TASK ASSIGNMENT FOR SCHEDULING JOBS AND RESOURCES IN PARALLEL DISTRIBUTED SYSTEMS

PHAM HONG HANH, VALERY SIMONENKO

Abstract. In managing multiprocessing of parallel distributed systems the central issue is the scheduling of jobs and resources in the optimum way. This paper describes a new approach for the solution of this problem. The proposed approach allows us to create an algorithm that adapts to any kind of systems constraints and the optimization criterion as well. The key idea in our approach is to divide the process of the scheduling into preliminary analyzing initial data and finding the solution with the support of the results of this analysis. This algorithm for analyzing is built on the principle of step-by-step forming and is called Adaptive Multi-analyzing Algorithm (AMA). The proposed algorithm is based on our development of the Malgrange method for task assignment. The results of our investigation are presented in a system of theorems which are shown in this paper. The time complexity of the proposed algorithm varies from $O[N\log(N) + E]$ to less time, depending on the characters of the initial data of the systems analyzed. The adding of this algorithm based on our theoretical system for analyzing initial data allow us to decrease the whole time complexity for finding the schedule of jobs and resources. These advances of AMA are shown theoretically by describing the analyzing process and through the results of experiments in the simulation multiprocessing systems as well.

I. INTRODUCTION

In recent years, parallel distributed systems have received increasing attention in the design of multiprocessing systems. A central issue for the multiprocessing is the scheduling of jobs and computer resources. Numerical algorithms, methods for scheduling of jobs and resources have already been provided by several resherechers. They are based on different approaches for optimizing a particular criterion for instance: run time, compiling time, processor balance, communication costs etc. The general problem is to schedule a stream of jobs and a system of resources so that each job can be executed by fit resource in the required time according to the optimization criterion. This problems is usually solved for meeting one of the constrains of job-stream features or the constraints of the resource system structure. In the first case, the features of jobs and the optimization criterion are given and we need to define the structure of the system of resources that is to fit to such the given data [16]. In the second case, the structure of the system is fixed and we

have to schedule the jobs so that they are optimized by some criterion executed by such a system [17].

This paper proposed a new general approach to state and solve the problem of scheduling jobs and resources throughout any parallel computer system and in any application problem, where the task assignment takes place. In this light, the problem of scheduling jobs and resources can be generally stated as the follows: Into a system of N resources a stream of M jobs is coming at a moment in time, all the requirements of the process including the criterion of optimization for these N resources and these M jobs are represented by K limits in two separate binary matrices: $T[i,j]$, $i = 1,..., K$, $j = 1,..., M$ for the job constraints and $R[i,j]$, $i = 1,..., K$, $j = 1,..., N$ for the resource constraints, where K, N, $M \in \mathbf{N}$; the requirement of the problem is to define a maximum number of assignments of jobs on resources $A\{(I_i, J_j)\}$, $I_i \in I = \{1, 2,..., M\}$ and $J_j \in J = \{1, 2,..., N\}$, $I_i \notin I \setminus I_i$ and $J_i \notin J \setminus J_j$ (that is only one job can be assigned on one resource and vice versa) so that $T[l, I_i] \approx R[l, J_j]$ $\forall l = 1,..., K$. For simplicity, from two matrices T and R we can form a new Boolean matrix called the matrix of connections $MC[i,j]$, $i = 1,..., M$, $j = 1,..., N$, which represents the fit of JOB(i) and RESOURCE(j) by all of K limits or the possibility of the assignments of these Job(i) and Resource(j). Then this problem becomes the problem of assignment in Combinatorial theory. We will study further with the fact that $M = N$ because it has been shown that the general case when $M \neq N$ could be changed into the case when $M = N$ using the Hungarian algorithm [5, p. 406].

