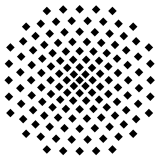
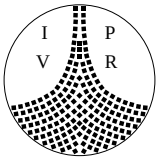


Universität Stuttgart
Fakultät Informatik



Foundations for Mapping of Distributed Multimedia Applications to Distributed Computer System

Alexander Hagin

CR-Klassifikation: C.2.4, C.4, G.1.6, G2.2, I.6

Foundations for Mapping of Distributed Multimedia Applications to Distributed Computer System

Alexander Hagin

Fakultätsbericht 13/1994
Technical Report
October 1994

Fakultät Informatik
Institut für Parallele und
Verteilte Höchstleistungsrechner
Universität Stuttgart
Breitwiesenstraße 20 - 22
D-70565 Stuttgart

Abstract

Distributed multimedia applications need convenient framework for their realization in a distributed computer system. This paper deals with the problem of mapping distributed multimedia applications to a distributed computer system. An approach for formalization of this problem as well as a set of abstractions, mathematical programming formulations, models and solution procedures are proposed.

Distributed multimedia application is represented by one or a set of data flow graphs that specify quality of service and resource requirements of the application. Distributed computer system is represented as a relevant directed weighted graph with information about available computer and communication resources. A problem of best placement (from the point of preliminary defined cost criteria) of data flow graphs onto distributed computer system is decided.

CONTENTS

1. INTRODUCTION

Recent technological developments in high speed networks and multimedia workstations are making possible entirely new classes of distributed application such as distance learning, desktop videoconferencing, remote multimedia database access and so on. Multimedia systems combine a variety of information sources, such as voice, graphics, animation, images, audio, and full-motion video, into a wide range of applications. Research and development efforts in multimedia computing can be divided into two groups. One group centers its efforts on the stand-alone multimedia workstation and associated software systems and tools. The other combines multimedia computing with distributed systems. Potential new applications based on distributed multimedia systems [Furht, 94].

Works on distributed multimedia systems are based more or less on intuitive feelings about the required capabilities. Because requirements are mostly srecified in an informal and imprecise way, they are open to different interpretations, which can cause disagreement about the design choices. Often, disagreements detected only after progress has been made on the design of protocols or protocol entities.

Distributed MultiMedia Application (DMMApplication) can be represented by one or a set of **Data Flow Graph**s (DFGraph) [Rothermel, 94]. In a DFGraph the nodes represent so-called **component**s that are interconnected by **edge**s representing **stream**s. Each component is associated at least with one device that produces (**source** component), processes (intermediate component - **filter** or **mixer**) or consumes (**sink** component) data streams. The DFGraph includes weights per components or nodes (such as type of component, CPU service time requirement, memory requirement and so on) and per streams or edges (message length, stream rate, bandwidth requirement and so on). Media streams may originate at multiple sources, traverse a number of intermediate components and end at multiple sinks.

A **Distributed Computer System** (DCSystem) considered consists of computers communicating with each other by local or/and global network(s). For every DMMApplication a relevant part of DCSystem each computer that can be used for execution of functions of one or more components always will be considered.

Our interest in this paper is on the system support aspects.The main purpose consists of investigation of the problem of DMMApplication mapping to DCSystem that roughly is formulated as following one:

Given Data Flow Graph(s) of a Distributed MultiMedia Application and Distributed Computer System with information about available resources does there exist a mapping function such that calculates:

- the best (from the point of preliminary defined cost criteria) placement of the DFGraph onto DCSystem,
- the multipoint routes in DCSystem realizing connections between computers to which sources(s), intermediate and sink(s) components are mapped,
- the needed computer and communication resources (CPU service rate, memory size, buffer size, bandwidth),

- computer schedules for placed components,
- values of Quality of Service (QoS) such that QoS requirements of DMMApplication are satisfied?

Further we will :

- define a row of notions needed widely used in multimedia subject literature but that are often different treated,
- specify a DFGraph topologies and stream parameters for DFGraph [Section 2],
- define QoS parameters of DMMApplication and consider QoS negotiation problem [Section 3.1, 3.2],
- propose the method of resource capacity requirement assignment for components and edges of DFGraph that allows to specify and negotiate resource requirements of DFGraph elements [Section 3.3],
- specify and accurately formulate the mapping problem, propose hierarchical system of methods and models for its decision, plan possible approachs for mapping decision [Section 4],
- propose some models for performance and reliability evaluations of DCSystem, its subsystem and elements and mechanisms supporting DMMApplication [Section 5],
- consider an example of mapping problem decision [Section 6],
- summarize the paper and suggest future work.

Note, due to time limitations of this work (two weeks) the author couldn't decide all menthioned problems but he retains its in contents to represent main tasks associated with these problems and hopes to decide its nearest future.

2. DATA FLOW GRAPH OF DMMApplications

2.1. DFGraph topologies

An DMMApplication is established for purpose of computing and transferring data between components. Many DFGraph topologies of DMMApplications may be defined. It is necessary (particular for mapping problem formulation too) to clearly specify possible types of DFGraph topologies and define for their needed QoS requirements. The proposed classification of DFGraph topologies is based on concepts [Rothermel,94], [Maisonniaux,94].

2.1.1. Basic components

Let's denote graphically a component of DFGraph by a rectangle and component input and output ports by small circles (see Figure 2.1).

There are following basic components (see Figure 2.1):

- The One-point Source (or simply Source) used for producing and sending data to at least one component.
- The Multisource used for producing and sending different data to at least two other components.
- The One-point Sink (or simply Sink) used for receiving data from only one other component.
- The Mixer-Sink used for receiving data that are mixed from at least two other components.
- The Filter used for any intermediate process of transmitted data or exchanging of its transfer pameters.
- The Mixer used for receiving data that are mixed from at least two other components and for following sending the same data through its one output port.
- The Mixer-multioutput used for receiving different data that are mixed from at least two other components and for following sending the different mixed data through its some output ports to at least two other components. For n input ports of Mixer-multioutput there are $2^n - 1$ possible combinations of mixed data and the same number of output ports.
-

2.1.2. Basic schemes

There are following basic schemes (see Figure 2.2):

- Point-to-point scheme,
- Multicast scheme,
- Collector scheme,
- Emission scheme.

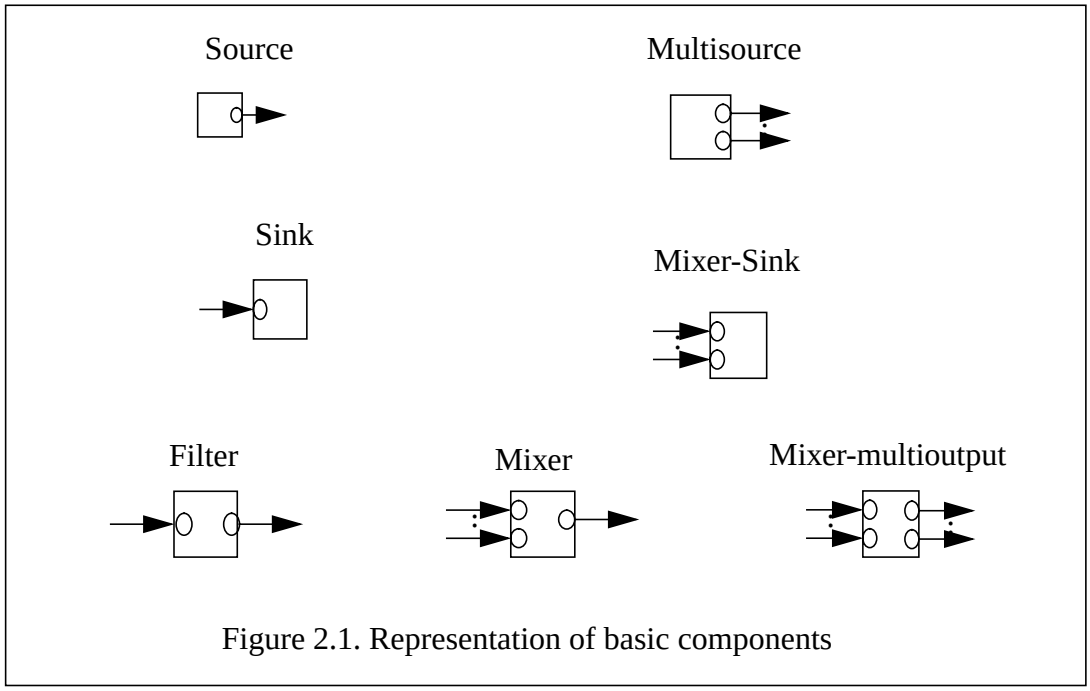


Figure 2.1. Representation of basic components

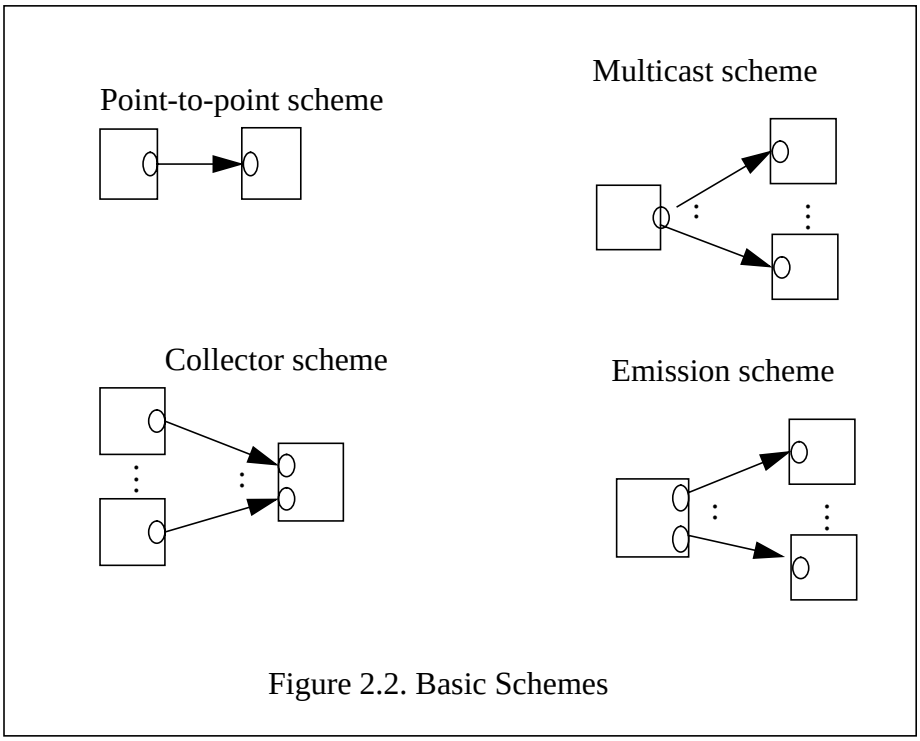


Figure 2.2. Basic Schemes

The first basic scheme stands for the point-to-point data exchanges. It associates a unique Source to its Sink. The three other schemes are dedicated to multipoint data exchanges. The Multicast scheme associates a unique Source to its Sinks. The Collector scheme associates a

unique Mixer-Sink to its Sources. The Emission sceme associates a unique Multisource to its Sinks.

2.1.3. Centralised and decentralised topologies

Proposed basic components and schemes allow to construct different DFGraph topologies.

2.1.3.1. Centralised topologies

In the Centralised Topology (see Figure 2.3) there is only a unique central component. The other components can only play a point-to-point role according to the central component. There are following types of Central Topology:

- Multicast Topology characterises a communication made of only one Multicast sheme where the central component is Source.
- Collector Topology characterises a communication made of only one Collector scheme where the central component is Mixer-Sink.
- Emission Topology characterises a communication made of only one Multicust scheme where the central component is Multisource with multiple output ports one for each Sink.
- Collector-Multicast Topology characterises a communication made of one Multicast Scheme and one Collector scheme where the central component is Mixer .
- Collector-Emission Topology characterises a communication made of one Multicast Scheme and one Collector scheme where the central component is Mixer-multioutput.

2.1.3.2. Decentralised topologies

The Decentralised Topologies allow many Multicast and/or Collector and/or Emission and/or Point-to-point schemes with distinct central components or without their (see Figure 2.4). In Decentralised Topology there is not a central component which is connected with all other that play only point-to-point role according to the central component.

