Control, computer engineering and information

UDC 681.3.06

DETERMINATION OF DATA TRANSFER CONTENT IN COMPUTER NETWORK FOR SPECIFIED MODEL OF SOFTWARE LOAD

A.V. Pogrebnoy

TPU Institute «Cybernetic centre»
E-mail: Sasha@ad.cctpu.edu.ru

The factors which determine the data content transferred between the stations of MP computer system have been revealed. The methods for construction of information graph of software load model and its cutset for forming the plan of station resource use are suggested. It is shown that criterion taken in the cutset problem corresponds to content minimization of the transferred data.

Introduction

When designing distributed real-time systems (RTS) the reduction of time expenditures for carrying out applied functions due to paralleling results in growth of data transmission volumes in computer system network. If the number of microprocessor stations in computer system is specified then the volume of transmitted data among the network stations depends on the following factors – conditions of control object operation, values of software module parameter, realizing applied functions of RTS and constituting the main part of system software load, software load station distribution.

The volume of data transmitted by network corresponds to certain time expenditures which increase completion time for carrying out the applied functions of software load. Time expenditures for data network transmission may be comparable with the time of processors operation at carrying out software load modules. Therefore, when designing RTS the analysis of the mentioned factors influence on the volume of network transmitted data is of great importance for making solutions at designing the structure of computer system network.

Modern RTS are as a rule designed in a network variant. The developer of distributed RTS when starting the project feels a great uncertainty with respect to the necessary set of various stations, their arrangement at object territory, network topology, being capable to transmit the data between the stations in time, software load station distribution. These problems are interconnected but there are no methods of their joint solution. Therefore the problem of determining a number of stations is solved independently after that the variant of network topology is accepted manually and then the problem of software load station distribution is solved more often by criteria of minimal network load [1, 2].

For many other supplements unbound with hard RTS design (automatization of design and scientific works, control of manufacturing etc.) the number of stations and network topology are predetermined by an object and accepted as the specified. In these cases the problem of software load distributing over network stations and obtaining the plan of resources use become the main ones. Many operations are known in which the problem of obtaining the plan of resources use is stated as nonlinear problem of mathematical programming with Boolean variables, for example [1]. Such problems may be solved using linearization technique but as it was mentioned in [1] for distributing 25 program modules over 15 stations the linear problem for 2500 variables and 8000 restrictions occurs.

Both problems concerning distributed RTS are stated in [2]. The problem of distributing the number of stations and their positioning in [2] is stated as the problem of linear programming but for the territory of stations positioning represented by a mesh with dimension 20×50 the number of variables achieves one million. The problem of obtaining the plan of resources use in [2] is stated as nonlinear problem of mathematical programming with Boolean variables for two variants of network topology.

Besides large dimension of the problems the main disadvantage of such approaches is in the fact that when solving the problem of obtaining the plan of resources use the network topology is accepted as specified. If the network topology is discounted by means of additional variables introduction then there is unacceptable problem for solving.

The significant disadvantage is also the fact that conditions of control object operation such as conditions of output data supply and correspondingly processes starting, conditions of updating output data state, selection rules of output data states at their transmitting among the processes are not taken into account. The listed conditions influence directly the volume of data transmitted in network.

Following the module design technology of distributed RTS [3] another approach to solving the concerned problems is suggested. In this approach the conditions of object operation are reflected to the model of software load and discounted at determination of data minimal volume transmitted among the stations. The problem of determining network topology is solved for the obtained data volume by the criteria of minimal time expenditures for data transmission among the network stations. This problem is especially important at designing hard RTS. To determine the number of stations the technique described in [4] is used. In this case all problem statements have the dimension acceptable for practical application.

The main purpose of investigations presented in the given paper is come to the fact that for the specified conditions of object operation, software load model and number of stations to represent the conditions of object operation to the model of software load and determine the data volume which is required for transmission among the stations at RTS operation. In this case the problem of distributing software load over the stations is stated as the problem of graph cutting on minimally connected subgraphs. In the given paper the quality estimation for the variants of cutting is determined and boundary of this estimation is introduced.

The investigations are carried out in the conditions when the model of software load is presented in the form of data flow graph (DFG) realizing applied functions of designed RTS [3]. In DEG there are two types of vertices – fragments of algorithms (modules) and variables (data). Arcs of the graph connect modules and data and indicate the relations of modules to one or another data consuming and forming.