Therefore the initial data for the problem of assignment can be represented in a binary Boolean matrix (Fig. 1a). and in common bipartite unweighted graph as well (Fig. 1b)

		RESOURCE					
P		1	2	3	4	5	6
R	1	1	0	1	0	1	0
O	2	0	1	1	1	1	1
C	3	1	1	0	0	1	0
E	4	1	1	1	0	1	0
S	5	1	1	0	0	1	0
S	6	0	0	1	1	0	1
E							
S							

Fig. 1a

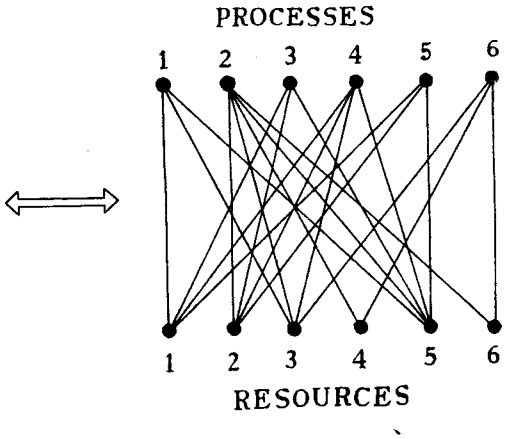

Fig. 1b

Because of the equivalence of these two types of presentation we can study further with the matrix form using some theorems of Graph theory.

II. TASK ASSIGNMENT

The task assignment has many application in many different areas. However, it has been shown that the given task is an NP-completeness task and the solution has exponential time complexity [20]. For parallel, distributed computer systems several approaches have been suggested [1-4, 17, 18, 19]. Most of them are based on heuristic or iterate approaches. All of them have great time complexity for getting at least one variant of maximum matching and the best algorithm by time complexity is proposed in [3].

In this paper we propose a new approach to solve the task assignment or scheduling jobs and resources in computing systems. The key idea is to divide the process of scheduling into preliminary analysis of the data and finding the solution with the help of the results of this analysis. It may be strange but our simulation results show us that the adding of the analysing data algorithm substantially decreases the total solution time in spite of which method for solving is used.

The proposed algorithm for analysing is based on the principle of step-by-step forming initial data and we call Adaptive Multi-analysing Algorithm (AMA).

III. TASK SCHEDULING

We have shown above that the key issue of scheduling jobs and resources in distributed, parallel processing computer systems is the task assignment. Now returning to the task scheduling, we determine this task as follows: For the system of N resource, which has to execute N jobs at a moment in time with all the kinds of requirements represented in Boolean matrix of connection $MC[i,j]$, $i, j = 1, 2, ..., N$, we have to define at least one job-resource schedule: $A = \{(I_i, J_j)\}$, $I_i \in I = \{1, 2, ..., N\}$, $J_j \in J == \{1, 2, ..., N\}$ so that $\forall I_i \notin I \setminus I_i$, $J_j \notin J \setminus J_j$ and $\forall (I_i, J_j) \in A : MC[I_i, J_j] = 1$. That means one resource can serve only one job at one moment of time, the process of serving one job can not be interrupted, every job has an individual characteristic and lay claim to the capture only one resource. The '1' in the matrix C means the pair of Job(i) and Resource(j) meet all K requirements of the given system and of the processing as well. The '0' means there is no such meeting [Fig. 2a).

For further performance of our proposed algorithm, we form the vector $VJ[i]$, $i = 1, ..., N$ for jobs and the vector $VR[j]$, $j = 1, ..., N$ for resources (Fig. 2b) as the follows:

$$VJ[i] = \sum_{j=1}^{N} MC[i,j], \; i = 1, ..., N.$$

This is called a ratio of claimant for Job(i).

$$VR[j] = \sum_{i=1}^{N} MC[i,j], \; j = 1, ..., N.$$

This is called a ratio of placement of Resource(j).

	1	2	3	4	5	6
1	1	0	1	0	1	0
2	0	1	1	1	1	1
3	1	0	1	0	1	0
4	1	1	1	1	0	1
5	1	0	1	0	1	0
6	1	0	1	0	1	0

Fig. 2a. Matrix of Connection

	VJ		VR
1	(3)	1	(5)
2	(5)	2	(2)
3	(3)	3	(6)
4	(5)	4	(2)
5	(3)	5	(5)
6	(3)	6	(2)

Fig. 2b. Vectors of ratios for Jobs and Resources

For solving the problem of job scheduling it is necessary to define the conditions of the possibility of its solution, that is, the full distribution of all jobs on resources in the given system. The process of scheduling jobs should be divided into two phases; the first phase is a quick analysis of the initial data and the results of this step is the recommendation that will be used in the solving process later. The second phase is finding the whole schedule of jobs and resources.

