УДК 004.42
DOI: 10.17277/vestnik.2016.03.pp.350-357

# PARALLEL MPI-IMPLEMENTATION OF THE BRANCH-AND-BOUND ALGORITHM FOR OPTIMAL SELECTION OF PRODUCTION EQUIPMENT

**A. B. Borisenko[1], S. Gorlatch[2]**

*Department "Computer-Integrated Systems in Mechanical Engineering",
TSTU, Tambov, Russia (1); borisenko@mail.gaps.tstu.ru;
Department "Mathematics and Computer Science",
Westfälische Wilhelms-Universität, Münster, Germany (2)*

**Keywords:** combinatorial optimization; master-worker paradigm; MPI; multiproduct batch plant; optimal equipment selection; parallel branch-and-bound.

**Abstract:** In this paper, we propose a parallel implementation of the branch-and-bound optimization technique on distributed-memory systems using Message Passing Interface (MPI). We employ parallel branch-and-bound to accelerate a real-world example application: optimal selection of production equipment for multi-product batch plants. We describe the master-worker organization of our parallel algorithm: a single master process dispatches a subset of computations to multiple worker processes and gathers computed results from them. For exchanging messages between master and worker we use MPI's point-to-point communication functions. We report experimental results about the speedup and efficiency of our parallel implementation. We observe that the master process may become a bottleneck for the overall application performance if it controls too many workers processes, because of the increasing communication overhead.

## Introduction

Selecting the production equipment of a Chemical-Engineering System (**CES**) is one of the main problems when designing chemical multi-product batch plants, e.g., for synthesizing chemical dyes and intermediate products, photographic materials, food, pharmaceuticals etc. Using multi-product batch plants offers the opportunity of quick response to market changes. Building these plants from standardized modules can additionally help to reduce time to market and costs.

The design problem consists in determining the number and capacity of the major processing equipment items, utilities, and storage tanks, such that the design and production objectives are met at the lowest possible capital and operating cost. The problem of optimal design of multi-product batch plants is typically formulated as a mixed integer nonlinear programming (**MINLP**) problem [1]. Many nonlinear models for batch design, which are based on the assumption of continuous sizes, can be reformulated as MILP problems when sizes are restricted to discrete values [2, 3]. Finding an optimal solution to this NP-hard problem can be a difficult task due to "combinatorial explosion" – the number of combinations to be examined grows exponentially, such that even the fastest computers will require an intolerable amount of time to analyze them.

The Branch-and-bound (**B&B**) approach is a popular technique for solving such NP-hard optimization problems. In the B&B method, the search space is represented as a tree whose root node is the original unsolved problem, the internal nodes are partially solved subproblems and the leaves are the potential solution(s). The Branch-and-bound proceeds in several iterations during which the best solution found so far (upper bound) is progressively improved. During the exploration, a bounding mechanism, based on a lower bound function, is used to eliminate all the subproblems (i.e., cut their corresponding sub-trees) that are not likely to lead to optimal solutions. This powerful mechanism reduces significantly the size of the explored search space and, thus, its exploration time cost. In this paper, the focus is on a particular practical application of the B&B algorithm – the optimal selection of chemical equipment for multi-product batch plants. ). For improving algorithm performance we investigate an implementation of the B&B method on a parallel distributed-memory system via Message Passing Interface (**MPI**) [4].

## Problem Formulation

A *Chemical-Engineering System* is a set of equipment (reactors, tanks, filters, dryers etc.) which implement the processing stages for manufacturing certain products. Assuming that the number of units at every stage of CES is fixed, the problem can be formulated as follows: CES consists of a sequence of $I$ processing stages. Each $i$-th processing stage of the system can be equipped with equipment units from a finite set $X_i$, with $J_i$ being the number of equipment units variants in $X_i$. All equipment unit variants of a CES are described as $X_i = \left\{x_{i,j}\right\}$, $i = \overline{1,I}$, $j = \overline{1,J_i}$, where $x_{i,j}$ is the main size $j$ (working volume, working surface) of the unit suitable for processing stage *i*.

A Chemical-Engineering System variant $\Omega_e$, $e = \overline{1,E}$ of a CES, where $E = \Pi_{i=1}^{I} J_i$ is the number of all possible system variants, is an ordered set of equipment unit variants, selected from the respective sets.