In Figure 2.4 the topology b) is a hierarchical one with two level of mixing

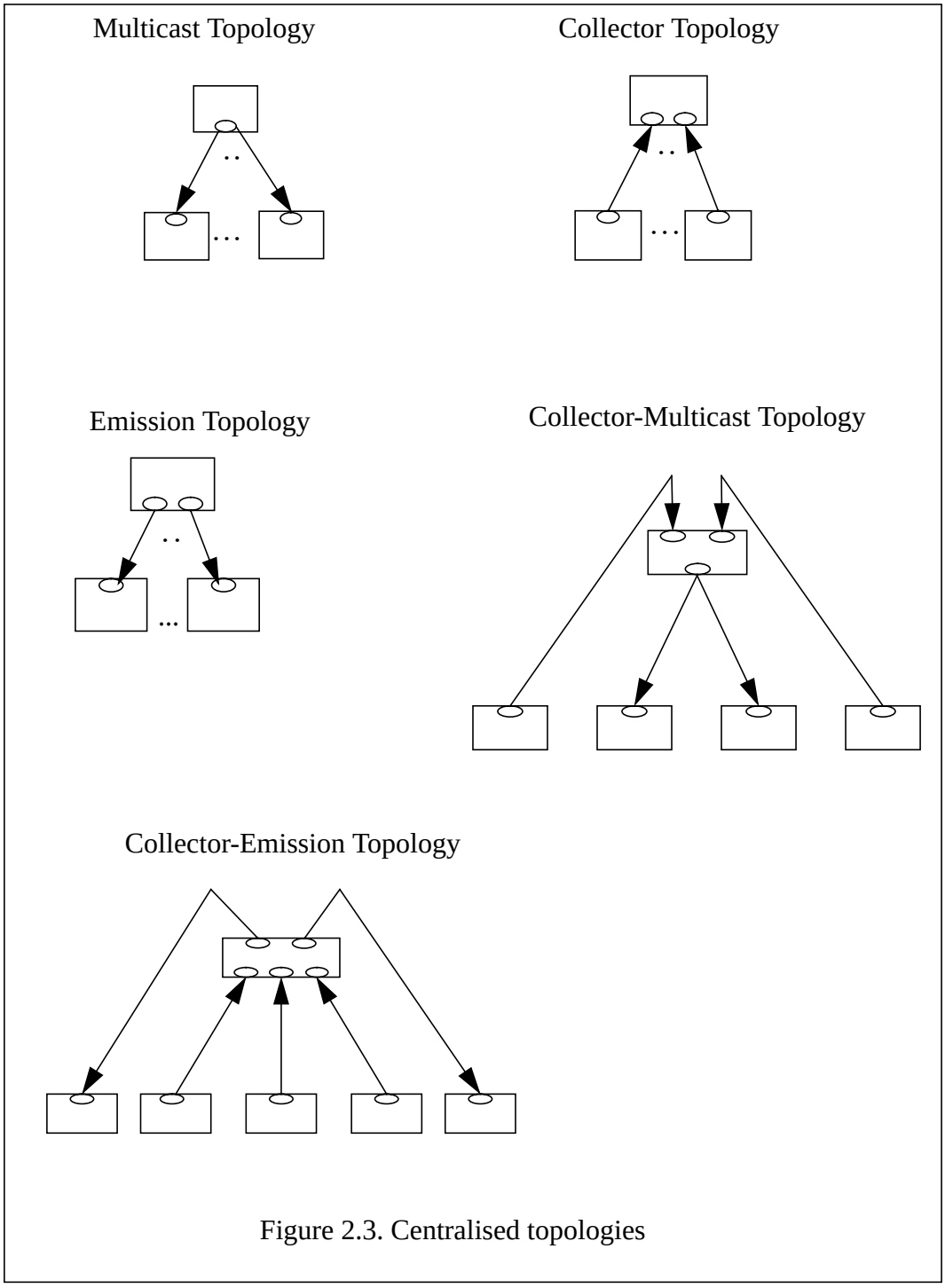
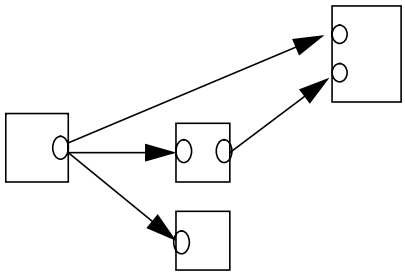
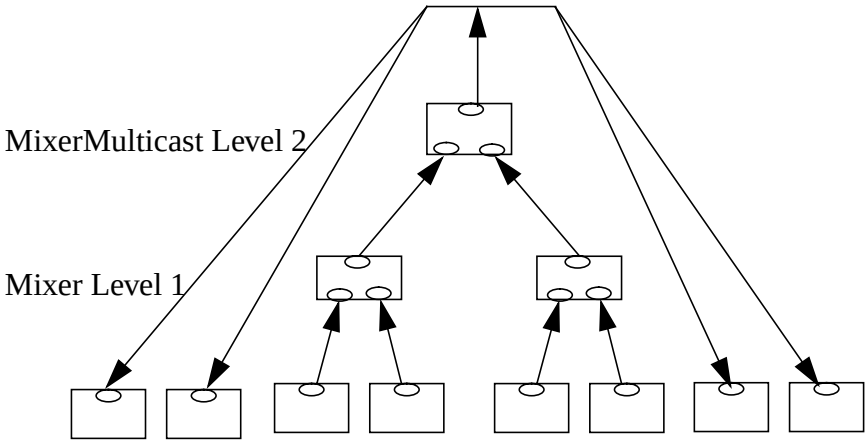


Figure 2.3. Centralised topologies



a)

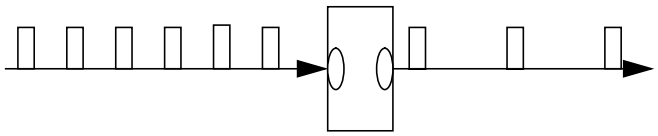


b)

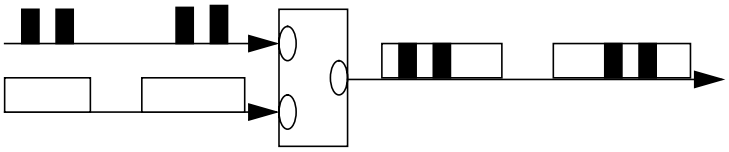
Figure 2.4. Decentralised topologies

2.2. DFGraph stream parameters

Filters and mixers, processing messages of input streams (composing/decomposing, compressing/decompressing), may change stream parameters such that stream rate and message length. Due to filters and mixers DFGraph can not be consider as a traditional communication network. It belongs a network with message absorptions and multiplications. Two examples of this phenomena are presented in Figure 2.5.



a)



b)

Figure 2.5. Stream modification
a) by filter (as a result of message compressing),
b) by mixer (as a result of message composing)

To specify DFGraph we consider the task of computing stream parameters such as stream rate λ (message/sec) and message length V (bit/sec). Let's consider Mixer-multipoint component with n input ports and m output ports (see Figure 2.6). Let a_{ij} be the rate modification factor for stream directed from input port i to output port j, $a_{ij} \geq 0$. Thus matrix $(a_{ij}, i = \overline{1, n}, j = \overline{1, m})$ determines all possible rate modification of streams transfered through mixer.

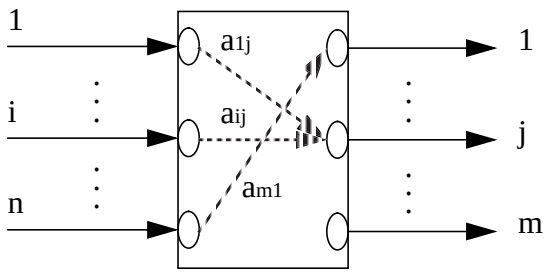


Figure 2.6. Structure of Mixer-multipoint component

For every output port j we can define the summary rate of output stream

$$\lambda_j^{out} = \sum_{i=1}^{n} \lambda_i^{in} a_{ij},\ j = \overline{1, m} \qquad (2.1)$$

where λ_i^{n}, λ_j^{out} are stream rates for input port i and output port j correspondingly.

The summary rate of component input stream

$$\lambda^{in} = \sum_{i=1}^{n} \lambda_i^{in} \qquad (2.2)$$

and the summary rate of component output stream

$$\lambda^{out} = \sum_{j=1}^{m} \lambda_j^{out} \qquad (2.3)$$

For a source $\lambda^{n} = 0$ and $\lambda^{out} > 0$. For a sink $\lambda^{n} > 0$ and $\lambda^{out} = 0$. For filter n = m = 1, $\lambda^{out} = a\lambda^{n}$.

Similar, denote b_{ij} the message length modification factor for stream directed from input port i to output port j, $b_{ij} \geq 0$. Thus matrix $(b_{ij}, i = \overline{1, n}, j = \overline{1, m})$ determine all possible message length modification of streams transferred through component.

If message from input port k absorbs exactly one message from other input ports (associated with output port j) and the summary message is directed to output port j then stream parameters of output port j are determined as

$$V_j^{out} = \sum_{i=1}^{n} V_i^{in} b_{ij} \qquad (2.4)$$

$$\lambda_j^{out} = \lambda_k^{in}, a_{kj} = 1, \forall i \therefore a_{ij} = 0 \qquad (2.5)$$

For every scheme of message mixing we have to determine similar equations.

Now consider the stream parameter calculation for DFGraph.

Assumption. DFGraph is a directed graph without cycles.

Let's number components and edges of DFGraph so that $i = \overline{1, N_C}$ are numbers of all N_C components and $j = \overline{N_C + 1, N_C + N_E}$ are numbers of all N_E edges.

Let's consider any component that one of its input ports is connected with edge i and one of its output ports is connected with edge j. Then we can interpret a_{ij} as the rate modification factor of the stream from edge i to edge j by the component. So determine factors a_{ij} over the set of edges $j = \overline{N_C + 1, N_C + N_E}$. We demand that $a_{ij} = 0$, if, in particular, edges i and j are not connected with each other immediately, that is edges i and j are not a pair such that edge i is the input one for a component and edge j is the output one for the same component.

Using factors a_{ij} write the stream rate conservation law for every edges:

$$\lambda_j = \sum_l \lambda_l a_{lj}, j = \overline{N_C + 1, N_C + N_E} \qquad (2.6)$$

Assume that stream rates λ_j^0 for output edges of source-components are given. Then a system (2.6) is heterogeneous one of M linear equations with M variables such that $M = N_E - N_E^0$, where N_E^0 is a number of source edges with given stream rates λ_j^0. Thus we get from equations (2.6) **all stream rates in DFGraph edges**.

Now given stream rates in edges **the summary rate of input stream for every component** can be calculated using equation (2.2).

Analogously we can determine the message length of a stream for every elements of DFGraph.

Thus the stream parameters can be calculated for every element (component and edge) of DFGraph. These parameters characterize loads (communication traffic, processing loads) cre-

ated by every elements of DFGraph. The load requirements of every DFGraph elements are used further for computing needed resource requirements of every DFGraph elements (see, Section 3.3).

3. QUALITY OF SERVICE REQUIREMENTS OF DMMApplications

3.1. Problem of QoS negotiation

A DMMApplication defines the set of QoS requirements that must be satisfied to allow the realization of the DMMApplication and can be possibly re-negotiated during the DMMApplication lifetime. Negotiation of QoS requirements includes:

- negotiation between QoS requirements and component parameters of DFGraph (in particular, for excluding component parameter conflicts in DFGraph - for example, between picture sizes),
- negotiation between different kinds of streams. For example, to support video connections high throughput is required and therefore high bandwidth guarantees will have to be done. Audio, on the other hand, will not require such a high bandwidth,
- mapping QoS requirements of DMMApplication down onto the communication subsystem, that is, high level requirements must be mapped onto transport level,
- partition of DMMA QoS requirements among components and edges of DFGraph as well as among computer and communication resources along transport paths in DCSystem and network(s).

DMMApplications that can accurately forecast their requirements may prefer to reserve processing, transport and network resources in advance via a connection oriented service. Also transactions that require fast response often cannot wait for the connection time associated with resource allocation which is greater than twice the round trip delay (for all protocols requiring explicit connection set-up). DMMApplications that require any guaranteed QoS higher than best-effort must request that resources be allocated to their connection in the network, the transport and application interfaces.

Thus guaranteed QoS demands the integration of a range of QoS configurable protocols and mechanisms in both the hosts (computers of DCSystem) and network [Campbell,94]. In hosts, these include thread or process scheduling, buffer allocation, jitter correction and coordination over multiple related connections. In communication systems, protocol support such as end-to-end QoS negotiation, renegotiation and indication of QoS degradation are required. Suitable resource reservation protocol and service disciplines in switch queues are needed. It is also necessary to provide maintenance and management of QoS over all system layers. This includes management functions such as admission control for new connections and monitoring to ensure that QoS levels are being maintained by the service provider.

The development of DCSystems to support DMMApplications which exploit continuous media introduces new syncronization requirements. In particular, the upper layers require support for the following two varieties of syncronization [Leopold,92]:

- event-driven synchronization: This is the act of notifying that a relevant event or set of events has taken place, and then causing an associated real-time action or actions. For example, a user clicking on the stop button relating to a video play-out could cause the play-out to stop instantaneously.

- continuous synchronization: This is where two or more continuous streams need to be kept in step by an on going commitment to a repetitive pattern of event driven synchronization relationships. An example scenario where it is required to form a continuous synchronizatio relationships is a language laboratory where separate audio tracks in different languages are stored on a single server but are to be distributed to different workstations in a real-time interactive language lesson.

Some authors refer to these sorts of linkage as orchestration [Leopold,92], [Woo,94]. They employ the term “orcherstration” rather than “synchronization” for two reasons. Firstly, cross-stream relationships may encompass more than just temporal co-ordination, and secondly the term synchronization is already overloaded and given a different emphasis in current OSI usage (the OSI concepts pertains to checkpointing and synchronization between peer entities).