The step of analyzing by our proposet algorithm gives us a preliminary answer to the question of whwther or not for the solution existing of the scheduling problem.

For executing the second phase of problem we can use any well-known algorithm for maximum matching and the matrix after correction (MA) is the input data for this phase.

IV. ADAPTIVE MULTI-ANALYZING ALGORITHM

We suggest using out algorithm is suggested for finding a first approximation for the best method to solve the problem of scheduling. The analysis of the input data in carried out by several steps:

1. Preparing the initial data for analyzing.

2. Finding certain assignments of the schedule.

3. Correcting the initial data for excluding cases, when the process of scheduling has "conflicting points" or "not stable" points.

The first step is creating the matrix of connections (MC) which reflects the possibility of the matching of every job to the resources in the given system. In this step, two vectors VJ, VR of ratios for every job and every resource are created (Fig. 2a, Fig. 2b).

The second step is finding certain assignments of the schedule. For getting them, the following results are used:

Rule 1. (From the Consequence 4.1)

If the matrix of the connection C is a low-triangle matrix and the principal diagonal consists of '1's only, then the task assignment has only solution which is the very this diagonal. The process of analyzing must be interrupted.

Rule 2. (From the Consequence 1.2)

If in the matrix MC there is any Fan (according to the Consequence 1.2). the task scheduling has no whole solution and the process of analyzing must be interrupted.

Rule 3. (From the consequence 1.1)

Any single assignment in the matrix of connection (Fig. 1a, 1b) always participates in the solutions of maximum matching. They are called certain assignments of the schedule. These pairs of job(p)s and resource(q)s must be moved away from the matrix MC in the next steps.

Then the size of the given task may be decreased by the number of pairs found in rule 3. This means that the search of matching will continue from the new matrix, which has been reconstructed from the initial one.

Our experiments for our proposed algorithm have given us the following statistical results: When the ratio of fullness of '1's in the initial matrix MC called Rf is less than 18%, our proposed algorithm gives us the whole solution (that is the whole schedule of jobs) after the second step! When the ratio Rf is more than 18% and less than 50%, the size of the job scheduling's problem is decreased in the next steps of the given algorithm. The time complexity of the second step for finding the certain assignment of the schedule is $O[2N]$.

The third step is analyzing the initial data for confirming the availability of the solutions. Sometimes this step even gives us the whole solution to the job - resources scheduling problem. The aim of the analysis is to pick out the assignments which we do not need to do because it leads us to conflicting states. That is why we call these assignments as "Conflicting points" or "Unstable points".

Rule 4.

If after the equivalent transformation we obtain an MT with principal diagonal of '1's, then all the jobs can match their resources and the scheduling problem has

at least one solution corresponding to the principal diagonal. The process of analyzing is finished.

Rule 5. (From the Theorem 3)

If in the matrix C after transformation we can define any submatrix MM of '0's as in the Consequence 2.1. with the size $S * T$, where $S + T > N$ then the given task has no whole solution. The process of analysing must be interrupted.

Rule 6. (From the Theorem 3)

If in the matrix C after transformation we can define any submatrix MM of '0's as in the Consequence 3.1. with the size $S * T$, where $S + T = N$ then all the assignments: the '1's in the symmetrical polygon must be nullified because they are critical in the given initial data. The process of analysing must be continued with the new corrected matrix.

Rule 7. (From the consequence 4.1)

If in the part above the principle diagonal we can find any prominent polygon concerning the principal diagonal (Fig. 2b), then the '1's that concern this side are "unstable" points and the '1's in the symmetrical polygon must be nullified because they are "Conflicting points". Now the '1's in the principal diagonal become certain assignments and the jobs corresponding to those '1's, must be assigned only to the corresponding resources according to the rule 3. Then, after this exclusion which throw out these unstable assignments, a new corrected matrix MA is created. The process of analyzing must be now continued with the new corrected matrix.