Each variant $\Omega_e$, of a system must be in operable condition (*compatibility constraint*) i.e., it must satisfy the conditions of a joint action for all its processing stages: $S(\Omega_e) = 0$ if compatibility constraint is satisfied. An operable variant of a CES must run at a given production rate in a given period of time (*processing time constraint*), such that it satisfies the restrictions for the duration of its operating period $T(\Omega_e) \le T_{\max}$, where $T_{\max}$ is a given maximum period of time.

Thus, designing an optimal CES can be formulated as the following optimization problem: to find a variant $\Omega_e^* \in \Omega_e$, $e = \overline{1,E}$ of a CES, where the optimality criterion – equipment costs $\mathrm{Cost}(\Omega_e^*)$ reaches a minimum and both compatibility constraint and processing time constraint are satisfied:

$$\Omega_e^* = \operatorname{argmin} \mathrm{Cost}(\Omega_e), \ e = \overline{1,E};$$

$$\Omega_e = \left\{x_{1,j_1}, x_{2,j_2}, \dots, x_{I,j_I} \mid j_i = \overline{1,J_i}, i = \overline{1,I}\right\}, \ e = \overline{1,E};$$

$$x_{i,j} \in X_i, \ i = \overline{1,I}, \ j = \overline{1,J_i};$$

$$S(\Omega_e) = 0, \ e = \overline{1,E};$$

$$T(\Omega_e) \le T_{\max}, \ e = \overline{1,E}.$$

We use the comprehensive mathematical model of CES operation, including expressions for checking constraints, calculating the optimization criterion, etc., which was initially presented in [5, 6].

## Shared-memory Approach for Algorithm Parallelization

All possible variants of a CES with I stages can be represented by a tree of height $I$ (see Figure 1). Each level of the tree corresponds to one processing stage of the CES. Each edge corresponds to a selected device variant taken from set $X_i$, where $X_i$ is the set of possible device variants at stage $i$ of the CES. Each node $n_{i,k}$ at the tree layer $N_i = \left\{n_{i,1}, n_{i,2}, \dots, n_{i,k}\right\}$, $i = \overline{1, I}$, $k = \overline{1, K_i}$, $K_i = \Pi_{l=1}^{i} J_l$ corresponds to a variant of a beginning part of the CES, composed of devices for stages $1$ to $i$ of the CES. Each path from the tree's root to one of its leaves thus represents a complete variant of the CES.

To enumerate all possible variants of a CES in the aforementioned tree, a depth-first traversal is performed: starting at level 0 of the tree, all device variants of the CES at a given level are enumerated and appended to the valid beginning parts of the CES. Valid beginning parts are obtained at previous levels, starting with an empty beginning part at level 0. This process continues recursively for all valid beginning parts that result from appending device variants of the current level to the valid beginning parts from previous levels. When a leaf node is reached, the recursive process stops and the current solution is compared to the current optimal solution, possibly replacing it.

In B&B, the search space is usually considered as a tree, which allows for a structured exploration of the search space. The tree structure provides a natural parallelism allowing concurrent evaluation of subproblems (calculations for the various branches can be carried out simultaneously) using parallel computing technology. In this paper, we use the *master-worker paradigm* [7, 8] for a distributed-memory approach to parallelization of B&B algorithm: single *master* process dispatches a subset of computations to multiple *worker* processes and gathers computed results from them. This approach is illustrated by Figure 2.

The master process performs a depth-first traversal of the tree using a recursive procedure to some level *G* (*granularity*), *1 ≤ G ≤ I*, where *I* is the tree levels number. Using this procedure, the master creates beginning parts of the CES. At the last level of recursion, the master waits for worker messages, which can be of two types: *solution* (SOLUTION) or *job request* (REQUEST_WORK). If the master receives a *solution* message, the costs of the received solution are compared to the costs of the current