Most of DMMApplications use some pre-orchestrated, stored multimedia data, precipitating the necessity for management of heterogeneous data with vasty different storage, communication, and presentation requirements. Especially with respect to communication, this data posses drastically different performance and reliability characteristics. For example, audio and video data are isochronous in nature, requiring a service supporting real-time delivery and fine-grain synchronization. On the other hand, traditional data such as text or graphics posses less stringent end-to-end delivery requirements yet needs superior reliability in transmission. However, if this data has sythetic timing relationships with other data, as is a case of pre-orchestrated multimedia information, it should be delivered on time without error.

Diverse requirements of DMMApplications can be specified through an extended set of QoS parameters, which includes:

- throughput (minimum, maximum, average),
- end-to-end delay (maximum, average),
- jitter (maximum),
- error (loss) rate (maximum).

The values of these parameters are negotiated at the time of connection establishment and are guaranteed during the data transfer phase. An efficient way for supporting desired QoS is to provide a set of application and transport protocols which can be applied for multiple virtual circuits over the communication network. Multiple virtual circuits are used for transferring different media type data on separate vurtual channels, thus utilizing different QoS parameters for each channel. Guaranteeing QoS for a particular media using an independent virtual channels is viable in a broadband network, e.g., the ATM technology.

Note, that the task of QoS requirement determination for DMMApplication is not simple. At first, all QoS parameters mentioned above are interdependent. For example, if message delivered to sink with delay more than admissible maximum delay is discarded then the maximum message delay Tmax must be negotiate with maximum loss rate (probability) Rloss such that

$$P(t > Tmax) < Rloss ,$$

where P(t > Tmax) is the probability that the message delay t is more than Tmax.

At second, components and edges of DFGraph of DMMApplication must be specified by resource capacity requirements that meet QoS requirements of DMMApplication. It is a task of DMMA QoS requirement partition among components and edges of DFGraph.

At third, it's needed to determine for which objects of DFGraph QoS requirements must be given? For each separate element (component and edge) of DFGraph, for each path between sources and sinks or for any other sets of DFGraph elements? In the first case it's very difficult to choose negotiated QoS requirements for all elements. In the second case it's impossible for path consisting , for example, of mixers to meet different requirements of different kinds of streams along the path from the source to the sink. Let's look the Figure 6.2. From component C1 to component C5 there is an audio stream and further from C5 to C7 there is mixed one combining video and audio streams. It's known that QoS requirements for video and audio streams are strong different [Furth,94].

We offer to use the termin "**media path**" (or simply **path**) for the sequence of adjacent elements in DFGraph for which we can specify particular QoS requirements. A media path is the sequence of adjacent DFGraph elements through which one or some kinds of media streams with the same or close parameters and QoS requirements are transmitted. In particular, a path can coincide with a path from source to sink and, on other hand, a path can consist of only two elements - a component with its output edge. A degree of DFGraph decomposition into paths can also depend on the necessary precision of calculations.

We name the first element in a path as **sender** and the last one as **receiver**.

Further we assume that paths and corresponding requirements for every path are given. And we demand that **each element of DFGraph must be used at least in one path.** The last claim is needed if we want to determine resource requirements for every element of DFGraph. On other hand, we don't need all possible paths in DFGraph. The number of DFGraph paths have to be enough for DMMApplication QoS requiremen assignment for every kind of streams as well as for computing a resource requirements of every DFGraph elements. In particular, a number of paths may coincide with number of stream types in DFGraph.

Thus **decision of DFGraph mapping to DCSystem includes:**

1. Specification and negotiation of QoS requirements of DMMApplication for all DFGraph media paths,
2. Specification and negotiation of resource requirements of DFGraph components and edges,
3. Itself mapping of DFGraph to DCSystem,
4. Relaxation of resources requested in plenty.

Further we follow these steps of mapping problem decision.

3.2. Relationships for QoS requirement negotiation

Let's denote QoS requirements of DFGraph path m=1,...,M :

$T_0^{(m)}, T_{min}^{(m)}, T_{max}^{(m)}$ - average, minimum and maximum message delays,

$H_0^{(m)}, H_{min}^{(m)}, H_{max}^{(m)}$ - average, minimum and maximum throughputs,

$R^{(m)}$ - maximum message loss rate (probability).

Let T_i be the average message delay in the element i (component or edge) of DFGraph; C_i the capacity needed by element i of DFGraph (further we say simple 'the capacity of element i'); π_m the m-th path of the m-th path in DFGraph.

Assume that a message can not be multiplied along a parth from the sender to the receiver (but can be combined with other messages in a mixer). Then the average message delay for m-th path

$$T^{(m)} = \sum_{i \in \pi_m} T_i \le T_0^{(m)} \qquad (3.1)$$

The minimum message delay for m-th path we get if buffer sizes in intermediate elements of the path are chose such that $T_{min}^{(m)}$ is reached on condition that the loss rate caused by buffer overflow is less than $R^{(m)}$. This is an optimization problem of buffer size choosing:

$$T_{min}^{(m)} = \left(\sum_{i \in \pi_m} T_i(L_i) \to min \right) \qquad \prod_{i \in \pi_m} R_i(L_i) \le R^{(m)} \qquad (3.2)$$

where $R_i(L_i)$ is probability that message is lost (due to buffer overflow) in the i-th element that uses buffer size L_i.

Assume that a message delivered to the receiver with delay more than T_{max} doesn't play out and we can consider such the message as a lost one. Then it is needed

$$P(t_m > T_{max}^{(m)}) < R^m, m = \overline{1, M} \qquad (3.3)$$

To determine throughput let's consider a DFGraph as a communication network and take into account that more than one message is allowed to be on transit through the network at the same time. This message multiplexing is possible due to pipelining along a given path and to alternate routing along many paths. If a message can not be multiplied or lost along a parth we can calculate the average m-th path throughput associated with sink-side using the relation [Kleinrock, 76]

$$H^{(m)} = \frac{n^{(m)} V^{(m)}}{T^{(m)}} \qquad (3.4)$$

where $n^{(m)}$ is the average number of messages being in the network path m, $V^{(m)}$ - length of message (bits), arrived to the receiver of m-th communicating pair. (Note that $n^{(m)}$ and $T^{(m)}$ are interdependent parameters).

On other hand, if messages can not be multiplied and lost along the parth between sender and receiver then the message arrival rate $\lambda^{(m)}$ from source into m-th path and the message departure rate $H^{(m)}/V^{(m)}$ must be equal.

Now consider when message may be lost into a path (caused, for example, by buffer overflow or by transmit error) with rate $R^{(m)}$. Let $\lambda^{(m)}, \lambda^{(m)}_{min}, \lambda^{(m)}_{max}$ be the average, minimum and maximum message arrival rates from source into the path of m-th communicating pair. Then:

$$\begin{aligned} H^{(m)} &= \lambda^{(m)}(1 - R^{(m)})V^{(m)} \\ H^{(m)}_{min} &= \lambda^{(m)}_{min}(1 - R^{(m)})V^{(m)} \\ H^{(m)}_{max} &= \lambda^{(m)}_{max}(1 - R^{(m)})V^{(m)} \end{aligned} \qquad (3.5)$$

The last relations are right on condition that there is no a bottleneck along the m-th parth. Otherwise when there is at least one resource i (node or link) with capacity $C_i \le \lambda^{(m)}$ and resource i is used in m-th path, $i \in \pi_m$, then $H^{(m)}_{max} = C_i(1 - R^{(m)})$ or for more general case

$$\forall (i \in \pi_m) \therefore (C_i \le \lambda^{(m)}) \Rightarrow H^{(m)}_{max} = (1 - R^{(m)}) min|_i \{C_i\} \qquad (3.6)$$

From (3.6) we get the necessary condition to meet delay and throughput requirements of DMMApplication

$$C_i \ge H^{(m)}_{max}/(1 - R^{(m)}),\ i = \overline{1, N}, \forall m/(i \in \pi_m)$$

or else

$$C_i \ge max|_{m/(i \in \pi_m)}(H^{(m)}_{max}/(1 - R^{(m)})),\ i = \overline{1, N} \qquad (3.7)$$

where N is the overall number of DFGraph elements (components and edges).

Thus QoS requirements of DMMApplication must be negotiated with each other so that relations (3.1) - (3.7) are right. (Note, further we will add the expression for jitter to these relations).

Supporting new kinds of properties and models of data streams in DFGraph we may get new relations between QoS requirements. But each time QoS requirement negotiation is needed.

3.3. Assignment of resource capacity requirements

There is the following problem of assignment of resource capacity requirements to the components and edges of DFGraph.

GIVEN:

- $(\lambda_i, i = \overline{1, N})$ - the average rates of message streams (mes/sec) through every DFG element (components and edges). Here N is the overall number of DFGraph elements. Note, if arrival stream rates are given for every source then we can calculate stream rates for every elements of DFGraph for given modification factors (see, Section 2.2);
- V_i - the message length (bits) for every edge i and number of computer operations needed for message processing (op/sec) for every component i. (Further we name V_i simply message length);
- cost function S as a sum of all resource costs $S = \sum_{i=1}^{N} S_i$, where S_i is the cost of resources needed for i-th element of DFG;
- negotiated QoS requirements of DMMApplication for every DFGraph path $T_0^{(m)}, T_{max}^{(m)}, R^{(m)}, m = \overline{1, M}$.

IT IS NEEDED

to select the resource capacity requirements C_i (op/sec - rapidity) for every component i and C_j (bit/sec - bandwidth) for every edge j of DFGraph so that

$$S = \sum_{i=1}^{N} S_i(C_i) \to min \qquad (3.8)$$

under constraints for every corresponding pair path

$$T^{(m)} = \sum_{i \in \pi_m} T_i(C_i) \le T_0^{(m)}, m = \overline{1, M} \qquad (3.11)$$

$$P(t_m > T_{max}^{(m)}) < R^m, m = \overline{1, M} \qquad ,(3.12)$$

where t_m is (random) message delay in the m-th path.

Note, that computing Ci is not a trivial task when DFGraph elements (for example a component) are realized in DCSystem by a resource with a queue (a single CPU computer) shared between messages of the same and different streams.

Now we determine the cost function $S_i(C_i)$ and message delay $T_i(C_i)$ for DFGraph element i = 1,...,N. Note that we may be satisfied by a pessimistic engineering decision of this problem because the decision can be improved on the relaxing stage of reservation protocol.

We consider the case of linear capacity costs, namely $S_i(C_i) = \alpha_i C_i$. Here α_i is the cost incurred for each unit of capacity needed for the function realization of ith element of DFGraph. In fact we can just as easily handle the case of a constant plus linear cost, namely, $\alpha_i C_i + \alpha_0$, where α_0 is an extra constant cost. In this case we get the same decision of the optimization problem. This remark is right too for $S_i(C_i) = \alpha_i C_i T$, where T is the duration of DMMApplication.

Note that the cost rate α_i may vary in an arbitrary way with respect to any parameter of the computer and channel except that it must be linear with capacity; for example, for channel α_i is often taken to be proportional to the physical length of the channel (specially, the distance between its two end points).

Now we are faced with solving for a message's average delay in a single component (placed on a computer) and edge (placed on a channel if adjacent components are mapped to different computers). First we will consider two types of models for element performance evaluation:

- multichannel model without queue allows to consider a communication and processing resources that support a simultaneously transmission or processing more than one messages (for example, X.25 public network with datagram mode);
- model of server with unlimited buffer that can be used for modelling message processing by computer or for modelling message transmission through network with unique channel (for example, LAN Ethernet). Assumption of unlimited buffer allows to get maximum of average delay that is pessimistic value which can be further improved by limitation of buffer capacity.

For first model $T_i = V_i / C_i$.

For second model we will use the M/M/1 system with Poisson arrivals at a rate λ_i and exponential service time of mean V_i / C_i . As for assumption of Poisson arrivals of messages in network we refer to [Kleinrock,76]. We assume that a time of message's service has exponential distribution because

- if even the length of message is constant (not random variable) for each element of DFGraph the assumption of exponential distributed length gives a pessimistic performance parameters that is admissible for engineering design;
- this assumption allows us to get complete expressions for problem decision.

Thus for second model

$$T_i = \frac{V_i}{C_i - \lambda_i V_i}$$

Considering DFGraph as an exponential queueing network we can determine probability $P(t_m > T_{max}^{(m)})$ for the m-th path using convolution integral of L_m-th order, where L_m is the number of DFGraph elements in the m-th path. But this complete expression is very complex for further calculation and therefore we assume that the message delay t_m has exponential distribution with mean $T^{(m)}$, that for practice is often admissible. Then

$$P(t_m > T_{max}^{(m)}) = e^{-\frac{T_{max}^{(m)}}{T^{(m)}}} .$$

Thus the problem of assignment of resource capacity requirements C_i (3.8) - (3.10) can be rewritten as following one

$$S = \sum_{i=1}^{N} \alpha_i C_i \to min \qquad (3.11)$$

$$T^{(m)} = \sum_{i \in \pi_m} \frac{V_i}{C_i} + \sum_{j \in \pi_m} \frac{V_j}{C_j - \lambda_j V_j} = T_0^{(m)}, m = \overline{1, M} \qquad (3.12)$$

$$P(t_m > T_{max}^{(m)}) = e^{-\frac{T_{max}^{(m)}}{T^{(m)}}} = R^{(m)}, m = \overline{1, M} \qquad (3.13)$$

$$C_i \geq 0, i = \overline{1, N} \qquad (3.14)$$

where the first sum in expression (3.12) is written for DFGraph elements associated with multichannel model and the second sum is written for ones associated with queue model mentioned above.