Analysis of resource consuming conditions

When planning resources use the stations are considered as objects giving resources such as processors and storage. Objects of consuming these resources are the vertices of DEG – data and modules. Availability of resources and their consuming is estimated at a certain time interval called the simulation run. The value of time interval of simulation run is selected so that the period (regeneration cycle [1]) given for a single performance for any applied function of RTS can set in it integer times. For example if regeneration cycles for applied functions of RTS correspond to the segments $\{k_l\}=\{2,3,5,10\}$ then simulation run is accepted as equal to the least common denominator $t^*=30$ or to the value μt^*, $\mu=2,3,...$.

The problem of planning is in the fact that to indicate concrete stations which resources are used by all objects of DFG (data, modules, programs). Storage budget is used at data assignment to the station. Module in general case may be assigned to two stations – module operation, loading the processor of one station and module program occupying the storage of another one.

To simplify the problem let us consider that module operation and program always consume resources of only one station. In this case modules and data become the objects of resources consuming that is obviously reflected by DFG. It is accepted also that data as modules can not be separated in parts that is they are assigned fully to the stations resources. Thus, the problem of planning comes to the problem of module and data distributing over the stations.

For module and data distributing it is necessary to specify the volume of consumed resources for them. Let us estimate the processor resource by the time which is given for module performance in one simulation run. Appropriately for each module $f_m \in F$, $m=1,2,...,M$ the processor time consumed by module for one simulation run is $\tau_m \rho_m$. Here τ_m is the time of module performance, and ρ_m is the frequency of module f_m performance in one simulation run.

The storage of i station assigned for the DFG components storing is noted by P_i, $i=1,2,...,n$. Input and output data, module programs and intermediate data formed and consumed by modules are in storage. Input and output data $d_q \in D$,, $q=1,2,...,Q$ require storage p_q for placing and updated according to the receipt and regeneration cycle. Module programs require storage p_m and occupy it during the whole simulation run.

Intermediate data are divided into types. Datum $d_q \in D$ the modes of which are updated after each performance of producer module and require storage at the rate of p_q are referred to the first type. The second type corresponds to the situation when several modes of datum $d_q \in D$ obtained according to the specified rule of mode attachment should be stored [3]. The presence of data user with a certain set of modes for a module at input assumes the possibility of selection of certain modes from this set. When developing DFG the rules of updating with modes attachment should be coordinated with the rules of selection. The rules of selection often dictate the necessity of using the certain rules of attachment. The situation when the number of attached stations exceeds the number of modes obtained by the module per one simulation run is also possible. In all these cases datum $d_q \in D$ for which the rule of attachment b_q modes is determined requires storage in the rate of $b_q p_q$.

Construction of information graph for software load model

The volume of data transmitted by DFG arcs (d_q, f_m) and (d_m, f_q) is computed for one simulation run and accepted as arcs weights of information graph. For solving the problem of module and data distributing it does not matter in what direction the data are transmitted to the module or from it. Therefore, on the basis of DFG the information graph with the vertex matrix $A=\|a_{qm}\|_{Q\times M}$,

a_{qm}=1, may be constructed if the datum d_q is incident to the module f_m, a_{qm}=0, if not.

The weight r_{qm} of the arc (d_q,f_m) depends on storage p_q rate occupied by the datum d_q, frequency of datum d_q transmission over the arc (d_q,f_m) and number of attached modes. The frequency of datum transmission over the arc (d_q,f_m), incident to the module f_m equals to the frequency of this module ρ_m performance in simulation run. The rules of attachment and selection define the number of modes c_{qm} of the datum d_q transmitted over the arc (d_q,f_m).

On the basis of the matrix A, values $p_q\rho_m$ numbers of transmitted modes c_{qm} the weight matrix is constructed $R=\|r_{qm}\|_{Q\times M}$. Its elements are determined by the expression

$$r_{qm} = a_{qm} c_{qm} p_q \rho_m . \quad (1)$$

The example of DFG containing 5 modules and 9 data is shown in Fig. 1. In DFG data d_q are numbered in circles, f_m modules numbers are put down by the planks. The number at the arc defines the frequency of data transmitted over it per one simulation run. Near input data the conditions of their receipt are specified for example Ц (2). Here in brackets the number of receipt and starting cycles steps of a proper process is specified. Numbers arranged near circles specify the number of regeneration cycles steps.