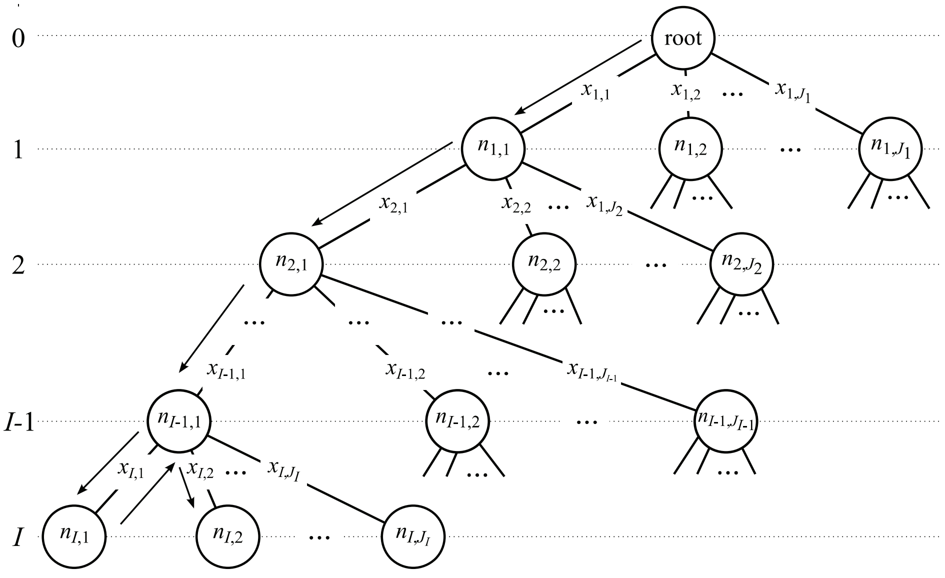

**Fig. 1. Tree traversal in depth-first search**

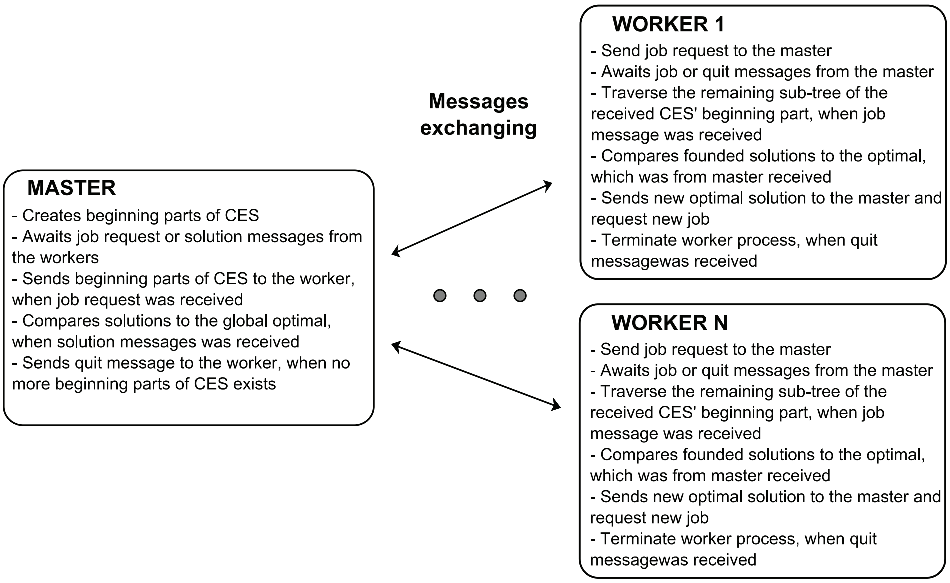
Messages exchanging
MASTER
- Creates beginning parts of CES
- Awaits job request or solution messages from the workers
- Sends beginning parts of CES to the worker, when job request was received
- Compares solutions to the global optimal, when solution messages was received
- Sends quit message to the worker, when no more beginning parts of CES exists
WORKER 1
- Send job request to the master
- Awaits job or quit messages from the master
- Traverse the remaining sub-tree of the received CES' beginning part, when job message was received
- Compares founded solutions to the optimal, which was from master received
- Sends new optimal solution to the master and request new job
- Terminate worker process, when quit messagewas received
WORKER N
- Send job request to the master
- Awaits job or quit messages from the master
- Traverse the remaining sub-tree of the received CES' beginning part, when job message was received
- Compares founded solutions to the optimal, which was from master received
- Sends new optimal solution to the master and request new job
- Terminate worker process, when quit messagewas received

**Fig. 2. Distributed-memory parallel branch-and-bound.**
**Master-worker paradigm**

optimal solution. If a better solution has been found by the worker, it is stored and replaces the current optimal solution. When a *job request* is received, the master responds by sending *job message* (DO_WORK) containing the current beginning part of the CES and the current optimal solution to the worker. Afterwards, a new beginning part of the CES is generated to be passed to a worker. If no new beginning part of the CES can be generated, the master returns from the recursive procedure. The master continues receiving solutions from workers and compares them to the optimal solution. However, if a worker sends a job request, the master sends a *quit message* (QUIT) to the worker, to terminate the worker process. After *quit messages* have been sent to all workers, the master process ends.

