Prace Naukowe Wyższej Szkoły Pedagogicznej w Częstochowie
Matematyka IV, Częstochowa 1998

The Procedures of the Assigvment of Tasks Distribution for Parallel Procrssors System

Adam Borowik, Henryk Piech

Abstract

In this paper one of possible optimum conditions distribution processes algorithmes with precised (limited) number of processors is described. This method accepts that realization lines of particular variavles with initial supplementary procedures application are specified earlier. Results are produced as a time-vectors of particular rules realization with determine each variable creation stages. In this algorithm criterions relavant with minimalization following rules expection of time which determine particular variable creation stages and also with conferment highest priorites lines which are characterized by langest realization time is regarded. Moreover in inbestigations is experimentated a simulation program of proposed algorithm.

Introduction

In this paper we propose to handle multiprocessor systems as a mass - serwice system about n- channels (n is equal with processars' mumber). The time t_{i_j} af the single problem is dimension that depends an calculation procedures contained in work - tasks, where letter i means a number of variable on the parallel processors' load map (according to the program) and letter to this variable calculations. Enclosed map is an Mustration, produced for description, of parallel realization with unlimited processors' number without regard the time differentation relevant to particulars problems realization.

In illustration 1. proposed model of mass - service system is represented on the assumption that processors' number (n) is limited and the problem's realization time is differentiated.

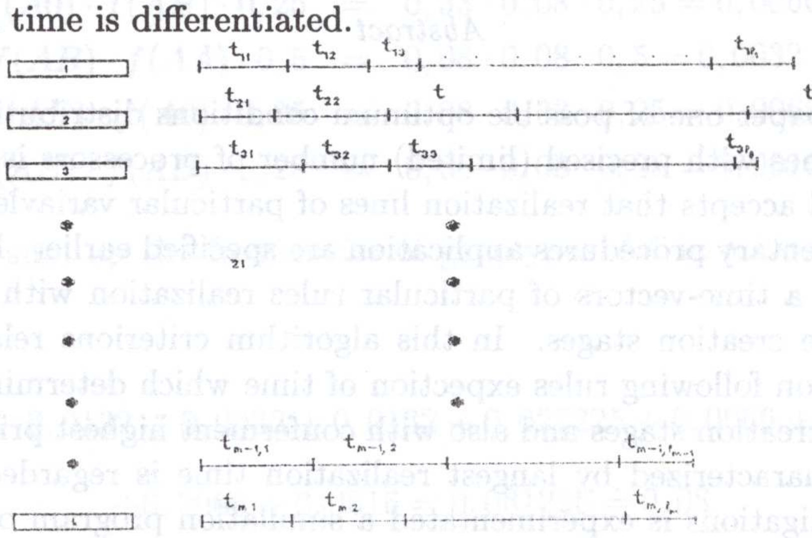


Fig. 1

Letter k is a service channel (k - number of processor - it's calculation unit). $\sum_{l=1}^{m_j} t_{jl}$ is the total realization time of variable-j l - is the number of component procedure at variable'sj creation line; n - number of variables designated in problem m_j - is the number of the last temporal component during variable - j oreation.

Example variable realization's procedure illustrates the following part of program.

$a := sqrt(x) + sqrt(y);$

$e := a - sin(p);$

$for\ i := 1\ to\ n\ do$

$for\ k := 1\ to\ mdo$

$d := d + e + v[i, j];$

Corfoming to program's sequence creation line of variable d can be represented like this:

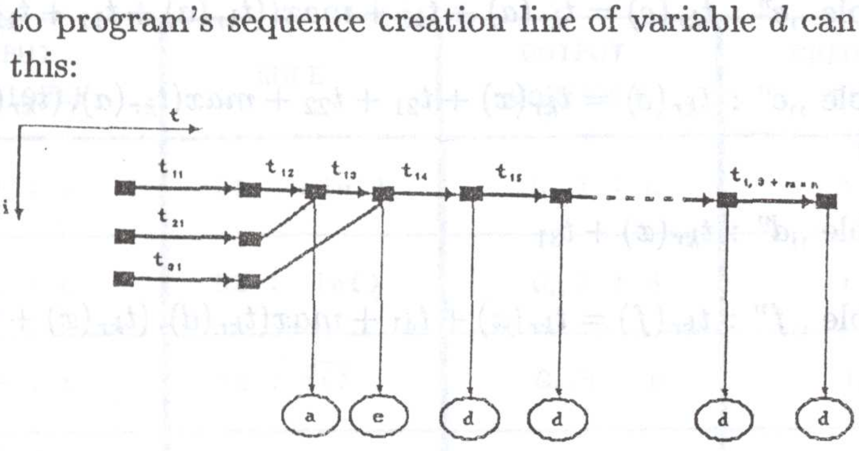


Fig. 2

During creation of variable d, variables' „a" and „e" values were calculated too. The following example contains description for situation in which one of variables is a basis to form some new variables (c, d, e, f; illustration 3).