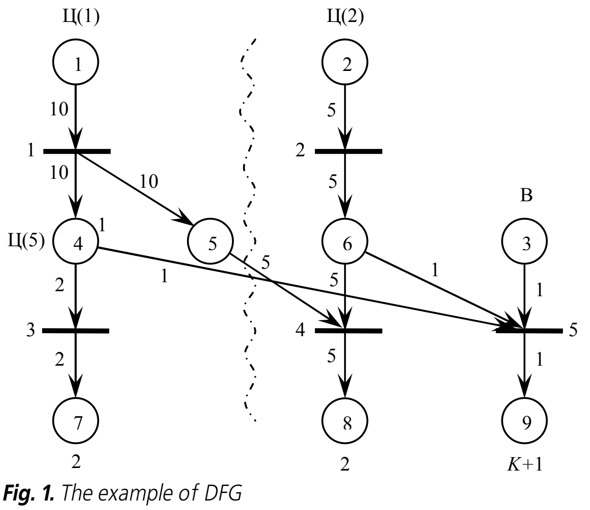

Fig. 1. The example of DFG

DFG in Fig. 1 contains 4 processes [3]. Three of them are started cyclically by the conditions of receipt of Ц(1), Ц(5) and Ц(2) and corresponding regeneration cycles: 1, 5, 2. The forth process is started by signal B with specified expectancy of receipt and forms the output signal in group k–1. It follows from Fig. 1 that the necessity of storage and transmission of more than one data mode appears in two cases. Module f_3 is started after module f_1 has being functioned 5 times. By this moment 5 modes of datum d_4 which enters the output of module f_3 will be obtained. Similarly module f_4 is started after 2 modes d_5 have being obtained. Therefore, in matrix C the elements of which determine the number of transmitted modes, Fig. 2, it is reflected that all data are transmitted by one mode excluding two stated cases. One of five modes enters to the input of module f_5 from datum d_4 that is determined by the rules of selection.

$C=$

	1	2	3	4	5	p_q
1	1					2
2		1				3
3					1	1
4	1		5		1	2
5	1			2		4
6		1		1	1	2
7			1			1
8				1		1
9					1	1
ρ_m =	10	5	2	5	1	

$R=$

	1	2	3	4	5
1	20				
2		15			
3					1
4	20		20		2
5	40			40	
6		10		10	2
7			2		
8				2	
9					1

Fig . 2. *The example of matrices C and R construction*

Intermediate datum d_6 refers to the first type, that is, it is regenerated after module f_2 functioning. The dimensions of data p_q are given to the right of matrix C and the data of modules performance frequency computed in terms of simulation run accepted as equal tacts are given below. The elements of matrix R, Fig. 2, are determined by the expression (1).

Forming the plan of station resources use

To solve the problem of modules and data station resource distribution we have informational graph form of software load model. For this model the volumes of consumption by modules of processor time in the rate of $\rho_m\tau_m$, storage for data storing in the rate of $b_q p_q$ and module programs in the rate of p_m as well as data r_{qm} volumes transmitted over the graphs arcs per one simulation run are determined. The performance criterion of problem solving is finding the variant of module and data station distribution delivering minimal total data volume transmitted in local network. In terms of titled criterion at solving the problem it is important that modules and data among which a great volume of information is transmitted were distributed to one station. Therefore, problem solution should be so that total data volume transmitted over arcs connecting modules and data arranged in different stations was minimal. Such solution corresponds to the known problem of graph cutting on minimally connected parts [5].

In respect to the considered problem of modules and data distribution we have the information graph for the model of software load in the form of bipartite weighted graph $G=(D,F,R)$, where D is the set of data vertices $D=\{d_q\}$ with indication of dimension of required storage $b_q p_q$ for each $d_q \in D$; F is the set of module vertices $F=\{f_m\}$ with indication of value of consumed processor time $\rho_m\tau_m$ for each $f_m \in F$; R is the matrix of transmitted data volumes among graph vertices.

Let us denote the sum of edge attachment weights S_{ij} of the parts $G_i=(D_i,F_i,S_i)$ and $G_j=(D_j,F_j,S_j)$ of the graph G by the value $r_{ij} = \sum_{s_{qm} \in S_{ij}} r_{qm}$. Here s_{qm} is the rib of graph G, connecting vertices d_q and f_m; S_{ij} is the set of ribs connecting vertices one of which belongs to the part G_i, and the second one belongs to the part G_j. The size of parts G_j is determined by station resources according to the storage P_i and processor time T_i, which can not exceed the simulation run. By the agreed notations the problem of cutting graph G per n parts G_i, i=1,2,...,n is written as:

$$\min r = \sum_{i=1}^{n}\sum_{j=1}^{n} r_{ij}; \qquad (2)$$

$$\sum_{d_q \in D_i} b_q p_q \le P_i, \quad i = 1,2,...,n; \qquad (3)$$

$$\sum_{f_m \in F_i} \rho_m \tau_m \le T_i, \quad i = 1,2,...,n. \qquad (4)$$

Graph G is cut per n parts by the number of computer system stations. In this case the conditions should be fulfilled:

$$\sum_{d_q \in D} b_q p_q \le \sum_{i=1}^{n} P_i, \quad \sum_{f_m \in F} \rho_m \tau_m \le \sum_{i=1}^{n} T_i. \qquad (5)$$

Fulfillment of ratios (5) stipulates the possibility of solving the problem (2)–(4) that is computer system resources should be enough for distributing all graph vertices subject to the values by storage and processor time parameters. As a rule values P_i and T_i are specified with a certain assurance factor.