The worker process starts by sending a *job request* to the master and waits for the response. The response can be of one of two types: *job message* (DO_WORK) or *quit* (QUIT). If a *job message* comprising a beginning part of the CES and the current upper bound of the optimality criterion is received, the worker calls the recursive procedure. Within this procedure, the worker traverses the remaining sub-tree from levels $G$+1 to $I$, of the received CES' beginning part to find full solutions. If the worker finds a solution which costs do not exceed the upper bound of the optimality criterion, it makes this solution the new optimal solution. When the recursive procedure ends, the worker sends its new optimal solution, if any, to the master and requests a new job. If a *quit message* is received, the worker process terminates.

## Experimental Results

To study the speedup of our parallelization approach, we created C++ program implementations and conducted runtime experiments on a heterogeneous cluster consisting of:

– 36 nodes with 2 quad-core processors (Intel X5550 Nehalem, running at 2.6 GHz) with 3 GB RAM each;

– 198 nodes with 2 hexa-core processors (Intel Westmere X5650, running at 2.6 GHz) with 2 or 4 GB RAM each;

– 4 nodes with 4 eight-core processors (Intel Xeon E7550, running at 2 GHz) with 128 GB RAM each.

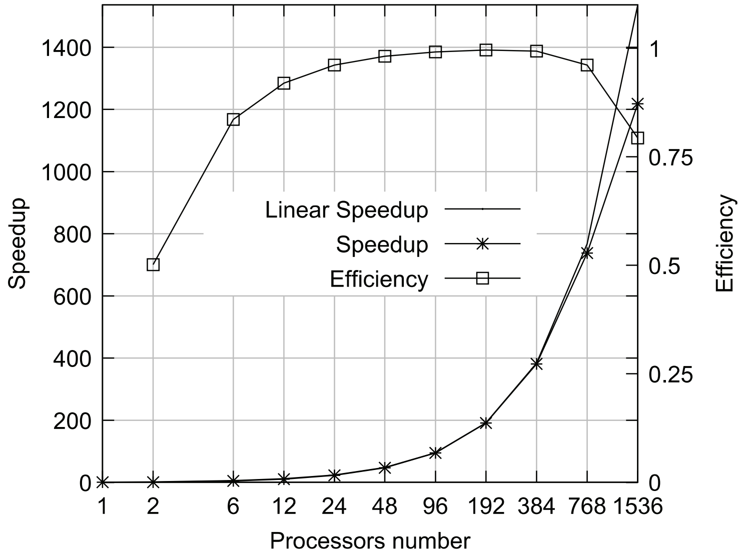

**Fig. 3. Measured speedup and efficiency of the MPI-based parallel program**

We studied the design of a CES consisting of 16 processing stages with 5 variants of devices at every stage as test case (total $5^{16} \approx 10^{11}$ CES variants). The implementations were written in C++ using MPI.

In our master-worker implementation, we used MPI's point-to-point communication functions *send* and *recv* for exchanging messages between master and worker.

Figure 3 shows the speedup and efficiency of our MPI-based implementation (the vertical axis has a logarithmic scale) using up to 64 Westmere nodes (up to 1536 processors).

The minimum number of processors for running the program is two (master and one worker). While there is no speedup when using 2 processors, it increases nearly linearly when using up to 768 processors. With greater numbers of processors, the growth of speedup slows down. The performance of the master process may become a bottleneck of application performance when it controls too many worker processes, because the master frequently communicates with all workers.

Runtime experiments for our implementations using a real-world example of a multi-product batch plant show that our solution provides considerable speedup. This is well correlated with experimental results obtained in [9], where near-linear speedups were also observed.

In future work we will investigate the use of a hierarchical master-worker implementation, in order to reduce the communication bottleneck which we observed in our current implementation. This paper presents a parallel version only of the branch-and-bound algorithm. In addition, quite interesting would be the parallelization of comprehensive mathematical model of CES operation. This problem requires also deeper and more detailed research in further works.

## Acknowledgement

*This work was generously supported by the DAAD (German Academic Exchange Service) and by the Ministry of Education and Science of the Russian Federation under "Mikhail Lomonosov II"-Programme.*