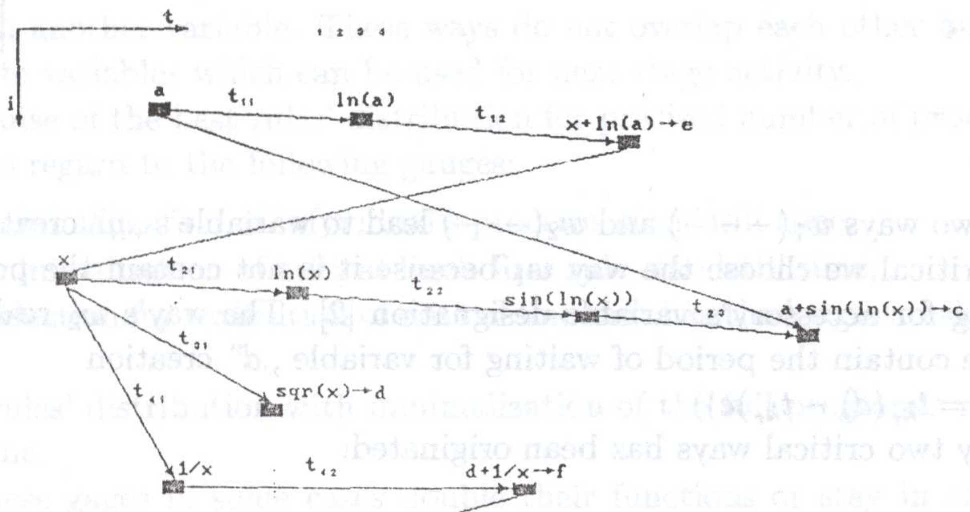


Fig. 3

t_{11} – finding the logarithm
t_{12} – summation
t_{21} – finding the logarithm
t_{22} – sine function
t_{20} – summation
t_{31} – involution
t_{41} – reciprocal
t_{42} – summation
Variables creation times are equal:

variable „e” : $t_{kr}(e) = t_{kr}(a) + t_{11} + max((t_{kr}(a) + t_{11} + t_{12}), t_{kr}(x))$

variable „c” : $t_{kr}(c) = t_{kr}(x) + t_{21} + t_{22} + max(t_{kr}(a), (t_{kr}(x) + t_{21} + t_{22} + t_{23}))$

variable „d” : $t_{kr}(x) + t_{31}$

variable „f” : $t_{kr}(f) = t_{kr}(x) + t_{41} + max(t_{kr}(d), (t_{kr}(x) + t_{41} + t_{42}))$

$t_{kr}(x)$ - is the moment in which variable „x” is generaled Variable creation procedure consist of operations' set forming so-called critical way that cannot contain periods of waiting for example

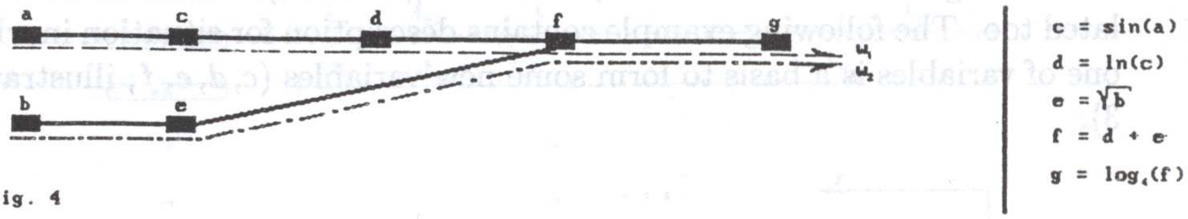


Fig. 4

Two ways $w_1(- - -)$ and $w_2(- \cdot -)$ lead to wariable's „g” creation but as a critical we choose the way w_1 because it is not contain the periad of waiting for accessory's variable designation [2]. The way's w_2 realization will be contain the period of waiting for variable „d” creation
$(\bullet)(t_o = t_{k_r}(d) - t_{k_r})e))$
Finally two critical ways has bean originated:

- w_1 (already described) and

- w_2 - critical way creation of variable e $(b - e)$.

