Automatic control of the distribution of tasks under conditions of uncertainty with the use of adaptive approach

E Zhigalov and M I Ozerova
Vladimir State University named after Alexander Grigorievich and Nikolai Grigorievich Stoletov, Vladimir, Russia

E-mail: ozerova.marina@gmail.com

Abstract. The subject of this article is the development of an adaptive approach for the optimal distribution of problems among solvers in conditions of uncertainty. Despite the large amount of research related to the construction of solutions for automatic control of task distribution, this issue remains relevant. As an alternative approach, a multi-level adaptive algorithm is proposed, which at each level filters incoming tasks according to solution methods, thereby significantly reducing the computational load. A distinctive feature of this algorithm is taking into account the time of task preprocessing, in particular, related to the current load of solvers and the distribution of tasks by solvers, in accordance with the maximum load.

1. Introduction

The problem of building automatic task distribution systems remains a hot issue despite a large number of research conducted in the area. There are several reasons for that:

- steady database size growth, that in turn leads to the problem of effective big data processing;
- automatic handling of task distribution at all stages of continuous improvement, in order to provide reduction of processing time and structuring the input data;
- the possibility of making in-place corrections based on the domain-specific input data analysis;
- reducing the user work by tuning and optimizing models and input data quality cost function;
- using models and methods of data mining, as well as machine learning, thus allowing to adapt the process of task distribution in case of uncertainty [1].

The task flow in question has several hundred thousands of elements, thus making it a problem in a large search space. It's worth pointing out, that such task can be solved by several methods, and therefore one has to choose a method. It's not enough just to analyze input data, but also to make a decision. It is a very common problem for enterprise-scale databases that have to deal with retail, banking, and web.

The huge flow of input data calls for various algorithms and approaches to data analysis, depending on specific requirements and technical possibilities. Therefore, creation of new algorithms that would allow to reduce drastically the time for task distribution, as well as amount of computation, taking in consideration all factors, is a relevant task. Data exchange is shown at figure 1.
2. State of art

Most applied tasks in economics, and also logic tasks including routing can be solved by various methods of linear programming (LP) [2-4]. Many approaches exist in that area [5]. But the problem can become really hard due to it's high dimensions. Optimization methods for high-dimension tasks are described for LP tasks, and finding global optima in [6, 7]. A math model for cutting sheet materials problem is described via transformation into linear integer programming with implicit information placed inside columns of cut matrix. A genetic algorithm is proposed as a solution.

An application of well-known algorithm “Classification and Regression Tree” [8] for credit scores is described in [5]. The algorithm in question solved the problem of classifying the clients using several features predefined by the bank, thus proposing the final decision about the credit, taking in account the risks. The client features serve as rules in this case.

The data analysis goes like this: a dependent variable is introduced, as well as the class for that variable. Independent variables are assigned and tree building criteria are used. Model type is defined, and finally a model is built.

Binary decision tree algorithm is widely used, although not always recommended instead of classic statistical methods. This is due to the fact that decision tree algorithms often produce complex trees overfilled with data, as well as over fitting, with too much nodes and branches.

Such branching trees can be hard for explanation and understanding [9]. For example, if sampling distribution has some specific properties and meets the preconditions of classic methods, it's more reasonable to use those methods instead of trees.

The Base Group Labs developer portal [10] offers Deductor Credit Pipeline for automation of decision making at all stages of credit score pipeline. This solution is implemented through a group of web services with SOAP interaction.

During the credit score process Deductor Credit Pipeline serves as decision agent, providing integration with various bank systems, e.g. CRM and office solutions, as well as ABC and databases. Combined models of big data analysis are also possible [11].

Each request handling takes about 1-3 minutes including interaction with external services, business rule checks and etc. Up to 1000 requests per minute can be handled in cluster with hot replace.

Paper [12] describes an algorithm of knowledge retrieval and structuring. This knowledge reflects the credit behavior and it's goal is the automation of borrower personal profile construction. The algorithm is based on graph models and production rule database.
Many solutions use data mining for the analysis. Despite the multiple algorithms provided by data mining, it's not always possible to find a good enough solution, especially in systems that cannot be describe appropriately in formal math.

It is also hard to adapt to the fact, that data changes with time, and the current model may not be as good as it's used to be. For example [13], a predictive model is proposed for system using the equivalent of Shroedinger equation with generic parameters.

Data mining is usually applied to big volumes of data, as well as small sets of non-formal data. Model combining uses a simple idea of separating data into groups that are easy to handle with simple algorithms, and groups that are not that easy to handle.

A part of the data is analyzed and excluded from further analyze, thus reducing the total time required for processing, allowing the economy of computational resources [11].