After logariphming equation (3.13) the optimization problem can be rewritten:

$$S = \sum_{i=1}^{N} \alpha_i C_i \to min \qquad (3.15)$$

$$T^{(m)} = \sum_{i \in \pi_m} \frac{V_i}{C_i} + \sum_{j \in \pi_m} \frac{V_j}{C_j - \lambda_j V_j} = min\left(T_0^{(m)}, -\frac{T_{max}^{(m)}}{ln(R^{(m)})} \right)$$

$$m = \overline{1, M} \qquad (3.16)$$

$$C_i \geq 0, i = \overline{1, N} \qquad (3.17)$$

This problem is an optimization one with a nonlinear constraints. Using the Lagrangian method we get :

$$C_i = \sqrt{\frac{V_i \xi_i}{\alpha_i}} \qquad (3.18)$$

$$C_j = \sqrt{\frac{V_j \xi_j}{\alpha_j}} + \lambda_j V_j \qquad (3.19)$$

$$\xi_i = \sum_{(m / i \in \pi_m)} \beta_m, i = \overline{1, N} \qquad (3.20)$$

where β_m are determined from system of M nonlinear equations:

$$\sum_{i \in \pi_m} \sqrt{\frac{V_i \alpha_i}{\xi_i}} = min\left(T_0^{(m)}, -\left(\frac{T_{max}^{(m)}}{ln(R^{(m)})} \right) \right), m = \overline{1, M} \qquad (3.21)$$

The system (3.20) can be decided by one of numerical methods.

To get not optimal but a rational decision of the problem (3.15) - (3.16) that can be often usable for practical application we offer following approach:

- to decide the optimization problem (3.15), (3.16) for every separate path m = 1,...,M and further to correct resource capacities so that all M QoS requirements (3.16) are satisfied. Note if paths of DFGraph don't intersect with each other then this approach get **exact decision of original problem (3.15),(3.16).**

Let's decompose the problem (3.15)-(3.16) into M subproblems. The m-th subproblem is formulated as

$$S^{(m)} = \sum_{i \in \pi_m} \alpha_i C_i^{(m)} \to min$$

$$T^{(m)} = \sum_{i \in \pi_m} \frac{V_i}{C_i} + \sum_{j \in \pi_m} \frac{V_j}{C_j - \lambda_j V_j} = min\left(T_0^{(m)}, -\frac{T_{max}^{(m)}}{ln(R^{(m)})} \right)$$

Using Lagrangian method we get the following decision for m-th subproblem:

$$C_i^{(m)} = max\left(\frac{1}{T_0^{(m)}}, \left(-\frac{ln(R^{(m)})}{T_{max}^{(m)}} \right) \right) \sqrt{\frac{V_i}{\alpha_i}} \sum_{(n \in \pi_m)} \sqrt{\alpha_n V_n} \qquad (3.22)$$

$$C_j^{(m)} = \lambda_j V_j + max\left(\frac{1}{T_0^{(m)}}, \left(-\frac{ln(R^{(m)})}{T_{max}^{(m)}} \right) \right) \sqrt{\frac{V_j}{\alpha_j}} \sum_{(n \in \pi_m)} \sqrt{\alpha_n V_n} \qquad (3.23)$$

And now we choose the capacities of all elements on overall M paths

$$C_i = max|_{m/(i \in \pi_m)} \left\{ C_i^{(m)} \right\}, i = \overline{1, N} \qquad (3.24)$$

and summary resource cost for time unit (sec) is calculated by using equation (3.15) and for the DMMApplication duration T equals to SxT .

3.4. Stream synchronization parameters for basic schemes of DFGraph

In this section, we determine jitter and skew and get relations for its calculation.

3.4.1. Point-to-point scheme

In this case we deal with the subject of serial synchronization that determines the rate at which events must occur within a single data stream (intra-synchronization). In the wide sense of stands for all time-related issues a single medium stream has to deal with jitter. We define jitter as instantaneous difference between actual and desirable arrival time of message to receiver.

There are two approachs for jitter compensation:

- synchronization is required for every data unit (message) over all temporal stream,
- synchronization is required only between messages within a synchronization unit. The boundary of a synchronization unit is application-dependent, for example, the most common synchronization units are talkspurts for voice and frames for video. The using adaptive synchronization allows to divide the data (message) stream into smaller manageable units, and to consider the synchronization process only within a single synchronization unit.

Further we use the second model that was proposed by [Schulzrinne,92] and [Wang,93] and developed by [Stainov,94] for scheduling in sequence systems.

We differ two cases:

- recovery the regular data stream at the receiver-side, that is the receiver must get a regular data stream independent on temporal characteristics of an original stream,
- recovery the original data stream at the receiver-side, that is the receiver must get the reconstructed original data stream produced by receiver.

Let’s investigate the first case.

3.4.1.1. Recovery a regular stream

We assume that all messages are of the same size and consider point-to-point scheme presented by sender - server - receiver where a server can represent a separate computer or communication link. We say the message has arrivied or departed only when its last bit has arrivied or departed.

Suppose T is an interval of synchronization unit, M be the number of messages generated by sender during the synchronization unit. We assume that M is constant and is related with size of burst generated by the sender.

Let a_i be the arrival time of ith message and $A(t_1, t_2)$ - the number of messages arrived in the interval $[t_1, t_2)$. Then $A(a_1, a_m)$ is the number of messages arrived before $t = a_m$: $A(a_1, a_m) = m - 1$.

Define an average rate of sender in interval $[a_1, a_m)$, as

$$\lambda_m = \frac{A(a_1, a_m)}{a_m - a_1} = \frac{m-1}{a_m - a_1}, m = \overline{2, M} \qquad \textbf{(3.25)}$$

and minimum average rate of sender in its active phase $[a_1, a_m)$ of synchronization unit

$$\lambda_{min} = min\{\lambda_m, m = \overline{2, M}\} \qquad \textbf{(3.26)}$$

Define the maximum instantaneous rate of the sender

$$\lambda_{max} = max\left\{\frac{1}{a_m - a_1}, m = \overline{2, M}\right\} \qquad \textbf{(3.27)}$$

Let sender produces message stream with the rate λ that can vary in the interval $[\lambda_{min}, \lambda_{max}]$. Let ω be the desirable constant (deterministic) message arrival rate to the

receiver for data presentation, J - the maximum admissible jitter, μ - the server rate that can vary in the interval $[\mu_{min}, \mu_{max}]$, V - the size of server buffer.

A possible task of calculation point-to-point scheme parameters can be following:

- The receiver demands a regular arrival data stream with constant rate for presentation ω, admissible deviation from this rate is characterized by maximum jitter J. The sender is characterized by minimum and maximum rates $[\lambda_{min}, \lambda_{max}]$ so that $\lambda_{max} \geq \omega \geq \lambda_{min}$. It produces the burst of M messages. It's necessary to calculate synchronization interval T (for realizing start-stop mode for sender), admissible rates $[\mu_{min}, \mu_{max}]$ of the server and the size of server buffer V.

Obviously that

$$T = M/\omega \qquad \textbf{(3.28)}$$

1. CONSTANT SERVER RATE

First, we consider the simple case when the server provides the desirable constant rate for data presentation by receiver, that is $\mu = \omega$

In Figure 3.1a) the temporal behaviour of the point-to-point scheme is presented for the case when $\mu = \omega$ so that there are not server starvation or buffer overflow, server produces a desirable regular arrival stream to the receiver, and jitter J=0 always. (In Figure 3.1 μ is denoted as 'mu', λ_{max} is denoted as 'lmax' and ω is denoted as w).

For buffer size calculation we consider the worst case when the sender generate in a synchronization unit T all M message with maximum rate λ_{max}. Figures 3a) and 3b) explain together the behaviour of the system for this case. In Figure 3b) the line 'lmax' shows the process of buffer filling up with rate λ_{max} by sender on condition that server is stopped. The line

‘mu’ shows the process of empting full buffer by server with rate μ on condition that sender is stopped. And thik line ‘v’ shows the result process (current number of messages in the buffer) when sender and receiver perform with rates λ_{max} and μ simultaneously.

From Figure 3.1 follows Ta = M/λ_{max}. During the interval Ta -1/λ_{max} = (M-1)/λ_{max} server operates (M-1)μ/λ_{max} messages. So at the time point Tfull there are not more than

$$V = \left\lceil M - \frac{(M-1)\mu}{\lambda_{max}} \right\rceil \qquad \textbf{(3.29)}$$

messages in the buffer.

Relation (3.29) determines necessary buffer size. For our example in Figure 3.1 λ_{max}:μ = 2, M = 6 we get buffer size V = $\lceil 6 - 2.5 \rceil$ = 4 that is less than burst size M = 6 in the synchronization unit T.

If synchronization unit is chosen so that $T > M/\omega = M/\mu$ then at least to end of synchronization unit T we get the server starvation and regular stream will be destroyed (see Figure 3.2). For this case maximum jitter J = T - M/μ.

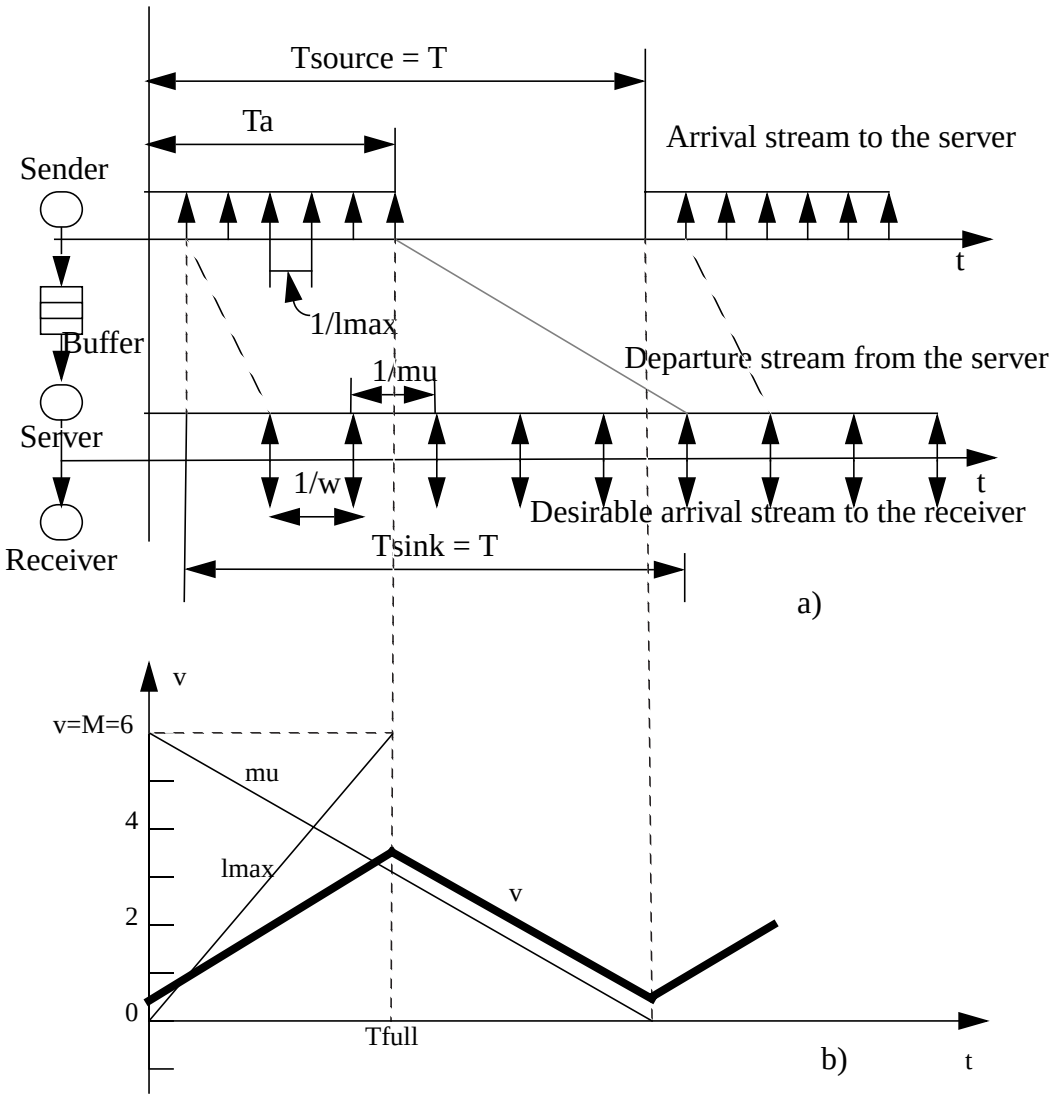


Figure 3.1. a) Time diagram for the point-to-point scheme,
b) Buffer behaviour for the worst case

2. VARIABLE SERVER RATE

Let's now the server has variable rate $\mu_{min} \le \mu \le \mu_{max}$. In this case the maximum jitter (see Figure 3.2 for the jitter calculation when $\mu = \mu_{max}$)

$$|J| = max\left\{T - \frac{M}{\mu_{max}}, \frac{M}{\mu_{min}} - T\right\} \quad \textbf{(3.30)}$$

Hence using (3.28)

$$\frac{M\omega}{M + J\omega} = \frac{M}{T + J} = \mu_{min} \le \mu \le \mu_{max} = \frac{M}{T - J} = \frac{M\omega}{M - J\omega} \quad \textbf{(3.31)}$$

If server succeeds to service all M messages to the end of synchronization unit (that is $\mu_{min} \ge \omega$ and we have only positive jitter J = T - M/ μ_{max}) then the needed buffer size is

$$V = \left\lceil M - \frac{(M-1)\mu_{min}}{\lambda_{max}} \right\rceil \quad \textbf{(3.32)}$$

If $\mu_{min} < \omega$ then there is no zero probability of buffer overflow because a remainder of unserviced messages in the buffer to the end of consequent synchronization units can be increased in time. Thus we get in this case probability task for buffer size calculation. (It will be considered further).