*References*

1. Ponsich A., Azzaro-Pantel, C., Domenech, S., Pibouleau, L. Mixed-integer nonlinear programming optimization strategies for batch plant design problems, *Ind. Eng. Chem. Res.*, 2007, vol. 46, no. 3, pp. 854-863.
2. Fumero Y., Corsano G., Montagna J.M. A Mixed Integer Linear Programming model for simultaneous design and scheduling of flowshop plants, *Appl. Math. Model*, 2013, vol. 37, no. 4, pp. 1652-1664.
3. Borisenko A.B., Antonenko A.V., Osovsky A.V., Filimonova O.A. [The System of Automated Selection of Auxiliary Equipment for Multi-Assortment Chemical Plants], *Transactions of the Tambov State Technical University*, 2012, vol. 18, no. 3, pp. 569-572. (In Russ., Abstract in Eng.)
4. Official MPI (Message Passing Interface) standards documents, errata, aviaeleble at: http://www.mpi-forum.org (accessed 16 January 2016).
5. Borisenko A.B., Karpushkin S.V. Hierarchy of processing equipment configuration design problems for multiproduct chemical plants, *J. Comput. Syst. Sci. Int.*, 2014, vol. 53, no. 3, pp. 410-419.
6. Karpushkin S.V., Zatsepina V.I., Zatsepin E.P. [Selecting Main Instruments to Enhance Process Systems of Multi-Assortment Chemical Plants], *Transactions of the Tambov State Technical University*, 2012, vol. 18, no. 3, pp. 552-557. (In Russ., Abstract in Eng.)
7. Ismail M.M., El-raoof O. abd, Abd EL-Wahed W.F. A Parallel Branch and Bound Algorithm for Solving Large Scale Integer Programming Problems, *Appl. Math. Inf. Sci.*, 2014, vol. 8, no. 4, pp. 1691-1698.
8. Borisenko A., Kegel P., Gorlatch S. Optimal Design of Multi-product Batch Plants Using a Parallel Branch-and-Bound Method, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6873 LNCS, pp. 417-430.
9. Gendron B., Crainic T.G. Parallel Branch-And-Bound Algorithms: Survey and Synthesis, *Oper. Res.*, 1994, vol. 42, no. 6, pp. 1042-1066.

# Применение параллельной реализации алгоритма ветвей и границ с использованием MPI для оптимального выбора технологического оборудования

**А. Б. Борисенко[1], С. Горлач[2]**

*Кафедра «Компьютерно-интегрированные системы в машиностроении», ТГТУ, Тамбов, Россия (1); borisenko@mail.gaps.tstu.ru;*
*кафедра «Математика и информатика», Вестфальский университет имени Вильгельма, Мюнстер (Германия) (2)*

**Ключевые слова:** комбинаторная оптимизация; многоассортиментные производства; оптимальный выбор технологического оборудования; парадигма «мастер–работник»; параллельный метод ветвей и границ.

**Аннотация:** Представлена параллельная реализация метода ветвей и границ для вычислительных систем с распределенной памятью с использованием интерфейса передачи сообщений – MPI (Message Passing Interface). Параллельный алгоритм используется для оптимального выбора технологического оборудования многоассортиментных химических производств. Параллельная программа разра-

ботана с использованием парадигмы «мастер–работник»: один управляющий мастер-процесс распределяет вычислительные подзадачи между множеством рабочих процессов, а затем собирает результаты их вычислений. Для обмена сообщениями между управляющим и рабочими процессами используются двухточечные (point-to-point communications) функции MPI. Представлены анализ ускорения работы программы и эффективности использования процессоров. Отметим, что эффективность использования процессоров и ускорение снижается при большом числе используемых процессоров. Это происходит вследствие того, что мастер-процесс становится узким местом программы из-за большого числа обслуживаемых им рабочих процессов.

*Список литературы*

1. Mixed-integer Nonlinear Programming Optimization Strategies for Batch Plant Design Problems / A. Ponsich [et al.] // Ind. Eng. Chem. Res. – 2007. – Vol. 46, No. 3. – P. 854 – 863.

2. Fumero, Y. A Mixed Integer Linear Programming Model for Simultaneous Design and Scheduling of Flowshop Plants / Y. Fumero, G. Corsano, J. M. Montagna // Appl. Math. Model. – 2013. – Vol. 37, № 4. – P. 1652 – 1664.

3. Система автоматизированного выбора вспомогательного оборудования многоассортиментных химических производств / А. Б. Борисенко [и др.] // Вестн. Тамб. гос. техн. ун-та. – 2012. – Т. 18, № 3. – С. 569 – 572.

4. Official MPI (Message Passing Interface) Standards Documents, Errata [Электронный ресурс]. – Режим доступа : http://www.mpi-forum.org (дата обращения: 16.01.2016)

5. Borisenko, A. B. Hierarchy of Processing Equipment Configuration Design Problems for Multiproduct Chemical Plants / A. B. Borisenko, S. V. Karpushkin // J. Comput. Syst. Sci. Int. – 2014. – Vol. 53, No. 3. – P. 410 – 419.