Paper [14] suggests an adaptive algorithm that allows to choose the most effective model, method and algorithm based on the problem domain. Adaptation criteria complement each other, thus forming a hierarchy, such that after processing all criteria is done.

The adaptation problem is solved from top to bottom:

- A general approach for modelling is chosen.
- A group of algorithms is chosen.
- A single algorithm is chosen for the solution.

Various numerical methods are used [15].

Data mining for classification and clustering is described in [16] for telecommunication corporations. Data retrieval is described in [17, 18].

3. Adaptive algorithm

A multilevel adaptive algorithm is suggested as an alternative for the data analysis in order to reduce amount and structuring the data. At each level the incoming tasks are filtered by solution methods, thus reducing computation cost drastically.

At each level tasks are filtered according to preset coefficients that correspond to the output requirements. Each level uses its own methods and algorithms matching the problem domain. The scheme of the algorithm is shown at figure 2.

![Figure 2. Combining algorithms, filtering tasks.](image-url)
3.1. Problem definition

The problem can be defined as building the optimal task distribution amongst solvers. There are \( n \) distinct tasks. \( N_i \) \((i=1..n)\) is the number of times each task is present in one pool, and each tasks is equal to \( a_i \) \((i=1..n)\) units of work. There are also \( M_j \) \((j=1..m)\) of solvers, each solver having \( M_j \) instances and requires \( b_j \) units of work. \( c_{ij} \) is the priority of \( i\)-th task chosen by \( j\)-th solver. \( x_{ij} \) is the number of instances of \( i\)-th task sent to the \( j\)-th solver. The problem is to assign all the tasks to the solvers so that the total time would be minimal.

\[
F = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij}x_{ij} \rightarrow \min \tag{1}
\]

All tasks that belong to one pool are completely distributed amongst solvers. An equality can be shown:

\[
\sum_{i=1}^{n} a_i = \sum_{j=1}^{m} b_j \tag{2}
\]

Special conditions.

Suppose that there are \( N \) tasks currently, and \( N_p \) are sent to solver. It is necessary to minimize the number of tasks not sent to the solver.

\[
\Delta N_{\min} = N - N_p
\]

\[
N_p = \sum N_i = \sum_{i=1}^{n} a_i
\]

\[
M_p = \sum M_j = \sum_{j=1}^{m} b_j \tag{3}
\]

\( t_{ij} \) — distribution time of \( i\)-th task for \( j\)-th solver. Therefore, the set can be rewritten like this:

\[
x_{ij} = t_i \sum_{i=1}^{n} a_i + t_j \sum_{j=1}^{m} b_j \tag{4}
\]

Problem constraints:

\[
\sum_{i=1}^{n} x_{ij} = a, i = 1..n
\]

\[
\sum_{j=1}^{m} x_{ij} = b, j = m
\]

\( x_{ij} \geq 0, i = 1..n, j = 1..m \) \tag{5}

There can be several distributions corresponding to the constraints from (5). Therefore we need to find optimal distribution, that has the minimal total priority.

1) Tasks must be distributed completely.

The amount of tasks exceeds the possibility of solvers, i.e. the following inequality is true:

\[
x_{ij} \geq 0, i = 1..n, j = 1..m \tag{6}
\]

In that case an additional fake solver \((b_n+1)\) is introduced with the priority:

\[
c_{n+1,j} = 0, j = 1..m
\]
where:
\[ b_{m+1} = \sum_{i=1}^{n} a_i - \sum_{j=1}^{m} b_j \]

After introduction of fake solver we get equality (2).

2) Solvers capacity exceeds the amount of one pool, i.e. the following inequality is true:
\[ t_{ij} \sum_{i=1}^{n} a_{i} t_{ij} \sum_{j=1}^{m} b_{j} \]  
(7)

In that case we introduce a fake task \((a_{n+1})\) with the priority defined as:
\[ c_{m=1,i} = 0, i = 1, n \]

where:
\[ a_{n+1} = \sum_{j=1}^{m} b_{j} - \sum_{i=1}^{n} a_{i} \]  
(8)

And that fake task transforms the problem to (2)

After that the problem can be solved using the potential method. All tasks must be distributed amongst solvers, therefore aggregation coefficient should be \(K_i\).
\[ \sum_{i=1}^{n} c_{ij} = K_i = 1 \]

The following inequality must hold
\[ c_{ij} \min < \beta \]

where \(Z_j\) is a relative value.
\[ (c_{ij} - Z_j) > \beta \]

where \(Z_j\) is the relative quantity, the number of tasks in the \(j\)-th solver \((S_j)\) divided by the total number of tasks \((S)\).
\[ Z_j = \frac{S_j}{S} \]  
(9)

Ideally, \(Z_{bj}\) is equal to \(c_{ij}\).