Probability of buffer overflow must be take into account for calculation of such the QoS parameter as data loss rate , see expressions (3.2).

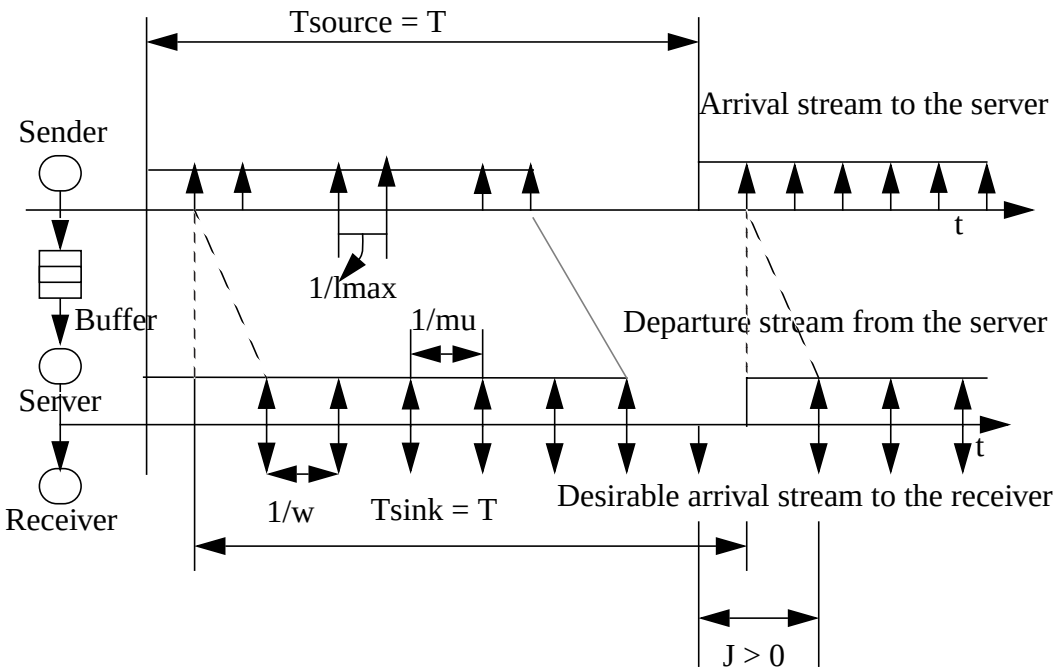


Figure 3.2. Time diagram for the incorrect synchronization unit T

3. ADAPTIVE SERVER RATE

4. SERVICE WITH ONE SYNCHRONIZATION UNIT DELAY

5. SEQUENCE OF SERVERS BETWEEN SENDER AND RECEIVER

3.4.1.2. Recovery an original stream

3.4.2. Collector scheme

4. MAPPING PROBLEM

There are two basic approaches for mapping decision: static and adaptive. In the static approach, a mapping is made before a session is established and is fixed for lifetime of the session if it is successful. During mapping it's assumed that the DCSystem will be not changed. In this case the success of mapping decision depends also on relation between a time needed for mapping execution and an actual time of invariable system state or of insignificant system state changing doesn't influencing on mapping decision. This disadvantage of static approach can be sometimes overcome by development of effective fast mapping methods.

In the adaptive approach the mapping is made during the session establiment phase (by pattern step-by-step) and is adjustable depending on current states of system components (computers and communication subsystem). But in this case there are other difficulties, for example, deadlocks. Besides only one of possible (not optimal) decision can be reached.

There is a middle approach too - substatic one based on system domains.

Obviously, it is helpful to have different kinds of mapping algorithms that are choosed depending on temporal and other features of DCSystem and DMMApplications.

In this section, we consider the analytical foundations for static approach that allows given a particular topology, node and links capacities of DCSystem, to meet the QoS requirements of DMMA with significantly reduced resource demands by preliminary proper calculations.

4.1.System of methods and models for mapping decision

System of methods and models for mapping decision is represented in Figure 4.1.

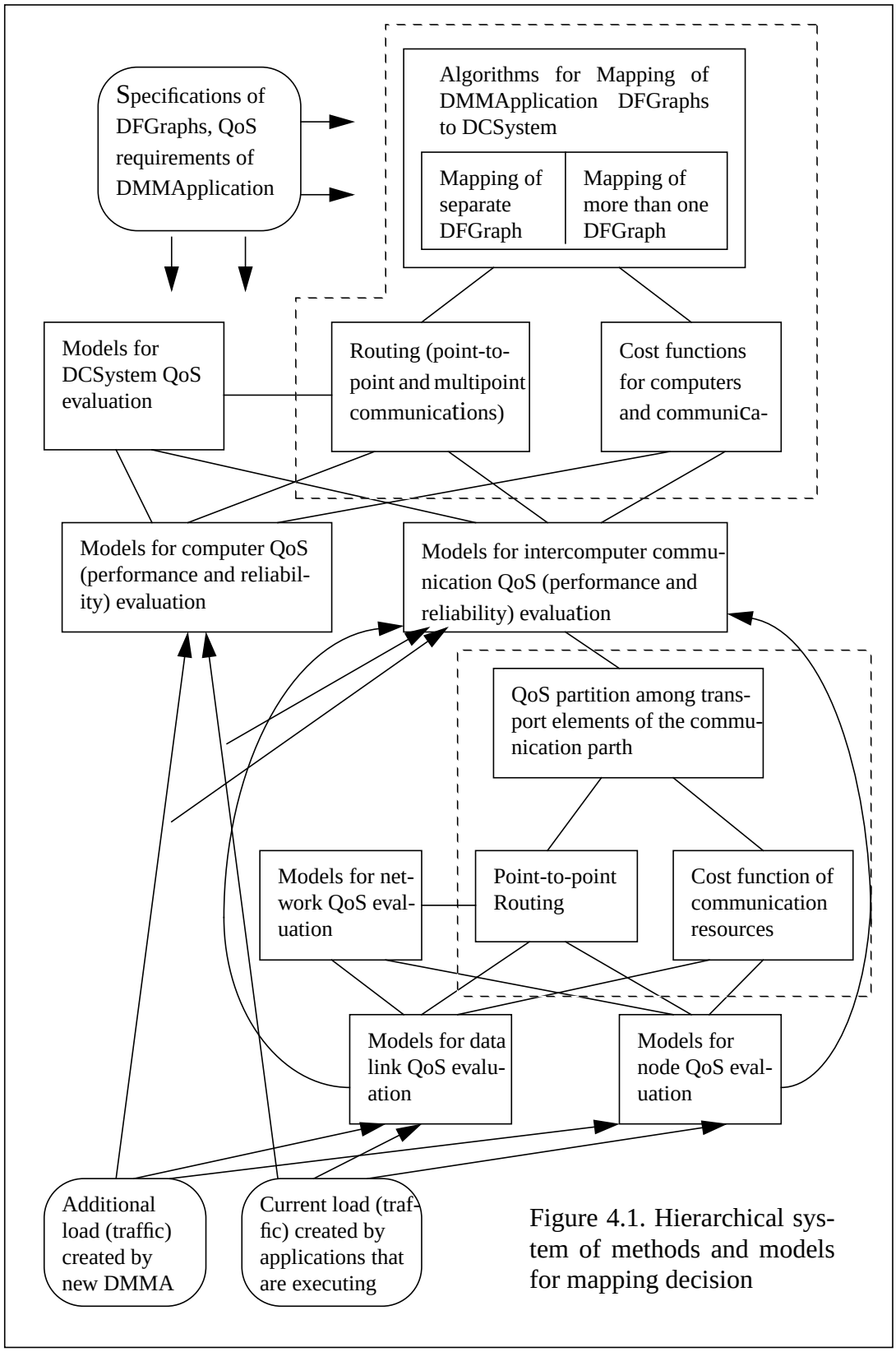


Figure 4.1. Hierarchical system of methods and models for mapping decision

4.2. Mapping task statement

In Appendix, a general task statement of DFGraph mapping to DCSystem is represented. In this section, we formulate the particular mapping task as a mathmetical programming one using decision of assignment of resource capacity requierements to DFGraph elements (see, Section 3.3).

GIVEN:

- DFGraph with

CD - the set of components;

$E = \{\langle i, j\rangle, i, j \in CD\}$ - the set of directed edges connecting components with each other;

$C = \{\{C_i, i \in CD\}, \{C_{ij}, \langle i, j\rangle \in E\}\}$ - the set of needed resources (capacities) for components (C_i) and for edges (C_{ij}). These data we get using the approach represented in Section 3.3;

- DCSystem with

P - the set of computers (nodes);

$L = \{\langle i, j\rangle, i, j \in P\}$ - the set of directed links connecting computers. Here a link is a logical element which can represent actual subnetwork(s) that's nodes and links are not accessible for CINEMA's management;

$A = \{\{A_i, i \in P\}, \{A_{ij}, \langle i, j\rangle \in L\}\}$ - the set of available (vacant) resources (capacities) of computers (A_i) and links (A_{ij}) . Let's explain. If, for example, computer has an origin capacity (servise rate) Z and its utilization factor (by current processing tasks) equals to r < 1 then its available resource equals to A = (1-r) Z;

$\alpha = \{\{\alpha_i(C), i \in P\}, \{\alpha_{ij}(C), \langle i, j\rangle \in L\}\}$ - the set of resource cost functions for computers (α_i) and links (α_{ij}). Here C is the volume of a needed resource.

- $PL = \{PL_i, i \in CD\}$ - the set of admissible placing of components in DCSystem such that $PL_i = \{j \in P\}$ is a set of computers on which a component $i \in CD$ may be placed (that means, does computer configuration have needed devices and resources to realize multimedia functions of certain component type?);

THE DECISION VARIABLES are x_{ij} such that $x_{ij} = 1$ if component i is placed onto computer j and $x_{ij} = 0$ otherwise.

If δ_{ij} is defined as $\delta_{ij} = 1$ if i = j and $\delta_{ij} = 0$ otherwise then

MAPPING PROBLEM can be formulated as:

$$F(\{x_{ij}\}) =$$

$$min\left\{\begin{array}{c}\sum\limits_{(i \in CD)}\sum\limits_{(j \in PL_i)} x_{ij}\alpha_j(C_i) + \ldots \\ \sum\limits_{(\langle i,k\rangle \in E)}\sum\limits_{(j \in PL_i)}\sum\limits_{(n \in PL_k)} x_{ij}x_{kn}\delta_{jn}\alpha_{jn}(C_{ik})\end{array}\right\} \quad (4.1)$$

$$\sum_{(j \in PL_i)} x_{ij} = 1, \forall i \in CD \quad (4.2)$$

$$\sum_{(i \in CD)} x_{ij}C_i \le A_j, \forall j \in P \quad (4.3)$$

$$\sum_{\langle i,k\rangle \in E}\sum_{(j \in PL_i)}\sum_{(n \in PL_k)} x_{ij}x_{kn}\delta_{jn}C_{ik} \le A_{jn}, \forall \langle j,n\rangle \in L \quad (4.4)$$

$$x_{ij} \in [0,1], i \in CD, j \in P \quad (4.5)$$

In this formulation, the objective function F minimizes the summary resource cost. The first term in the objective function identifies the cost of resources of computers on which components are placed. The second term represents the cost of communication resources of links on which edges are placed. Note that multiplier δ_{jn} equals to 1 if and only if components i and k connected immediately with each other in DFGraph are placed on different computers j and n. In this case the edge connecting these both components (i and k) must be mapping to the link connecting the computer j with the computer n.

Constraint set (4.2) guarantees that every component $i \in C$ must be placed into DCSystem and can be placed only onto one computer.

