= 6 \cdot TTP,\\ D_3 &= \lceil (1460+5012+1012)/PS \rceil \cdot TTP\\ &= 6 \cdot TTP,\\ D_4 &= \lceil (1460+5012+1012+212)/PS \rceil \cdot TTP\\ &= 6 \cdot TTP. \end{aligned}$$

These messages was scheduled by a standard algorithm (RT FIFO) and the one proposed here (Algorithm 1). The times of message delivery are given in Table 2. The precision of time measurement was ± 10 ms, TTP was about 260 ms. The last packet for this set was not quite full.

Table 2. Times of scheduled messages in a real experiment.

msg. sequence	RT FIFO	Algorithm 1
1	$m_1 - 1291$ms	$m_2 - 520$ms
2	$m_3 - 1382$ms	$m_4 - 520$ms
3	$m_4 - 1382$ms	$m_1 - 1291$ms
4	$m_2 - 1382$ms	$m_3 - 1382$ms
$\sum C_j$ factor	$\sum C_j = 5437$ms	$\sum C_j = 3713$ms

The proposed algorithm decreased the total completion time factor and kept the times of delivery. This simple experiment supported the claims of this article.

8. Conclusions and future work

The minimization of the total completion time is important for making network systems more efficient and more fault-tolerant. For asynchronous transmission with packetization, it was proved that a low-complexity optimal algorithm can be applied. The proposed algorithm for packet transmission even gives a 10% smaller value of the factor considered than other standard algorithms.

Future work involves developing optimal and heuristic algorithms considering the total cost factor $\left(\sum w_j C_j\right)$.

References

Adler, M., Khanna S., Rajaraman, R., and Rosen, A., (1999). Time-constrained scheduling of weighted packets on trees and meshes, *Proceedings of the 1999 ACM Symposium on Parallel Algorithms and Architecture, New York, NY, USA*, pp. 1–12.

Adler, M., Rosenberg, A.L., Sitaraman, R., and Unger, W., (1998). Scheduling time-constrained communication in linear networks, *10-th ACM Symposiumon Parallel Algorithms and Architectures, New York, NY, USA*, pp. 269–278.

Bansal, N. and Harchol-Balter, M. (2000). Analysis of SRPT scheduling: Investigating unfairness, *Technical Report CMU-CS-00-149*, Carnegie Mellon University, Pittsburgh, PA.

Coulouris, C., Dollimore, J. and Kindberg, T. (2001). *Distributed Systems—Concepts and Design*, Addison-Wesley, Harlow.

Dobrin, R. and Fohler, G. (2001). Implementing off-line message scheduling on Controller Area Network (CAN), *8-th IEEE International Conference on Emerging Technologies & Factory Automation, Nice, France*.

Gouaux, F. and Simon-Chautemps, L. (2002). Ambient intelligence and pervasive systems for the monitoring of citizens at cardiac risk: New solutions from the EPI-MEDICS project, *IEEE Computers in Cardiology* **29**: 289–292.

Graham, R.L., Lawler, E.L., Lenstra, J.K. and Rinnooy Kan, A.H.G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey, *Annals of Operations Research* **5**(5): 287–326.

Hamalainen, P., Hannikainen, M., Hamalainen, T.D. and Soininen, R. (2003). Offline architecture for real-time betting, *ICME'03. Proceedings of the 2003 International Conference on Multimedia and Expo, Baltimore, MD, USA,* Vol. I, pp. 709–712.

Harchol-Balter, M., Bansal, N. and Schroeder, B. (2000). Implementation of SRPT scheduling in web servers, *Technical Report CMU-CS-00-170*, Carnegie Mellon School of Computer Science, Pittsburgh, PA.

Janiak, A. and Krysiak, T. (2007). Single processor scheduling with job values depending on their completion times, *Journal of Scheduling* **10**(2): 129–138.

Khoor, S., Nieberl, J., Fugedi, K. and Kail, E. (2003). Internet-based, GPRS, long-term ECG monitoring and non-linear heart-rate analysis for cardiovascular telemedicine management, *Computers in Cardiology* **30**: 209–212.

Kielmann, T., Hofman, R.F.H., Bal, H.E., Plaat, A. and Bhoedjang, R.A.F. (1999). MPI's reduction operations in clustered wide area systems, *Proceedings of MPIDC '99, Message Passing Interface Developer's and User's Conference, Atlanta, GA, USA*, pp. 43–52.

Liu, C.L., and Layland, J.W. (1973). Scheduling algorithms for multiprogramming in a hard real-time environment, *Journal of the ACM* **20**(1): 46–61.

Lui, K.S. and Zaks, S.(1997). Scheduling in synchronous networks and the greedy algorithm, *Proceedings of the 11-th International Workshop on Distributed Algorithms (WDAG), Saarbrucken, Germany*, Vol. 2, pp. 556–561.

Osman, M.Y.B., Kin, P.F.W., Leong, L.L.E. , Wong, Ch.V., Wong, K.N., Khoon, J.T.L., Goh, S.B., Zainol, M.N.B. and Prakash, E.C. (2000). Simple pools online betting software system—A UML use case analysis, *Proceedings of TENCON2000, Kuala Lumpur, Malaysia*, Vol. 2, pp. 556–561.

Ramanathan, P. and Rupnick, G.M. (1991). Deadline constrained message scheduling in point-to-point interconnection, *Proceedings of the System Design Synthesis Technology Workshop, Silver Spring, MD, USA*, pp. 183–192.

Smith, W.E. (1956). Various optimizers for single-stage production, *Naval Research Logistics* (2): 59–66.

Tsai, B.R. and Shin, K.G. (1996). Combined routing and scheduling of concurrent communication traffic in hypercube multicomputers, *Proceedings of the International Conference on Distributed Computing Systems, Hong Kong*, pp. 150–157.

Yang, B., Dayou L. and Kun Y. (2002). Communication performance optimization for mobile agent system, *Proceedings of the IEEE First International Conference on Machine Learning and Cybernetics (ICMLC2002), Beijing, China*, pp. 327–335.

Xu, X. (2008). The videotex hot line lottery-ticket-buying solution based on mobile GPRS system, *International Conference on Management of e-Commerce and e-Government, ICMECG'08, Washington, DC, USA*, pp. 30–35.

Zhu, X., Yu J. and Doyle J.(2001). Heavy tails, generalized coding and optimal web layout, *Proceedings of IEEE INFOCOM, Piscataway, ND, USA*, Vol. 3, pp. 1617–1626.

Adam Piórkowski received a Ph.D. degree in computer science in 2005 from the AGH University of Science and Technology. Currently he works for the Department of Geoinformatics and Applied Computer Science at the Faculty of Geology, Geophysics and Environmental Protection. His research interests include real-time systems, scheduling, data bases, component technologies and medical imaging.

Jan Werewka received an M.Sc. degree in electronic data processing from the Technical University of Dresden, Germany. He obtained a Ph.D. degree (with honors) in 1971 in the field of automatic control at the Electrical Engineering Department of the AGH University of Mining and Metallurgy in Cracow, Poland. In 1989 he was granted the D.Sc. degree (habilitation) in computer science from the Humboldt University of Berlin at the Faculty of Mathematics and Natural Sciences. Currently, he works at the Department of Automatics at the Computer Science Laboratory of the Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, as a professor of the AGH University of Science and Technology. His main research interests include such areas as system modeling, distributed real-time computer systems, IT project management. He is the author of over 100 scientific publications and books related to his research interests. He is a member of the IEEE and PMI and has vast industrial experience concerning the managing of software production and IT projects.

Received: 11 December 2008
Revised: 12 June 2009