For getting it, we use a special equivalent algorithm of transformation for MC. This algorithm of sorting is represented by changing the places of the rows so that the number of '1's in them increases and by changing the places of the columns so that the number of '1's in them decreases. This algorithm is based on the modification of the method for decomposition from an oriented graph to the maximum connected subgraph, which was supposed by Malgrange [5]. The next operations are based on the theorem of Frobenus-Kening [6] and Minc [7] about computing permanent $(0, 1)$ in matrices.

Therefore, in the first phase we can substantially decrease the scheduling time for the next phase by passing most of the assignments that are conflicting or unacceptable in the schedule.

V. THEORETICAL BASES FOR ANALYZING DATA

In this section we propose a system of the theorems and their consequences which are the base the building the rules used in previous section.

Theorem 1. (Single connection - Belonging)

In the matrix $MC[i,j]$, $i, j = 1,..., N$, *if* $\exists$ *such a single connection* (p,q):

$$MC[p,q] = 1 \text{ and}$$

$$MC[p,j] = 0 \ \forall j = 1,..., N \text{ and } j \neq q \qquad (1)$$

$$MC[i,q] = 0 \ \forall i = 1,..., N \text{ and } j \neq p \qquad (2)$$

Then this pair always participates in the assignment A, *which means* $(p,q) \in A$.

Proof. Using that $A\{(I_i, J_j)\}$ is an assignment for $2N$ vertices $I_i = 1,..., N$ and $J_j = 1,..., N$ we examine the assignment for $I_i = p$ and $J_j = q$. It is obvious that if (1) then $\exists A^* \in A : A^* = (p,x)$, $x \in \{1, 2,..., N\}$ and $MC[p,x] = 1$, from (1) we receive that $x = q$, which means $(p,q) \in A$.

By the same way if (2) then $\exists A^{**} \in A : A^{**} = (x,q)$, $x = 1,..., N$ and $MC[x,q] = 1$, and from (2) we get that $x = p$ and $(p,q) \in A$. Therefore (p,q) always belongs to A.

Consequence 1.1. *If one solution* A^* *of the task assignment can be divided into two parts the single assignments* (p,q) *and the others, then the remaining part (the others) can be found in the new matrix of connections* MC^* *after deleting the rows and columns corresponding to these single assignments.*

Proof. From the theorem about independent excluding of single edges [3] we have the following: if the solution $A = \{(p,q), B\}$ of N pairs, $B = \{(b,d)\}$ of $N-1$ pairs of vertices for the bipartite graph $G = (V,E)$ of $2N$ vertices then B can be found in the subgraph $G' = (V',E')$ of $2N-2$ vertices after deleting (p,q) and all connected edges with these p, q vertices. Translating the representation form of the initial data from a bipartite graph (Fig. 1b) to a matrix (Fig. 1a) and by Theorem 1 recall that the single assignments always participate in the solutions of the given task, it is obvious that the Consequence 1.1 takes place.

Consequence 1.2. *In the matrix* $MC[i,j]$, $i, j = 1,..., N$ *if* $\exists$ *such a Fan:*

$FA = \{(P_i,q)\}$, $P_i \in \{1,..., N\}$, $i = 1,..., F$, $2 \leq F \leq N$, *where* $MC[P_i,q] = 1$ *but* $MC[P_i,Q_j] = 0 \ \forall Q_j = 1,..., N$ *and* $Q_j \neq q$, *(3)*

or $FA = \{(p,Q_j)\}$, $Q_j \in \{1,..., N\}$, $j = 1,..., F$, $2 \leq F \leq N$, *where* $MC[p,Q_j] = 1$ *but* $MC[P_i,Q_j] = 0 \ \forall P_i = 1,..., N$ *and* $P_i \neq p$, *(4)*

then the task assignment has no whole solution A *and the size of maximum matching* $Z_a < N - F + 1$.

Note that it is possible that several Fans can exist at the same time and this consequence is true for every one of them.