Constraint set (4.3) guarantees that resources used by components placed on a computer does not exceed the available resource of the computer. Constraint (4.4) is similar (4.3) but for edges and links.

The formulation (4.1) - (4.5) allows to search the placing for every component of DFGraph. If some components have to be placed on certain computers (for example, sources and sinks) then corresponding variables x_{ij} must be fixed by 1 and constraint sets must be added by following kinds of equations: $x_{ij} = 1$ if component i have to be placed on computer j.

The problem (4.1) - (4.5) can be generalized for mapping of a group of DMMApplications. Let's denote SA - the set of DMMApplication DFGraphs, a - one of DFGraphs belonged the set SA, $a \in SA$. Let's add identificator a of DFGraph as a high index to all previous nota-

tions associated with DFGraph and used in (4.1) - (4.5): CD^a, E^a, C^a, PL^a, PL_i^a, x_{ij}^a. Then mapping problem for the set SA of DFGraphs of a DMMApplication group can be formulated as:

$$F\left(\left\{x_{ij}^a\right\}\right) =$$

$$min\left\{\begin{array}{l}\sum\limits_{(a \in SA)}\sum\limits_{(i \in CD^a)}\sum\limits_{(j \in PL_i^a)} x_{ij}^a \alpha_j(C_i^a) + \ldots \\ \sum\limits_{(a \in SA)}\sum\limits_{(\langle i,k\rangle \in E^a)}\sum\limits_{(j \in PL_i^a)}\sum\limits_{(n \in PL_k^a)} x_{ij}^a x_{kn}^a \delta_{jn} \alpha_{jn}(C_{ik}^a)\end{array}\right\} \quad (4.6)$$

$$\sum_{(j \in PL_i^a)} x_{ij}^a = 1, \forall i \in CD^a, \forall a \in SA \quad (4.7)$$

$$\sum_{(a \in SA)}\sum_{(i \in CD^a)} x_{ij}^a C_i^a \le A_j, \forall j \in P \quad (4.8)$$

$$\sum_{(a \in SA)}\sum_{\langle i,k\rangle \in E^a}\sum_{(j \in PL_i^a)}\sum_{(n \in PL_k^a)} x_{ij}^a x_{kn}^a \delta_{jn} C_{ik}^a \le A_{jn}, \forall \langle j,n\rangle \in L \quad (4.9)$$

$$x_{ij}^a \in [0,1], i \in CD^a, a \in SA, j \in P \quad (4.10)$$

4.3. Approachs and algorithms for mapping decision

See [Norman, 93], [Ramanathan,92], [Narasimhan,94], [Dimitrijevic, 94], [Tindell, 92]

5. MODELS for PERFORMABILITY and RELIABILITY EVALUATION of DISTRIBUTED COMPUTER SYSTEM (DCSystem), its SUBSYSTEM and ELEMENTS

5.1. Models for computer performance and reliability evaluation

See [Kapelnikov,89], [Cruz,91a], [Cruz,91b]

5.2. Models for evaluation of data stream

See [Cruz,91a], [Cruz,91b], [Stamoulis,94], [Stirpe,92], [Blum, 94], [Wang,94]

5.3. Models for data link performance and reliability evaluation

See [Hagin,94]

5.4. Models for network performance and reliability evaluation

See [Gagin, 91], [Hagin,94]

5.5. Models for DCSystem performance and reliability evaluation

See [Cruz,91a], [Cruz,91b], [Dimitrijevic, 94], [Kapelnikov,89], [Hagin,94]

5.6. Models for cost functions of computer and communication resources

6. EXAMPLE

Let’s consider an example of tasks associted with mapping problem. The DFGraph of a DMMApplication is represented in Figure 6.2 using graphical symbol notations shown in Figure 6.1. Let’s number DFGraph elements as in Figure 6.3. There are given following data:

- Numbers of computer operations needed for message processing by components:

  V1 = V2 = V3 = V7 = V8 = 2v, V4 = 3v, V5 = V6 = 4v;

  Message lengths for edges:

  V9 = V10 = V11 = V12 = V13 = V14 = V15 = V16 = v,

where v is multiplier, for example v= 2500 bytes for edges and v = 2500 operations for components. (Note, we introduce multiplier only for calculation reduction. In general case all Vi may be different).

- Set of media paths:

  π_1 = (1-->5) = (1,9,4,11), π_2 = (2-->5) = (2,10,4,11),

  π_3 = (1-->6) = (1,9,4,12), π_4 = (3-->7) = (3,13,5,15,7),

  π_5 = (3-->8) = (3,14,6,16,8).

  Note we get exact decision for the problem (3.15), (3.16) for this set of paths choosed so that there are not intersected ones.

- Loss rates for every paths:

  R1 = R2 = R3 = 0.1,

  R4 = R5 = 0.01;

- Average throughput for every path (in its last edge);

  H1 = H2 = H3 = 160 KBit/sec = 8 mes/sec,

  H4 = H5 = 20 Mbit/sec = 1000 mes/sec

- Average message delay for every path:

  T1 = T2 = T3 = 0.3 sec, T4 = T5 = 0.01 sec;

- Average stream rate for every DFGraph element:

  $\lambda_1 = \lambda_9 = \lambda_2 = \lambda_{10} = 4$ mes/sec,

  $\lambda_4 = \lambda_{11} = \lambda_{12} = 8$ mes/sec,

  $\lambda_3 = \lambda_{13} = \lambda_{14} = \lambda_5 = \lambda_6 = \lambda_{15} = \lambda_{16} = \lambda_7 = \lambda_8 = 1000$ mes/sec,

- Costs of resource unit for every element of DFGraph:

  $\alpha_i = 2, i = \overline{1, 8}$

  $\alpha_i = 1, i = \overline{9, 16}$

Using equation (3.13) we get maximum message delays for every path (note if we take into account jitter requirements the maximum delays may be corrected):

T1max = T2max = T3max = 0.69 sec,

T4max = T5max = 0.046 sec.

Using equation (3.22) for edges and equation (3.23) for components we get for every element of the first path π_1 the needed capacity requirements:

$C_1^{(1)} = 29.5v = 737\,500$ op./sec, $C_9^{(1)} = C_{11}^{(1)} = 21.5v = 430\,000$bit/sec,

$C_4^{(1)} = 50.3v = 123\,750$ op./sec,

for every elements of the forth path π_4 :

$C_3^{(4)} = C_7^{(4)} = 2883v = 7\,207\,500$ op./sec, $C_5^{(4)} = 5248v = 13\,120\,000$ op./sec,

$C_{13}^{(4)} = C_{15}^{(4)} = 883v = 17\,660\,000$bit/sec.

and so on.

Then using equation (3.24) we get resource requirements for all elements of DFGraph

$C_1 = C_2 = 29.5v = 737\,500$ op./sec,

$C_3 = C_7 = C_8 = 2883v = 7\,207\,500$ op./sec,

$C_4 = 50.3v = 123\,750$ op./sec,

$C_5 = C_6 = 5248v = 13\,120\,000$ op./sec,

$C_9 = C_{10} = C_{11} = C_{12} = 21.5v = 430\,000$ bit/sec,

$C_{13} = C_{14} = C_{15} = C_{16} = 883v = 17\,660\,000$ bit/sec.

Now we are ready to decide mapping task (4.1) - (4.5).

DCSystem shown in Figure 6.4 can be represented by complete (full) graph, in which every node (computer) is connected with all other nodes (computers). But taking into account given placements of sources and sinks and also the capabilities of computers to realiaze only particular types of components we can construct a relevant graph of DCSystem that simplifies the mapping problem decision. Such the relevant graph of DCSystem is presented in Figure 6.5. This graph is directed one and includes all possible placements of every component of DFGraph of particular DMMApplication in the DCSystem.

In Figures 6.6 - 6.8 possible mapping decisions are represented.