Estimation of data volume transmitted in the network for the accepted plan of resources use

The presence of matrix R allows estimating the solutions of the problem on graph G cutting that corresponds also to the estimating the solution of the problem on modules and data station distribution. Let us consider the distribution problem as the partition problem of the set V vertices of the information graph G into subsets of vertices V_i. Let us denote the variant of partition w among the variety of possible variants W by the combination $\{V_i\}_w$ of the vertices sets V_i. The set of ribs $S_w=\cup S_{ij}$ corresponds to the variant $w\in W$ $S_w=\cup S_{ij}$, where S_{ij} is the set of ribs connecting vertices from sets V_i and V_j of partition $\{V_i\}_w$. Thus, the total volume of the data transmitted over the network per one simulation run for the variant of partition $\{V_i\}_w$, is amounted to r_w, $r_w = \sum_{s_{qm} \in S_w} r_{qm}$. We obtain value r_w for the variant of modules and data two stations distributions for the example of DFG (Fig. 1). Let us use the rule of selecting vertex d_q with maximal sum weight of incidence ribs r_q^*, $r_q^* = \max_q \sum_{m=1}^{M} r_{qm}$, as the initial vertex for set formation.

For our example r_q^* corresponds to the vertex d_5 that is the sum of weights of the 5th line of matrix R (Fig. 2). Value r_5^*=80 units of volume. Therefore vertex d_5 and vertices (modules) f_1 and f_4 connected with it are included into the formed set that is distributed to one station. Vertices d_1,d_6,d_8 are included in the same set. Sets V_1 and V_2 are connected by ribs $S_w=S_{1,2}=\{(d_4,f_1),(d_6,f_5),(d_6,f_2)\}$ at such modules and data distribution. Therefore, the volume of transmitted data r_w amounts to 32 units. Let us observe that rather obvious variant of partition shown in Fig. 1 includes two ribs $S'_w=S'_{1,2}=\{(d_5,f_4),(d_4,f_5)\}$ and gives the estimation $r'_w=r'_{1,2}$=42 units. This estimation is considerably worth than the estimation r_w=32 units.

Let us introduce the estimation of sets compactness for estimating vertex partition of information graph $G=(V,R)$ into subsets V_j, j=1,2,...,n. Let us estimate compactness of subset V_j by value R_j equal to the sum of ribs weights connecting vertices of this subset. Then value R_w, $R_w = \sum_{j=1}^{n} R_j$ may be used for estimating compactness of partition variant $\{V_i\}_w$. Estimations R_w and r_w of partition $\{V_i\}_w$ are interconnected – increasing estimation R_w corresponds to decreasing estimation r_w.

Let us name the variant of partition $\{V_j^*\}_w$ which corresponds to the maximal estimation R_w^* by compact partition (C-partition). To obtain partitions the algorithm proposed in [4] may be used. The given algorithms allow obtaining the partitions $\{V_i\}_w$ with local optimum of value R_w. Such partitions are named local compact partitions (LC-partitions). Therefore when using the given algorithms it is very important to have the opportunity to estimate the closeness of the obtained LC-partition to C-partition. For this purpose we propose to enter a certain compactness bound R_0 which ideally may be obtained by C-partition. Such bound is calculated in terms of supposition about the fact that each vertex $v_i\in V$, i=1,2,...,z in ideal partition occurs in set V_j^* with maximum possible compact estimation. According to this supposition let us form the set V_i, $|V_i|=|V_j^*|$ with maximum possible compact bound R_i for each vertex $v_i\in V$. Then the compactness bound R_0 is determined by the expression:

$$R_0 = \frac{1}{\mu_j}\sum_{i=1}^{z} R_i, \quad \mu_j = |V_j| = \text{const}.$$

Bound R_0 is the upper one for estimating R_w^* of C-partition that is among the bounds the following condition should be fulfilled

$$R_w^* \le R_0. \qquad (6)$$

In ideal case when the above mentioned supposition is fulfilled the given bounds may coincide. It means that the ideal C-partition occurs. It is characterized by the fact that it is impossible to construct sets $V_i\neq V_j^*$, $v_i\in V_i$, $|V_i|=|V_j^*|$ with compactness bound better than for set V_j^* for the vertices $v_i\in V_j^*$.