Proof. We propose that there is one whole solution $A = \{(I_i, J_j)\}$. If (3) then that means for the vertices P_i, $i = 1,..., F$ we have F pairs-assignments Q_i', $I = 1,..., F$

so that $MC[P_i, Q_i'] = 1$. But from (3) we get $MC[P_i, q] = 1$ and $MC[P_i, Q_j] = 0\ \forall Q_i = 1, ..., N$ and $Q_i \neq q$. That means $Q_i' = q, \forall i = 1, ..., F$. But because $2 \leq F \leq N$ this contradicts the propose about the solution $A\{(I_i, J_j)\} : \forall I_i \notin I \setminus I_i$ and $J_j \notin J \setminus J_j$. If (4) we get the same result in similar way of considering. That is why such whole solution can not exist.

Now, it is obvious that for such MC with such FAN, the size Z_a can not be more than or equal to $N - F + 1$ because as is shown above, all F vertices P_i (or Q_j) can not be matching than.

Theorem 2. (Sub matrix of '0's in MT with the size $S * T$, $S + T > N$)

In the matrix $MT[i, j]$, $i, j = 1, ..., N$, if we can define such a submatrix $MM[i, j]$, $i = (N - S + 1), ..., N$, $j = T, ..., N$, where $S + T > N$ and $\forall MM[i, j] = 0$, then problem of assignment has no whole solution.

RESOURCES

		n-s				*s*		
P	1*	1	1	0	0	0	0	
R	1	1*	1	0	0	0	0	
O	1	1	1*	0	0	0	0	*t*
C	1	1	1	0*	0	0	0	
E	1	1	1	0	0*	0	0	
S	1	1	1	0	1	1*	1	
S	1	1	1	1	1	1	1*	*n-t*
E		*n-s*				*s*		
S								

Fig. 3. Submatrix of '0's - no solution

Proof. We propose that one whole solution $A * \{(I_i, J_j)\}$ exists. That means for each job of $I_i^* \in \{1, ..., T\}$ we have one corresponding pair (I_i^*, J_j^*) where $J_j^* = 1, ..., N$ so that $MT(I_i^*, J_j^*) = 1$ and $J_j^* \notin \{1, ..., N\} \setminus J_j^*$. It is obvious that for such T jobs I_i^* we need at least T resources J_j^*, because (I_i^*, J_j^*) participate in A by pairs. Now, from the claim that $S + T > N$ we have $T > N - S$, that means $\{J_j^*\}$, $J_j = \{1, ..., (N - S)\}$ is not enough for T jobs I_i^*. Then $\exists$ at least one $J_j^* : J_j^* = (N - S + 1), ..., N$ and $(I_i^*, J_j^*) \in A$ consequently $MM[I_i^*, J_j^*] = 1$, where $J_j = (N - S + 1), ..., N$, $I_i^* = 1, ..., T$ but this contradicts the condition about MM.

Therefore, suc' a whole solution A^* can not exist.

Consequence 2.1. *In the matrix* $MT[i,j]$, $i, j = 1,..., N$ *if we can define some such* $MM(h)$ *as* MM *in the Theorem 2 with size* $S_h^* T_h$, $h = 1,..., H$ *then the maximum matching has the size* $Z_a < \max\{(2N - S_h - T_h)\}$.

Proof. The considering to proof this consequence is similar to the proof of Consequence 1.2, with the fact is the theorem 2 that $(S_h + T_h - N)$ is not enough. Theorem the maximum size of possible matchings is $N - (S_h + T_h - N)$ or $2N - S_h - T_h$.

Consequence 2.2. *Theorem 2 is true not only for* MM *that lies in the top-right-hand corner of the matrix* MC *but also for any position of* MM *in* MC.

Proof. From [3]: an equivalent transformation of a matrix always exists so that MM can be moved from and position to the top-right hand corner of the matrix MC.

Theorem 3. (Submatrix of '0's in $MT(S * T)$, $S + T = N$)

In the matrix $MT[i,j]$, $i, j = 1,..., N$ *if we can define such a submatrix* $MN[i,j]$, $i = (N-S+1),..., N$, $j = T,..., N$, *where* $S+T = N$ *and* $\forall MN[i,j] = 0$, *then for* $I_i \in \{1,..., (N - S)\}$, $J_j \in \{(T + 1),..., N\} : \forall(I_i, J_j) \notin A$ *(if such a solution* A *exists). All of this* $(I_i, J_j) \notin A$ *create a submatrix called conflicting zone or conflicting points and must be excluded from the matrix* MT.