Description procedure with object's $< G, D, A >$ assistance [1] where: G - is an input - autput value's set and parameters; D - is the rules' set and A -s rules' attributes (stages); reqiures farther specify considering conversion's necessity of turning variable's execution into variable's creation ar individual problem's realization time. In this case a nev ovject $< G, D, T >$ will be generated where T means the set of particulares' rules realizations times. This object can be described by means of table

Tabl. 1

INPUT VARIABLE	RULE	OUTPUT VARIABLE	CREATION TIME
$G_{IN}1$: a	D1 : sin()	$G_{OUT}1$: c	t_{11}
$G_{IN}2$: c	D2 : ln()	$G_{OUT}2$: d	t_{12}
$G_{IN}3$: b	D3 : $\sqrt{()}$	$G_{OUT}3$: e	t_{21}
$G_{IN}4$: e $G_{IN}5$: d	D4 : "+"	$G_{OUT}4$: f	t_{13}
$G_{IN}6$: f	D5 : log ()	$G_{OUT}5$: g	t_{14}

The oritical way of variable's creation cannot be a part of oritical way creation another variable. These ways do not overlap each other but only calculate variables which can be used for next stage activity.

The couise of the best rules' distribution for precised number of processors requires regard to the following gauges:

– minimalization of infyvidual's procesor standstill time;

– minimalization of critical line's first rule standstill time;

– selection for realization of critical line that has the longest realization time;

– rules' distribution with minimalization of the full program's realization time.

These gages in some cases double their functions or stay in clash so elaborated optimization method must be relevant to right selection of the [3].

From many possible problem's solutions of optimum distribution we choosed the optimization based on the following ganges:

– minimalization of standstill time of the critical line first rule to creation one of variables.

– a choice for critical line realization which has the longest realization time; (next)

– a critical line's choice that augurs minimalization of realization time for complete program.

The first algorithm of optimization with assumption that all solutions' realizations' time are equal will be contained succeding processors alleviations by succeeding rules with in succession placed oritical lines (it's in succession of decreasing whole critical line's realization time.

Exemplary separation of processes for k - processors (service channels), n - calculation's procedures (critical lines) we propose here.

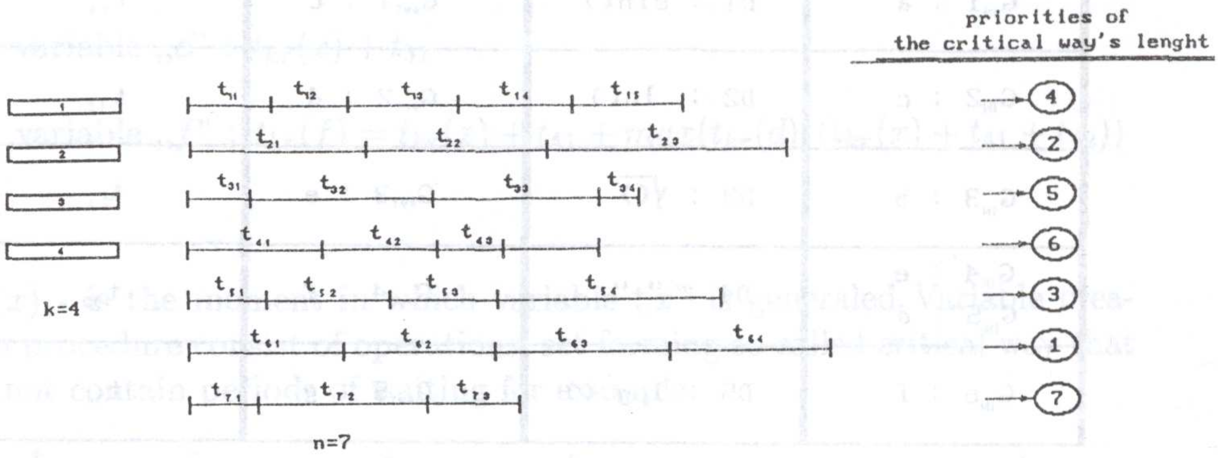


Fig. 5

Description of these and next stages can be partrayed by an algorithm:

1. Choice of maximal critical line (j) from unserviced.

2. Turning the following unrealized rule (i) to the first free processor (k - channel).

2.1. Designation of channel's and line's unlockation time.

$$R_k = R_k + t_{ji}; \quad S_j = R_k$$

R_k, S_j are the following moments' of channel - k and line - j unlockation.