It is not difficult to proof the condition (6). Let us use Fig. 3 for this purpose. One of the sets V_j^* of C-partition $\{V_j^*\}_w$ containing 5 vertices, μ_j=5 is presented in Fig. 3.

Sets V_i which are conventionally showern in Fig. 3 by the curved lines passing through the corresponding vertices are constructed for each vertex. Let us denote the estimation of set V_i compactness by the value R_{ij}. Estimation R_{ij} corresponds to the best compactness estimation of set V_i including vertex v_i, that is $R_{ij}\ge R_j^*$. It follows from the fact that estimation R_{ij} can not be less than as R_j^* when forming set V_i there is always a possibility to form $V_i=V_j^*$ and then the estimation $R_{ij}=R_j^*$.

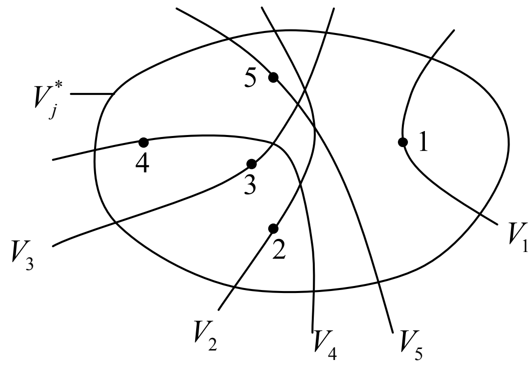

***Fig. 3**. Illustration to the proof of condition (6)*

It follows from Fig. 3 that all $V_i \neq V_j^*$ therefore, estimations R_{ij} exceed R_j^*. It follows from this that five sets averaged estimation also exceeds estimation R_j^* that is $\frac{1}{5}\sum_{i=1}^{5} R_{ij} \geq R_j^*$. Averaged estimations for other sets V_j^* of C-partition $\{V_j^*\}_w$ are obtained by the similar expression $\frac{1}{\mu_j}\sum_{i=1}^{\mu_j} R_{ij} \geq R_j^*$. Then for all sets V_j^*, $\sum_{j=1}^{n}(\frac{1}{\mu_j}\sum_{i=1}^{\mu_j} R_{ij}) \geq \sum_{j=1}^{n} R_j^*$ may be written. Taking into account that $n\mu_j=z$ we obtain the condition (6),

$$\frac{1}{\mu_j}\sum_{j=1}^{n}\sum_{i=1}^{\mu_j} R_{ij} = \frac{1}{\mu_j}\sum_{i=1}^{z} R_i = R_0 \geq R_w^*.$$

Conclusion

Information graph obtained as a result of analyzing the conditions of object functioning gives full idea about data transmission among graph vertices in one simulation run. It allows obtaining the estimations on network load of computer system depending on plans of resources use which are determined by the variants of vertex partition of information graph. Presence of such estimations allows making decisions on correcting the conditions of software load functioning (conditions of starting the processes, data regeneration and selection) and structure of computer system network.

The possibility of obtaining the bound has a great cognitive meaning at investigating C-partitions and practical meaning at solving the problem of minimizing the data transmission volumes. So after LC-partition has being obtained its bound departure and C-partition departure with a certain inaccuracy may be estimated. If the departure is great then it makes sense to continue searching another LC-partition with better estimation. Besides, if the estimation of this partition coincides with the bound then the ideal C-partition occurs.

REFERENCES

1. Chu W.W., Holloway L.J., Lan M.T., Efe K. Task allocation in distributed data processing // IEEE Trans. on Computers. – 1980. – V. 13. – № 11. – P. 57–69.
2. Shenbort I.M., Aliev V.M. Designing computer systems of distributed industrial control. – Moscow: Energoatomizdat, 1989. – 88 p.
3. Pogrebnoy V.K. Real-time systems. Modeling and computer-aided design. – Tomsk: TPU Press, 2006. – 208 p.
4. Pogrebnoy A.V. Determination of a number and topology of positioning stations of multiprocessor system // Bulletin of the Tomsk Polytechnic University. – 2006. – V. 309. – № 7. – P. 160–164.
5. Panteleev A.V., Letova T.A. Optimization techniques in examples and problems. – Moscow: Vysshaiya shkola, 2002. – 544 p.

Received on 11.12.2006