(a) Matrix of Connections

1	1	0	1	0	1	0
2	0	1	1	1	1	1
3	1	0	1	1	1	0
4	1	1	1	1	0	1
5	1	0	1	1	1	0
6	1	0	1	1	1	0
	1	**2**	**3**	**4**	**5**	**6**

(b) Matrix after Transformation

6	1*	1	1	1	0	0
1	1	1*	1	0	0	0
5	1	1	1*	1	0	0
3	1	1	1	1*	0	0
2	1	0	1	1	1*	1
4	1	1	0	1	1	1*
	3	**1**	**5**	**4**	**2**	**6**

(c) Matrix after Correction

1*	1	1	1	0	0
1	1*	1	0	0	0
1	1	1*	1	0	0
1	1	1	1*	0	0
0	0	0	0	1*	1
0	0	0	0	1	1*

Conflicting Zone

Fig. 4

Proof. Propose that there is one such a $MT[i^*, j^*] = 1 \in A$, where $i^* \in \}1,..., (N - S)\}$, $j^* \in \{(T + 1),..., N\}$. Then we examine the set of resource $J^* = \{1,..., (N-S)\} \subset J$, I^* has $(T+1)$ has elements and J^* has $(N-S)$ elements. In the solution A for those I^* we have $(T + 1)$ pairs and $(T + 1)$ corresponding resources J_j'. From the claim that $S + T = N$ we have $S + T + 1 > N$ that means $T+1 > N-S$ or $[J'] > [J^*]$ and there is at least one $J_j' \in J' : J_j \notin J^*$, but $J_j' \in J$ and recall that $J = 1,..., N$, $J^* = 1,..., (N - S)$ so $J_j' \in \{(N - S + 1),..., N\}$

and $MT[I_i', J_j'] = 1$ because $(I_i', J_j') \in A$. But this is contrary to the conditions of MN (such a $MT[I_i', J_j'] \in MN$).

In the case $MT[I_i', J_j'] = 0$ it is obvious that $MT[I_i', J_j'] \notin A$, therefore the Theorem has proofed the theorem.

Theorem 4. (Prominent polygonal)

In the matrix $MT[i,j]$, $i, j = 1, ..., N$ *if we can define such a prominent polygonal e.g. such a* $MM[i,j] : j = (N-S), ..., N$, $i = 1, ..., T$, $S + T = N + 1$ *so that* $\forall MM[i,j] = 0$ *except* $MM[T, (N-S)] = 1$ *then the symmetrical prominent polygonal consisting of the submatrix* $MM'[i,j]$, $i = 1, ..., (N - S + 1)$, $j = (T - 1), ..., N$ *is a Conflicting Zone because* $\forall (i,j) \notin A$ *except* $MM[T, (N-S)]$.

(a) Matrix of Connections

1	1	0	1	0	1	0
2	0	1	1	1	1	1
3	1	0	1	1	1	0
4	1	1	1	1	0	1
5	1	0	1	0	1	0
6	1	0	1	0	1	0
	1	**2**	**3**	**4**	**5**	**6**

(b) Matrix after Transformation

6	1*	1	1	0	0	0
1	1	1*	1	0	0	0
5	1	1	1*	0	0	0
3	1	1	1	1*	0	0
2	1	0	1	1	1*	1
4	1	1	0	1	1	1*
	3	**1**	**5**	**4**	**2**	**6**