2.2. Reducation of length the first segment of choosed line realization time.

$$DK_j = \sum_{i=1}^{m_j} t_{ji} - t_{j1} = \sum_{i=2}^{m_j} t_{ji}$$

where

DK_j – global realization time of line - j (it's the critical line) t_{ji} - temporal components.

3. Checking if channels ($Z_k = \{0,1\}$) are engaged. If they are free then go to top [1]

4. Checking if waitimg limes exist.

If they are found then go to [5] but if nat one needs to check when the longest of realization lines will be unlocked. $S0_j$ - standstill time of line - j

5. Estimating of all lines service pririty

$$P_j = DK_j + \alpha \cdot S0_j$$

where

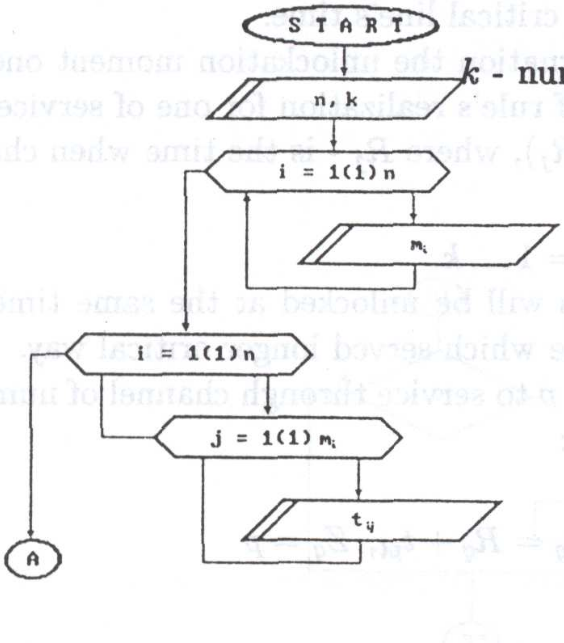
P_j - service priority of line - j

α - coefficient regarding on importance of standstill time with reference to on importance of particular rules.

6. Directing a line with maximal priority P_j to service if it is the standstill line for free channel. On the other hand if it is the engaged line one must find a channel which will be as the first unlocked $minR_k$ (pkt.1). Reading data in it means all critical lines relavant to volue designation of particular program's variables (t_{ij}) will be an introductiory activity.

($i = 1 \ldots n$, where n is the number of variables,

$j = 1 \ldots m_k$, where m_k is the number of variable's creation stages. (Figure 7.)



k - number of channel (processors)

Fig. 6

Before second stage one need to designate maximum realization time of crical way $\max\limits_{1 \le i \le n} \sum\limits_{j=1}^{m_i} t_{ij}$.

Maximal creation time of variable - i is also maximal priority for critical line - i.

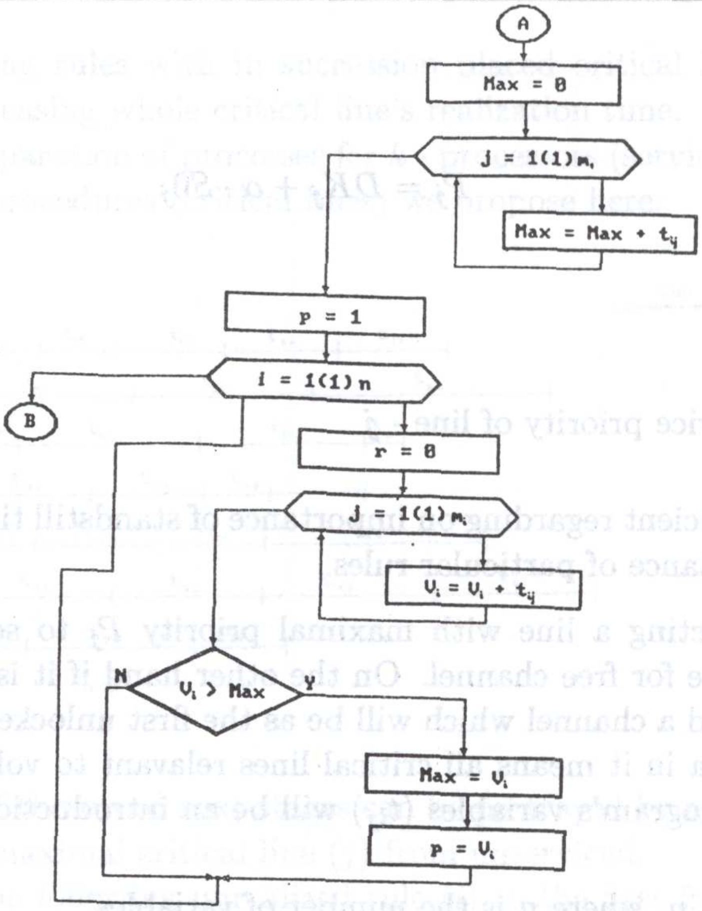


Fig. 7

$-V_i$ is realization's and critical line's time.