6. Карпушкин, С. В. Выбор основной аппаратуры для оснащения технологических систем многоассортиментных химических производств / С. В. Карпушкин, В. И. Зацепина, Е. П. Зацепин // Вестн. Тамб. гос. техн. ун-та. – 2012. – Т. 18, № 3. – С. 552 – 557.

7. Ismail, M. M. A Parallel Branch and Bound Algorithm for Solving Large Scale Integer Programming Problems / M. M. Ismail, O. Abd El-Raoof, W. F. Abd EL-Wahed // Appl. Math. Inf. Sci. – 2014. – Vol. 8, № 4. – P. 1691 – 1698.

8. Borisenko, A. Optimal Design of Multi-product Batch Plants Using a Parallel Branch-and-Bound Method / A. Borisenko, P. Kegel, S. Gorlatch // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). – 2011. – Vol. 6873 LNCS. – P. 417 – 430.

9. Gendron, B. Parallel Branch-And-Bound Algorithms: Survey and Synthesis / B. Gendron, T.G. Crainic // Oper. Res. – 1994. – Vol. 42, No. 6. – P. 1042 – 1066.

## Anwendung der parallelen Realisierung des Algorithmus der Zweige und der Grenzen unter Ausnutzung von MPI für die optimalen Auswahl der technologischen Ausrüstun

**Zusammenfassung:** Es ist die parallele Realisierung der Methode der Zweige und der Grenzen für die Computersysteme mit dem verteilten Gedächtnis unter Ausnutzung des Interfaces der Sendung der Mitteilungen – Message Passing Interface (MPI) angeführt. Der parallele Algorithmus wird für die optimalen Auswahl der technologischen Ausrüstung der vielsortimenten chemischen Produktionen verwendet.

Das parallele Programm ist unter Ausnutzung des Paradigmas der Meister - der Arbeiter ausgearbeitet: ein steuernden Meisterprozess verteilt die Rechenteilprobleme zwischen der Menge der Arbeitsprozesse und dann sammelt die Ergebnisse ihrer Berechnungen. Für den Austausch von den Mitteilungen zwischen dem Steuer- und den Arbeitsprozessen werden die Zweipunktfunktionen (point-to-point communications) MPI verwendet. Es sind die Analyse der Beschleunigung der Arbeit des Programms und der Effektivität der Nutzung der Prozessoren dargelegt. Es ist nötig zu bemerken, dass die Effektivität der Nutzung der Prozessoren und die Beschleunigung bei der großen Zahl der verwendeten Prozessoren sinkt. Es geschieht infolge dessen, dass der Meisterprozess ein Engplatz des Programms wegen großer Zahl der von ihm bedienenden Arbeitsprozesse wird.

## Application de la mise en œuvre parallèle de l'algorithme des branches et des frontières avec l'emploi de MPI pour le choix optimal de l'équipement technologique

**Résumé:** Est présentée la mise en œuvre parallèle de la méthode des branches et des frontières pour les systèmes informatiques avec une mémoire distribuée avec l'interface de transfert de message – Message Passing Interface (MPI). L'algorithme parallèle est utilisé pour le choix optimal de l'équipements technologique des productions chimiques à multi-produits. Le programme parallèle est élaboré avec l'emploi du paradigme maître - travailleur: un master processus qui dirige distribue les sous-tâches de calcul entre la multitude des processus de travail et ensuite collecte les résultats de ces calculs. Pour l'échange des messages entre les gestionnaires et les processus de travail sont utilisés les fonctions MPI à deux points (point-to-point communications). Est présentée l'analyse de l'accélération du fonctionnement du programme et de l'efficacité de l'utilisation des processeurs. L'efficacité de l'utilisation des processeurs et l'accélération diminuent avec un grand nombre des processeurs utilisés.

**Авторы:** ***Борисенко Андрей Борисович*** – кандидат технических наук, доцент кафедры «Компьютерно-интегрированные системы в машиностроении», ФГБОУ ВО «ТГТУ», г. Тамбов, Россия; ***Горлач Сергей*** – PhD, профессор кафедры «Математика и информатика», Вестфальский университет имени Вильгельма, г. Мюнстер, Германия.

**Рецензент:** ***Гатапова Наталья Цибиковна*** – доктор технических наук, профессор, заведующая кафедрой «Технологические процессы, аппараты и техносферная безопасность», ФГБОУ ВО «ТГТУ», г. Тамбов, Россия.