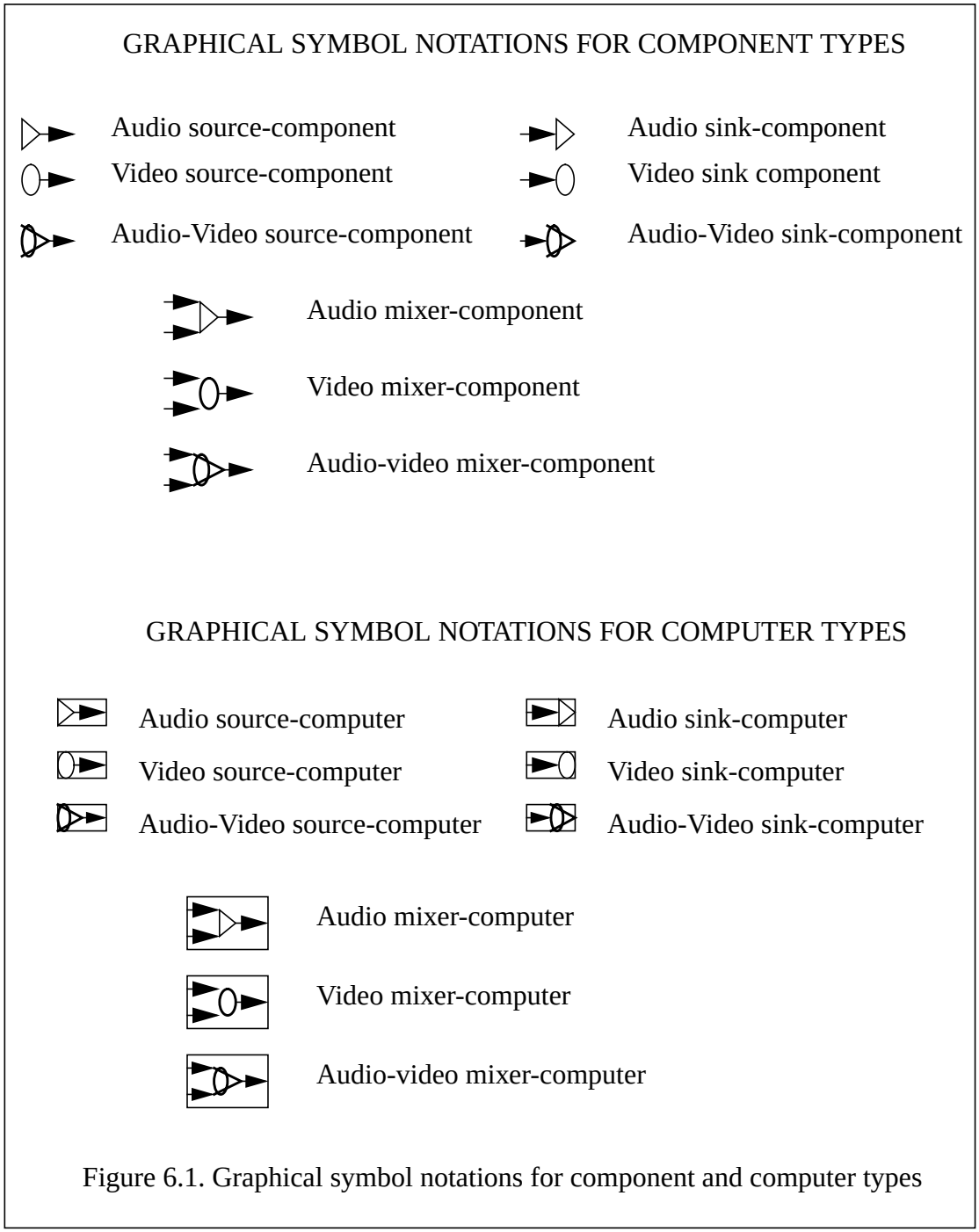


Figure 6.1. Graphical symbol notations for component and computer types

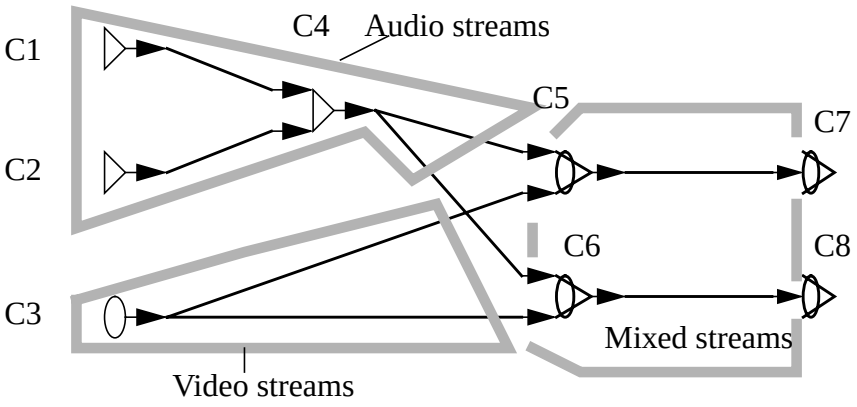


Figure 6.2. Data flow graph

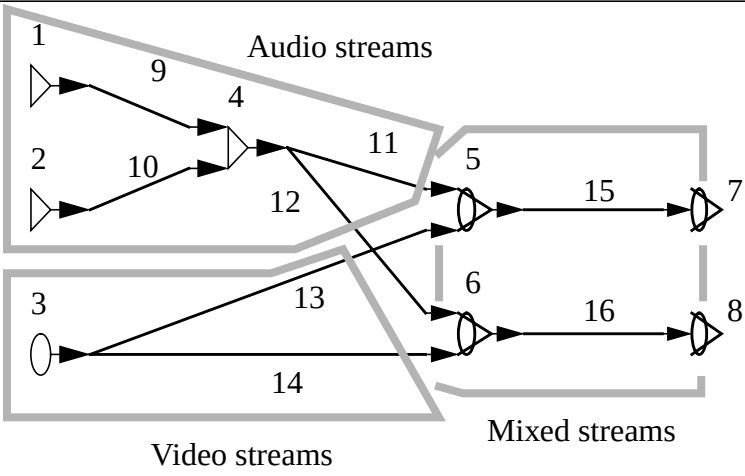


Figure 6.3. Numbered Data flow graph

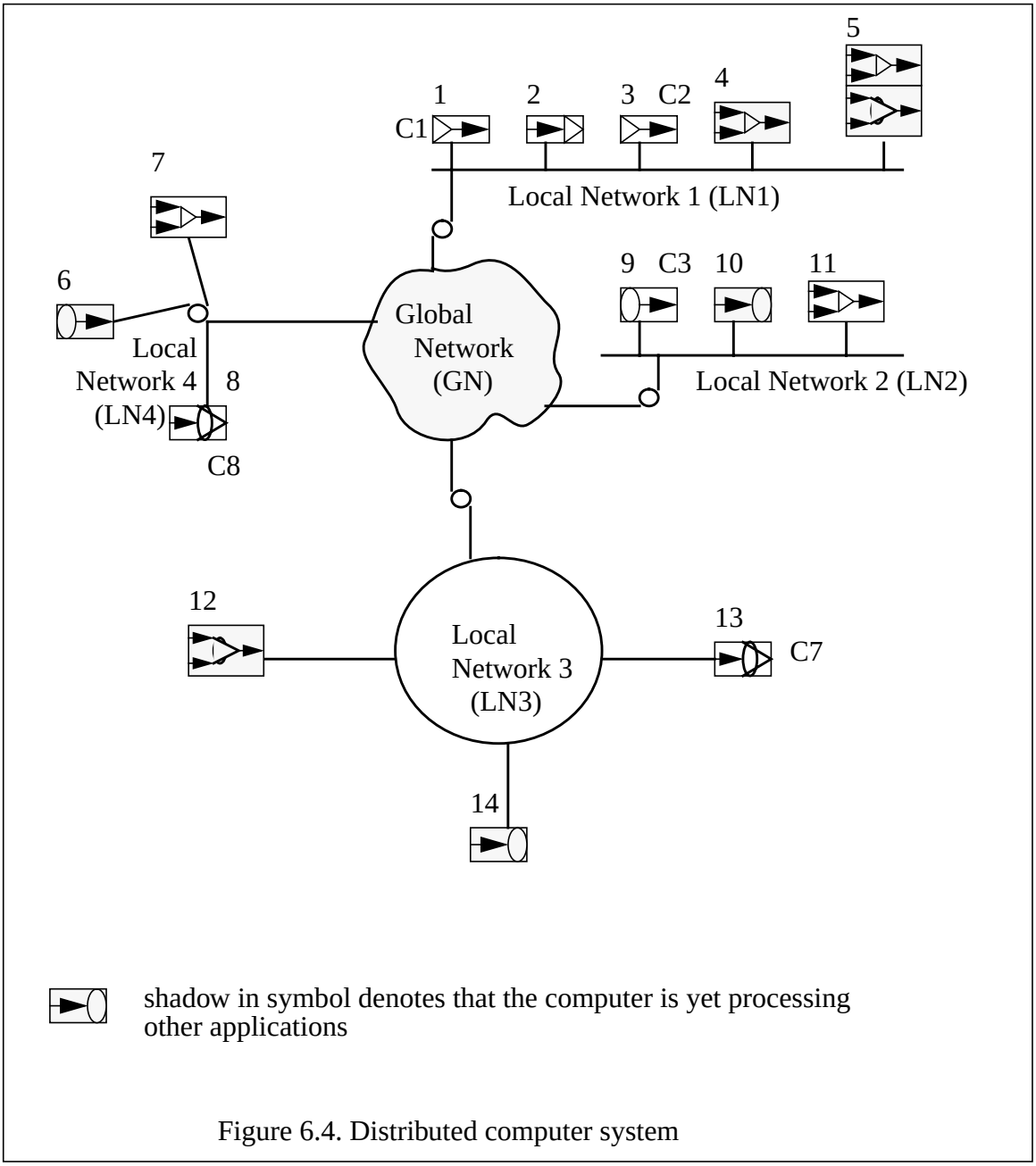


Figure 6.4. Distributed computer system

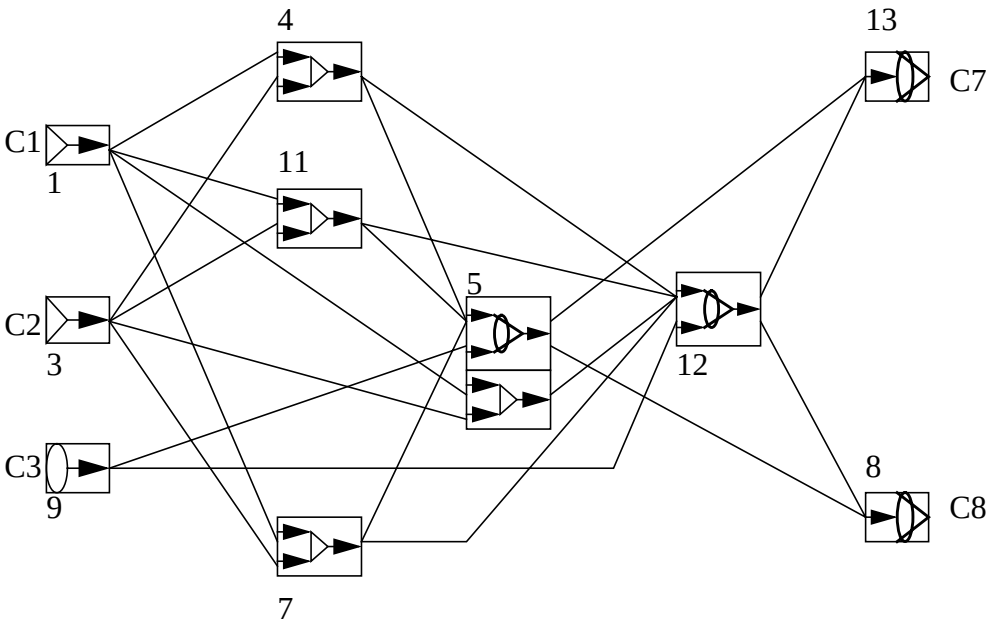


Figure 6.5. Relevant graph of DCSystem

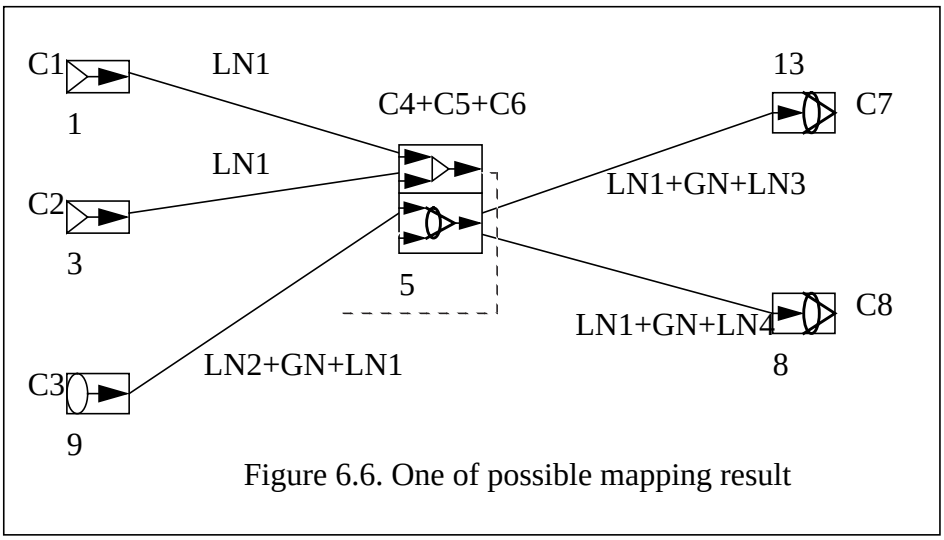


Figure 6.6. One of possible mapping result

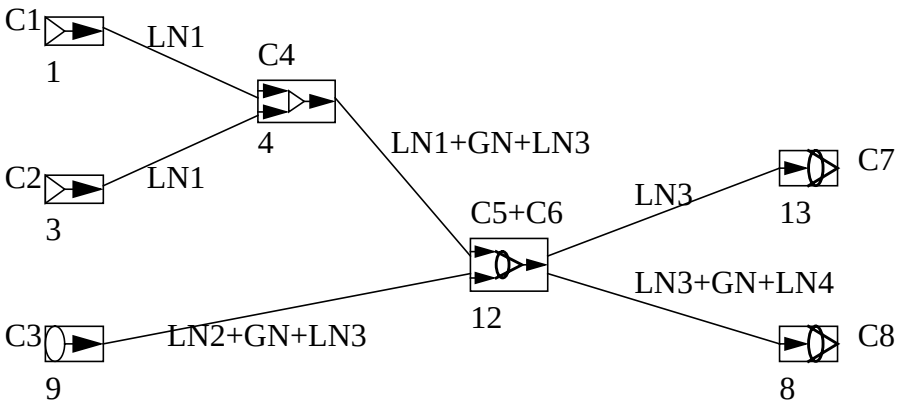


Figure 6.7. One of possible mapping result

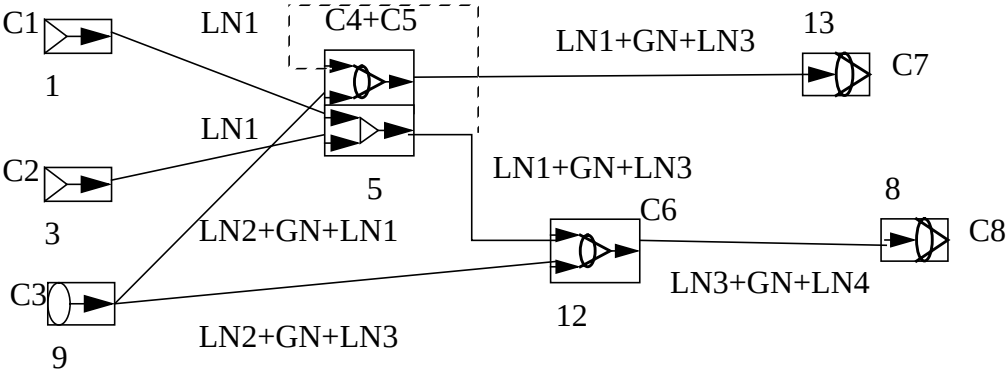


Figure 6.8. One of possible mapping result

7. CONCLUSION

In this paper, we examined problem of mapping of DMMApplication to DCSystem. We have formulated the problem and for some tasks got efficient solutions. But a set of models and solution procedures mentioned above must be yet developed.

For the support and for making this work possible for me I would like to thank Prof. K.Rothermel. I express my thanks to I.Barth, G.Dermler, T.Helbig for their useful comments and discussions. I thank research and administrative staff of Distributed Systems Department for their help in my work.

APPENDIX. GENERAL TASK STATEMENT of DFGraph MAPPING to DCSystem

Let's consider the statement of mapping task core that are shown by dotted line figure at the top of Figure 4.1.

THERE ARE GIVEN:
A 4-tuple (FG, Q, G_S, SF) , where

FG specifies a DFGraph of DMMApplication,

Q specifies QoS requirements of DMMApplication,

G_S specifies a relevant part of DCSystem,

SF specifies relations between DFGraph components and DCSystem elements.