The next stage will be designation the unlockation moment one of service channels it means the end of rule's realization for one of serviced lines.

It will be, of course, $\min_{1 \le j \le k} (R_j)$, where R_j - is the time when channel - j is engaged.

In begining stage $R_j = 0, j = 1 \ldots k$.

In case while some channels will be unlocked at the same time next as a first will be choosed that one which served longer oritical way.

In case while line of number p to service throngh channel of number q then will be following corrections:

$$R_q = R_q + t_{pl}, \; Z_q = p$$

$$V_p = V_p - t_{p,l}$$

where l - number of rule (succeeding stage), resulting from procedures' sequence which generate variables with index p.

Z_q - variable showing which line is served by channel q.

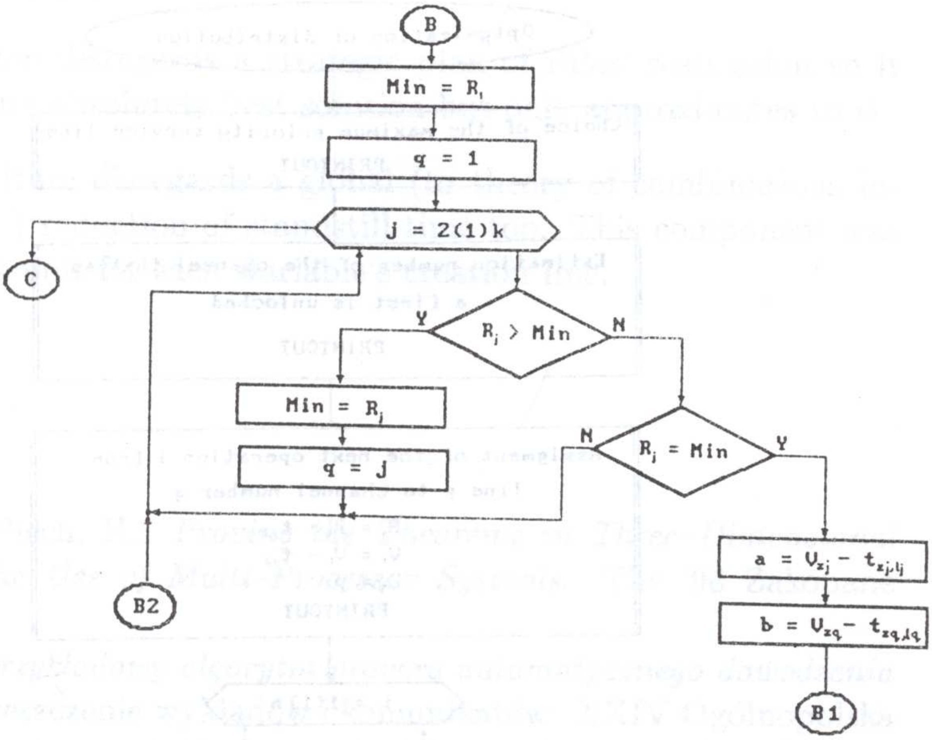

B
Min = R1
q = 1
j = 2(1)k
C
Y
Rj > Min
N
Min = Rj
N
Rj = Min
Y
q = j
a = Vzj − tzj,lj
B2
b = Vzq − tzq,lq
B1