(c) Matrix after Correction

1*	1	1	0	0	0
1	1*	1	0	0	0
1	1	1*	0	0	0
0	0	0	1*	0	0
0	0	0	0	1*	1
0	0	0	0	1	1*

Unstable Point

Conflicting Zone

Fig. 5

Proof. It is obvicus than $PP = MM_1[i_1, j_1] + MM_2[i_2, j_2]$, where $i_1 = (N - S), ..., N$, $j_1 = 1, ..., (T + 1)$, $i_2 = (N - S - 1), ..., N$, and $j_2 = 1, ..., T$. Put $T_1 = T - 1$ and $S_1 - S$, put $T_2 = T$ and $S_1 = S + 1$, then $[MM_1] = (S_1 * T_1)$ and $[MM_2] = (S_2 * T_2)$ where $S_1 + T1 = N$ and $S_2 + T_2 = N$. By Theorem 3: $\forall (I_i^*, J_j^*) \notin A$ where $I_1^* \in \{1, ..., (N - S_1)\}$ and $J_1^* \in \{(T_1 + 1), ..., N\}$; $\forall (i_2^*, j_2^*) \notin A$ where $i_2^* \in \{1, ..., (N - S_2)\}$ and $j_2^* \{(T_2 + 1), ..., N\}$ at the same time. Remember that when $T_1 = T - 1$, $S_1 = S$, $T_2 = T$, $S_2 = S + 1$, we get the following $\forall (i,j) \notin A$, for $i = (T-1), ..., N$ and $j = 1, ..., (N - S + 1)$ except $((N - S), T)$.

Consequence 4.1. *If there is one or several prominent polygonal consisting of several* MM *of '0's by the way in Theorem 4, then the correspondingly symmetrical* $PP's$ *are conflicting zones and must be nullified.*

Proof. It is obvi us that such prominent polygonal can be divided into several

pairs such MM_1, MM_2. As is in the proof of the Theorem 4 and according to this proof Theorem 4 is true for each of pairs. Therefore the consequence is proved.

Consequence 4.2. *If there is prominent polygonal consisting of $N-1$ different MM of '0's e.g. all the part of the matrix above the principal diagonal are '0's, then all the part of the matrix below the principal diagonal must be nullified.*

(a) Matrix of Connections

1	0	0	1	0	1	0
2	0	1	1	1	1	1
3	1	0	1	1	1	0
4	1	0	1	1	1	1
5	1	0	1	0	1	0
6	0	0	1	0	0	0
	1	**2**	**3**	**4**	**5**	**6**

(b) Matrix after Transformation

6	1*	0	0	0	0	0
1	1	1*	0	0	0	0
5	1	1	1*	0	0	0
3	1	1	1	1*	0	0
4	1	0	1	1	1*	0
2	1	1	0	1	1	1*
	3	**5**	**1**	**4**	**6**	**2**

(c) Matrix after Correction

1*	0	0	0	0	0
0	1*	0	0	0	0
0	0	1*	0	0	0
0	0	0	1*	0	0
0	0	0	0	1*	0
0	0	0	0	0	1*

Conflicting Zone

Fig. 6

Proof. We can prove this consequence complete similar to the above proof when the number of pairs MM_1 and MM_2 in the prominent polygonal is $N-1$.

VI. COMPUTING TIME COMPLEXITY

The time complexity of the adaptive multi-analyzing algorithm for premilinary processing and analyzing of the initial data consists of the time complexities of every step. This time complexity depends on the number of edges in the graph or of the number of the '1's in MC e.g. the value of Rf.

For the first step, the time complexity for creating the matrix MC is $O[E]$ and for creating two vectors V_J and V_R is $O[2N]$. Therefore the time complexity for this step is equal to $O[E+2N]$.

For the second step, the searching for certain assignments requires $O[N]$ and the reforming matrix MC in the case where such assignment exist it is $O[N+2N]$. If the rule 1 take place then the whole solution (whole schedule) is found for $O[E+2N]+O[3N]=O[E+5N]$ time complexity. If the rule 2 or the rule 3 take place we have defined that the given problem has no whole schedule for such a time complexity.

For the third step the time complexity of the special sorting, analyzing and correcting is $O[2N^*\log N + E/2 + E/2]$. If we use the Malgrange algorithm - the best algorithm for sorting [5] then the time complexity will be $O[8N^2 + E/2 + E/2]$.

Therefore, general time complexity for AMA is $O[3E + 2N^*(3 + \log N)]$. When N is a large number this time complexity can be represented in just $O[E + N\log N]$.