- $FG = (G_D, \Omega, CT, RC, F_D)$, where

$G_D = (C, E)$ is a directed weighted acyclic graph representing data flows between source(s) and sink(s) of a DMMApplication such that C is a set of $N_C = |C|$ components and E is a set of the $N_E = |E|$ edges connecting components and representing a partial order on the components;

$\Omega = \{\pi_m, m = \overline{1, K}\}$ is a set of K media paths (or simply paths) in DFGraph (for each path QoS requirements are given). Here we use termin "path" as it is defined in section 3.1;

$CT = \{\tau_c, c \in C\}$ is a set of component types such that τ_c is a type of component $c \in C$ of data flow graph G_D;

$RC = (RC_c, c \in C)$ is a set of components requests of the computer resources. (for instance, $RC_c = (NO_c, VM_c, VD_c, NIO_c, VIO_c)$, where NO_c is number of CPU operations nedeed for DMMApplication program execuation, VM_c is the size of nedeed main memory, VD_c is the size of needed disk memory, NIO_c is a number of input-output operations during CPU execution of DMMApplication program, VIO_c is average size of data trasmitted between main memory and disk memory per one input-output operation of DMMApplication program);

$F_D = \{d_e, e \in E\}$ is a set of values d_e indicating the number of bytes of data message

(paket) that must be transmitted through the edge e from the precedent component to its successor . Here a message is a data unit associated with QoS requirement;

- $Q = (Q_m = (T_m, H_m, R_m, J_m), m = \overline{1, K})$ is a set of QoS application requirements for K paths such that Q_m is a set of m-th subgroup QoS requriments namely delay T_m, throughput H_m, loss rate R_m and jitter J_m (the list of QoS parameters depends on DMMApplication);
- $G_S = (P, L)$ is an directed weighted graph representing a DCSystem such that P is a set of $N_P = |P|$ computers that are accessible for CINEMA's management and L is a set of the $N_L = |L|$ links connecting computers. Here a link is a logical element which can represent actual subnetwork(s) that's nodes and links are not accessible for CINEMA's management;
- $SF = (\gamma, \chi, F_S)$, where

γ is a mapping function $\gamma: \Omega \to P$ returning a computer $\gamma(c) = p \in P$ on which a source- or sink-component $c \in \Omega$ is placed. (We assume assignment of source- and sink-components to computers are given);

χ is a function $\chi: P \to \bigcup_c \tau_c$ that returns a set of component types $\chi(p)$ which are

able to be mapped to computer $p \in P$ (that means, does computer configuration have needed devices and resources to realize multimedia functions of certain component type?);

$F_S = \{\{s_p(c, q_{pc}), p \in P, c \in C\}, \{s_l(e, q_{le}), l \in L, e \in E\}\}$ is a set of cost functions

$\{s_p(c, q_{pc}), p \in P, c \in C\}$ for computers $p \in P$ to which components $c \in C$ are being mapped with a set of QoS parameters q_{pc} ,

$\{s_l(e, q_{le}), l \in L, e \in E\}$ for links $l \in L$ to which edges $e \in E$ are being mapped with a set of QoS parameters q_{le},

q_{pc}, q_{le} is a set of QoS parameter values accordingly for computer $p \in P$ to which component $c \in C$ is being mapped and for link $l \in L$ to which edge $e \in E$ is being mapped. For example $q_{pc} = (t, h, r, j)$, where t - message delay, h - throughput, r - loss rate and j - jitter in computer p to which component c is being mapped.

Note, that cost functions s_p, s_l are dinamic ones that are construct for available resources of a computer and a link and therefore must be corrected every time after placing any component or edge of DFG. If, for example, computer p doesn't have available resources then $s_p = \infty$.

ASSUMPTIONS:

1. A cost functions of F_S must be tractable, that is monotonic, ...
2.

ADDITIONAL NOTATIONS

Let $c_{pr}(e), c_{su}(e)$ denote the function-pair that define the pair of precedent and successor components correspondingly that are connected by edge e in data flow graph G_D.

Let (M, Ψ) denote the set of all mapping double-functions, (μ, ψ), such that $\mu: C \to P$ in accordance with function $\chi: P \to \bigcup_c \tau_c$, and $\psi: E \to L \cup 0$. Function $\mu \in M$

returns the computer $\mu(c) = p \in P$ on which a component $c \in C$ is executed in the mapping defined by μ such that component type τ_c belongs $\zeta(p), \tau_c \in \chi(p)$. (Assignment of

some components $c \in C$ to the same computer $p \in P$ is permitted). Function $\psi \in \Psi$ returns the link $\psi(e)= (l \in L)$ between computer $p_1= \mu(c_1(e))$ and computer $p_2= \mu(c_2(e))$ in actual distributed system; edge $e \in E$ is mapped to the link $\psi(e)$. If some components (for example, c_1 and c_2) are mapped to the same computer $p \in P$ (that is $\mu(c_1)= \mu(c_2) = p$), then a link of DCSystem is not needed and the function $\psi \in \Psi$ returns null-element 0, that is $\forall c_1, c_2 \in C \therefore \mu(c_1) = \mu(c_2) = p, \exists e \in E \therefore c_1 = c_{pr}(e), c_2 = c_{su}(e) \Rightarrow \psi(e) = 0$

DECISION PROBLEM

Given an 4-tuple (FG, Q, G_s, SF) , find a mapping double-functions $(\mu, \psi) \in (M, \Psi)$ such that

$$min|_{(\mu, \psi)} \left(\begin{array}{c} \sum\limits_{c \in C} s_{\mu(c)}(c, q_{\mu(c)c}) \\ \ldots + \sum\limits_{(e \in E)} \sum\limits_{(l \in \psi(e))} s_l(c, q_{le}) \end{array} \right)$$

subject to

$$\hat{T}_m(\{\varphi(c)\}, c \in \Omega_m) \le T_m, m = \overline{1, K}$$

$$\hat{R}_m(\{\varphi(c)\}, c \in \Omega_m) \le R_m, m = \overline{1, K}$$

$$\hat{H}_m(\{\varphi(c)\}, c \in \Omega_m) \ge H_m, m = \overline{1, K}$$

$$\hat{J}_m(\{\varphi(c)\}, c \in \Omega_m) \le J_m, m = \overline{1, K} \ ,$$

where $\hat{T}_m, \hat{R}_m, \hat{H}_m, \hat{J}$ are functions correspondingly of delay, loss rate, throughput and jitter, defined for path $m = \overline{1, K}$.

A result of mapping problem decision may be presented by a subgraph $\Gamma = (\hat{P}, \hat{L})$ of $G_S = (P, L)$ such that

$\Omega \subseteq \hat{P} = \{\mu(c), c \in C\} \subseteq P, \hat{L} = \{l \in \psi((e), e \in E)\} \subseteq L$. Graph Γ represents a transport graph from sources to sinks of group association $\Omega = \bigcap \Omega_m$ in actual distributed computer system. In particular, graph Γ can be consist of only one node that represents a computer $p \in P$ to which all components $c \in C$ of data flow graph G_S are mapped.

(Note that the question of possible cycles in Γ, when source(s) and sink(s) are mapped to the same computer, must be considered.)

THE DECISION OF MAPPING PROBLEM NEEDS:

- to define delay $\hat{T}_m$, loss rate $\hat{R}_m$, throughput $\hat{H}_m$ and jitter $\hat{J}_m$ for every path $m = \overline{1, K}$.
- to elaborate models for computing end-to-end QoS parameters;
- to construct models for computing QoS parameters of every separate computer $p \in P$ to which components $c \in C$ are being mapped, for different schedules. It is necessary to take into consideration such a fact if the computer executes already other applications then

these applications will influence the parameters of a new application as well as a new application placed on the computer may change the parametersof already executed applications. The models must take into account multiprogramming and multiprocessor performance modes of computers, profiles of tasks, schedule algorithms and so on.

Each time when the mapping problem must be decided for a new application it's necessary take into consideration the current load states of every computer of distributed system. The current load state of computer system is changed each time when one of DMMApplications is terminated, one new DMMApplication request is received, a failure happens and so on;

- to determine the cost functions $\{s_p(c, q_{pc}), p \in P, c \in C\}$ for computers and $\{s_l(e, q_{le}), l \in L, e \in E\}$ for links depending on QoS requirements q_{pc}, q_{le} ;
- to develop algorithms for solving the division of QoS requirements among computers and links (communication resources) of multipoint route that should optimize the cost of DMMApplication. It is so named QoS requirement horizontal mapping problem;
- to elaborate a method for QoS requirement mapping between different protocol layers - QoS vertical mapping problem;
- to develop multipoint routing algorithms taken into account peculiarities of DMMApplications;
- to develop models such that fault tolerance can be taken into consideration.

REFERENCES

[Anderson,93] D.P.Anderson, “Metascheduling for Continuous Media”, ACM Trans. Computer Systems, Vol. 11, No. 3, August 1993, pp. 226 - 252.

[Blum, 94] C.Blum, “Synchronization of Live Continuous Media Streams”, Proc. The 4th Open Workshop on High Speed Networks, Brest, September 7 - 9 ,1994: Universität Stuttgart, pp. 234 - 240.

[Campbell,94] A.Campbell, G.Coulson, D.Hutchison, “A Quality of Service Architecture”, Computer Communication Review, ACM SIGCOMM, Vol. 24, No 2, April 1994, pp. 6 - 27.

[Cruz,91a] R.L.Cruz, “A Calculus for Network Delay, Part 1: Network Elements in Isolation”, IEEE Trans. Information Theory, Vol. 37, No 1, January 1991, pp. 114 - 131.

[Cruz,91b] R.L.Cruz, “A Calculus for Network Delay, Part 2: Network Analysis”, IEEE Trans. Information Theory, Vol. 37, No 1, January 1991, pp. 132 - 141.

[Dimitrijevic, 94] D.D.Dimitrijevic, B.Maglaris, R.R.Boorstyn, “Routing in Multidomain Networks”, IEEE/ACM Trans. Networking, Vol. 2, No. 3, June 1994, pp. 252 - 262.

[Fuhrt,94] B.Fuhrt, “Multimedia Systems: An Overwiew”, IEEE MultiMedia, Vol.1, No.1, Spring 1994, pp. 47-59.

[Gagin, 91] A.A.Gagin, O.V.Klimovsky, “A Method for Computing Steady-state Reliability Indexes of a Network with Limited Repair”, Microelectron. Reliab., R-31, No. 5, 1991, pp. 985 - 999.

[Hagin,94] A.A.Hagin, “Performability, Reliability, and Survivability of Communication Networks: System of Methods and Models for Evaluation”, Proc. of the 14th International Conference on Distributed Computing Systems, Poznan, Poland June 21-24 1994, IEEE Computer Society Press, 1994, pp. 562 - 573.

[Kapelnikov,89] A.Kapelnikov, R.R.Muntz, M.D.Ercegovac, “A Modelling Methodology for the Analysis of Concurrent Systems and Computations”, J. Parallel and Distributed computing, No. 6, 1989, pp. 568 - 597.

[Leopold,92] H.Leopold, A.Campbel, D.Hutchison, N.Singer, “Towards an Integrated quality of Service Architecture (QoS-A) for Distributed Multimedia Communications”, High Performance Networking 92, IFIP, 1992,pp. D2-1 - D2-13.

[Kleinrock,76] L.Kleinrock, “Queueing Systems: Computer Applications”, Vol.2, N.Y.: John Wiley & Sons, 1976, p. 549.

[Maisonnaux,94] H.Maisonnaux, P.Cocquet, M. Gagnaire “New Concepts for Multipeer Communications”, Proc. 4th Open Workshop on High Speed Networks, Brest, September 7-9, 1994, pp.103-109.

[Narasimhan,94] S.Narasimhan, “Locating concetrators in a computer network with multiple coverage for terminals”, Computer Communication, Vol. 17, No. 2, February 1994, pp.94 -102.

[Norman, 93] M.G.Norman, P.Thanisch, “Models of Machines and Computation for Mapping in Multicomputers”, ACM Computing Surveys, Vol. 25, No. 3, September 1993, pp. 263 - 302.

[Ramanathan,92] S.Ramanathan, P.V.Rangan, M.M.Vin, T.Kaeppner, “Optimal Communication Architectures for Multimedia Conferencing in Distributed Systems”, Proc. The 12th International Conference on Distributed Computing Systems: Yokohama, Japan, June 9-12,1992, pp. 46 -53.

[Rothermel,94] K.Rothermel, I.Barth, T.Helbig, “CINEMA- An Arhitecture for Config-

urable Distributed Multimedia Applications", Tech. Report, Fakultaetbericht #3/1994, Universität Stuttgart, April 1994. p.20.

[Schulzrinne,92] H.Schulzrinne, " Issues in Designing a Transport Protocol for Audio and Video Conferences and other Multiparticipant Real-Time Applications", Internet-Draft, Dec. 1992.

[Stainov,94] R.Stainov, "Betriebsystemaspecteb des Realzeitscheduling in multimedialen Systemen", Tech. Report #12/1994, Universität Stuttgart, September 1994, p. 40.

[Stamoulis,94] G.D.Stamoulis, M.E.Anagnostou, A.D.Georgantas, "Traffic source models for ATM networks: a servey", Computer Communications, Vol. 17, No 6, June 1994, pp. 428 - 438.

[Stirpe,92] P.Stirpe, E.Pinsky, "Performance Analysis of an Asynchronous Multi-rate Crossbar with Bursty Traffic", Computer Communication Review, Vol. 22, No. 4, October 1992, pp. 150 - 160.

[Tindell, 92] K.W.Tindell, A.Burns, A.J.Wellings, "Allocating Hard Real-Time Tasks: An NP-Hard Problem Made Easy", J. Real-Time Systems, No. 4, 1992, pp. 145 - 165.

[Wang,94] Z.Wang, "Analysis of Burstiness and Jitter in Real-Time Communications", Proc. SIGCOMM'93 "Communications Architectures, Protocols and Applications", Computer Communication Review, Vol. 23, No.4, October 1993, pp. 13-19.

[Woo,94] M.Woo, N.U.Qazi, A.Ghafoor, "A Synchronization Framework for Communication of Pre-orchestrated Multimedia Information", IEEE Network, January/February 1994, pp. 52 - 61.