Fig. 8

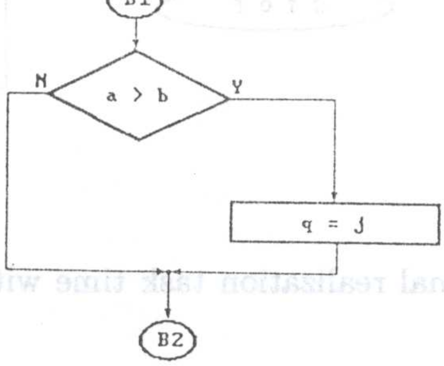

B1
N
a > b
Y
q = j
B2

Fig. 9

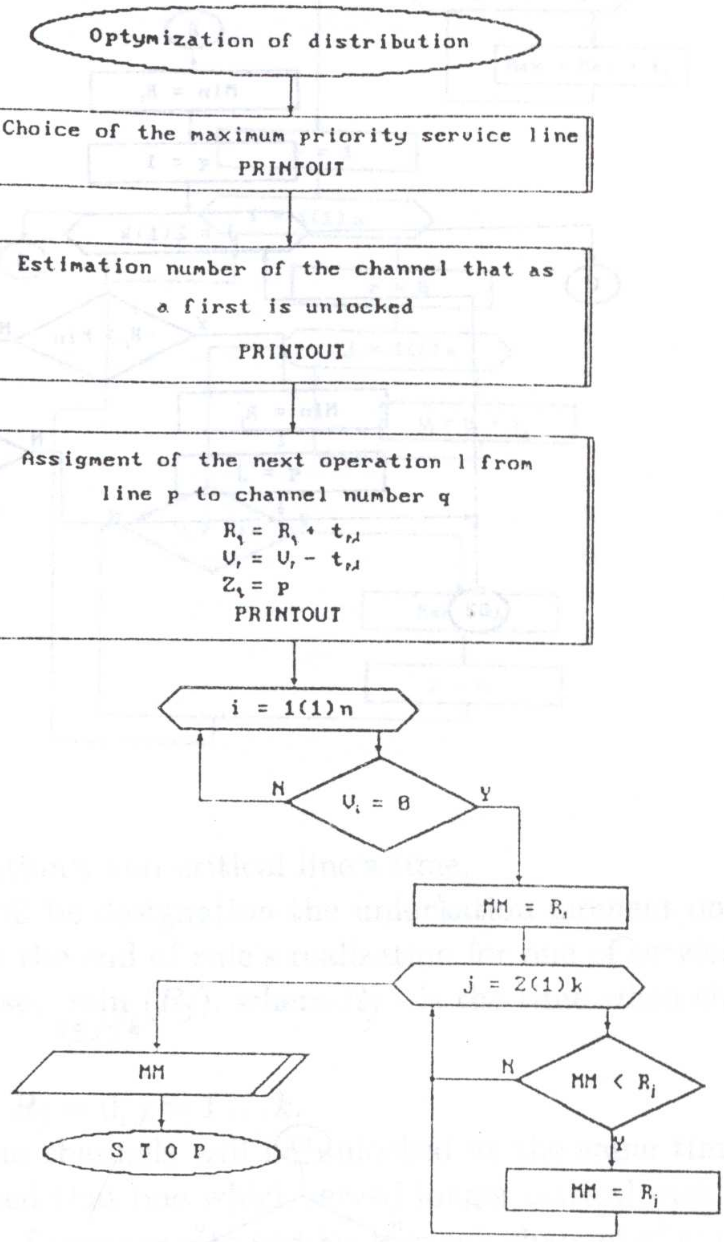


Fig. 18

MM - maximal realization task time with application of distribution algorithm.

Conclusions.

1. Presentated optimization of distribution algorithm is only one of many lines solution this problem.

2. In case line's realization which after current activity end is still the langest line one needn't to change the processor that executes next rule from the line.

3. The algorithm disregards a strategic plan of rules' realization so it doesn't ensure absolutely best solution but only approximates to it.

4. In the algorithm disregards a global (to theory of combinations interpretation) reduction of standstill time too. This component was handled as a unit for each wariable's creation line.

References

[1] Borowik, A., Piech, H.: *Proving the Theorems in Three–Dimensional Logics with the Use of Multi–Processor Systems.* The '96 Zakopane Conference.

[2] Borowik, P.: *Przykładowy algorytm provera automatycznego dowodzenia twierdzeń.* Streszczenie wykładów i komunikatów. XXIV Ogólnopolska Konferencja Zastosowań Matematyki, Zakopane'95, str. 13-14 (1995)

[3] Borowik, A., Piech, H.: *Procedures of the Assignment of Tasks to the Processors in Parallel Systems.* The'96 Zakopane Conference.

[4] Borowik, A., Chłopaś, T., Piech, H.: *The Use of Parallel Structures for Searching for the Extreme of a Function by the Fastest Drop Method.*

[5] Chłopaś, T., Piech, H., Kotkov, A., Pawlik, G.: *On the Problem of the Parallel Distribution of Processes.* Parallel Numerics '95, Sorrento, Italy.

[6] Rasiowa, H., Sikorski, R.: *The mathematics of metamathematics.* PWN, 1976.