VII. CONCLUSION

In this study, we have provided a new approach for solving the task assignment in scheduling jobs and resource in parallel distributed multiprocessing systems. The key idea in our approach is the divide the process of the scheduling into preliminary analysing initial data and finding the solution with the help of the results of our analysis. The time complexity of our proposed algorithm varies from $O[E + N\log N]$ to less, depending on the characters of the initial data of the process as is shown above. However, the experiments with simulation multiprocessing systems using our algorithm show that when Rf of MC is less than 50% (that is the number of separates assignment is less than half of the possible maximum number), these algorithms provide the whole schedule. Then the time complexity of our algorithm is the same time complexity of the whole problem of job scheduling. Note that the well-known algorithms have good results when Rf of MC$>$ 50% but when Rf of $MC < 50\%$ they have a large solution time. Even when the problem has no solution it takes longer time to find it out using other algorithms then our. The proposed approach allows us to create an algorithm that adapts to any kind of the system constraints and the optimization criterion as well. Note that for just solving the task assignment not all the theoretical base for our proposed algorithm is used. However, using Consequence 1.2, and Consequence 2.1, about the size of maximum matching this can provide more help the further process of scheduling. Therefore the new proposed approach really gives us an instrument for solving this intractability of the scheduling problem in parallel processing, particular in real-time systems where the main criterion of optimization is the solution time.

REFERENCES

1. B. Lipsky, *Combinatorics for programmers*, Mir, M., 1988, 212.
2. X. Papadimitry, K. Stayglitsh, *Combinatory optiminzation algorithm and complexity*, Mir, M., 1985, 512.
3. H. Alt, N. Blum, *Computing a maximum carnality matching in bipartite graph in time* $O[n^{1.5}\sqrt{m/\log(n)}]$, Elsevier Science Publishers B. V. (North-Holland), 0020 - 0190, 1991.
4. J. Cheriyan, T. Hagerup, and K. Mehlhorn, *Can a maximum flow be computed in O(nm) time?*, it was presented at the 17th ICALP, 1990.

5. A. Kaufmann, *Introduction a la combinatorique en vue des applications*, Dunod, Paris, 1968), 477.
6. H. Minc, *Permanents*, Addison-Wesley Publishing Company, Inc., 1978, p. 254.
7. M. Marcus, H. Minc, *Modern University Algebra*, New York, 1966, 350.
8. Berge C., *Theorie des graphes et ses applications*, Dunod, Paris, 1958.
9. Dahlhaus Elias, *Kerpinski Merk. Parallel construction of perfect matchings and Hamilonian cycles on dense graphs*, Theor. Comput. Sci., N. 2-3 (1988), 121 - 136.
10. Kedem Zri M., Palem Krishma V., *Optimal parallel algorithms for forest and term matching*, Theor. Comput. Sci., **2** (1993), 245 - 264.
11. Hopcroft J. E. and Karp R. M., *An algorithm for maximum matchings in birpartie graphs*, SIAM J. Comput., **2** (4) (1973).
12. Yu Ming-Shing, Yang Cheng-Hsing, *A linear time algorithm for the maximum matching problem on cographs*, BIT (Dan.), No. 3, 1993, 420 - 433, 225 - 231.
13. Ford L. R., Fulkerson D. R., *Flows in networks. Princeton university Press*, Princeton, N. J., 1962.
14. Glodberg A. V. and Tarjan R. E., *A new approach to the maximum flow problem*, J. ACM, **35** (1988), 921 - 940.
15. 1968 (1980).
16. Phillip Krueger, Ten-Hwang Lai, and Vibha A. Dixit-Radiya, *Job scheduling is more important then processor allocation for hypercube computers*, IEEE Trans. Parallel distrib. Syst., **5** (1994), 488 - 496.
17. Shen Shen Wu and David Sweeting, *Heuristic algorithms for task assignment and scheduling in a processor network*, Parallel Computing, **20** (1994) 1 - 14.
18. T. L. Casavant and J. G. Kuhl, *A taxonomy of scheduling in general-purpose distributed computing systems*, IEEE Trans. Comput., **C-30** (3) (1981), 207 - 214.
19. W. W. CHu and the others, *Task allocation in distributed data processing*, IEEE Comput., **13** (11) (1980), 57 - 69.
20. M. R. Garney and D. S. Johnson, *Computer and Intractability - A guide to the Theory of NP - completeness*, Freeman, New York, 1979.

Department of Computer Science
National University of Technical
Kiev, Ukraine.

Nhận bài ngày 9-6-1995