The priority \(c_{ij}\) depends on efficiency of solving \(i\)-th task \((a_i)\), that is calculated using expert knowledge and the importance of \(j\)-th solver \((\mu_j)\) that can be calculated like this:
\[ c_{ij} = \frac{a_i + \mu_j}{100} \]  
(10)

\(\mu_j\) - is the importance value of solver:
\[ \mu_j = 1 - \frac{R_j}{Z_j} \]  
(11)

\(R_j\) - maximal number of tasks that can be solved using \(j\)-th solver.

The order of the task analysis is shown at figure 3.
3.2. Task distribution algorithm

Suppose that task $N_i$ is defined by variable $X_i$.

Step 1. Input of task $X_i$

Step 2. Task is handled by the algorithms of each solver.

Step 3. A list of solution methods is acquired. A check is performed for each task in order to calculate the possible execution time by the corresponding method.

Step 4. A list of solution methods is acquired.

Possible situations

4.1. List includes some solution methods for all $i=1...n$, i.e. $D_B[i]=true$ if $i$-th methods is present in list, and false — otherwise. In that case go to 5.

4.2. List includes all methods $(m)$, i.e. all $D_B[i]=true$. In that case go to 5.

4.3. List includes no possible solutions, i.e. all $D_B[i]=false$. In that case go to 9.

Step 5. Generate random variable $r$ with even distribution.

Step 6. cycle $i = [1;n]$ step = 1.

Step 7. if ($r <= K AND D_B[i] = true$) $OR$ ($D_B[i] = true AND (K_i - Z_i) > \beta$), where $Z_i$ – relative value for $i$ – th method divided by total amount of tasks. Ideally this value is equal to $K_i$. In that case go to 9. Else go to 8.

Step 8. $r = r - K_i; i = i + 1$. if $i <=n$, go to 6, else go to 8.

Step 9. End of distribution.

Figure 3. The order of the task analysis.
Step 10. If the tasks have not been distributed, go to 4 and afterwards star the loop from 5. Otherwise end of work.

The algorithm structure is shown at figure 4.

![Distribution Algorithm](image)

**Figure 4.** Distribution Algorithm.

4. **Conclusion**
As a result of the conducted research, the problem of a large stream of tasks distribution under conditions of uncertainty was analyzed. A mathematical model and an algorithm for calculation of task flow distribution that were applied to the problem of distribution of applications for bank guarantees in a multi-banking environment are considered taking into account the specifics of the bank indicators influencing the positive solution. The algorithm is tested on 2500 requests, it adapts well to the assigned task.

As of July 2017 system prototype was implemented, which includes collection and transformation of data related to bank guarantees provision. Currently libraries for determination of weighting coefficients methods are developed, which will allow to fully automate the process of decision making for bank guarantees issuance.
References

[1] Shcherbakov M V 2014 Intellectual support in making managerial decisions in the cycle of constant improvement (Volgograd)

[2] Maranas S D and Floudas S A 1992 A global adjustment approach for Lennard-Jones microclusters J. Chem. Phys. 97 7667-78

[3] Eremin I I 1998 Theory of Linear Optimization (Ekaterinburg: UrB RAS)

[4] Vasiliev F P and Ivanitsky A Yu 2003 Linear Programming (Moscow: Factorial)

[5] Tsurkov V I 1981 Decomposition in Problems of Large Dimension (Moscow: Nauka)

[6] Nguyen Minh Hang et al 2006 A model combining genetic algorithm and simplex method for solving a production expense minimizing problem Journal of Computer Science and Cybernetics 22(4) 319-24

[7] Ngoc Thang Mai, Wang Muon Ha, Kamaev V A, Shcherbakov M V and Ku-ang Vinh Thai Modeling and optimization of the management of an intelligent hybrid power system with renewable energy sources Management of Large Systems: electron, coll. sci. tr. (Institute of management problems named after V A Trapeznikov)

[8] Breiman L, Friedman J H, Olshen R A and Stone C T 1984 Classification and Regression Trees (Belmont, California: Wadsworth)

[9] Lebedev B K, Lebedev O B and Lebedeva E M 2016 Ant algorithm for constructing a binary decision tree Izvestiya SFU. Technical science 7(180) 74-88

[10] Deductor Credit Pipeline "Automating Decision Making in Mass Crediting" (circulation date 21 October 17) Retrieved from: https://basegroup.ru/solutions/ready-solutions/credit-pipeline

[11] Arustamov Alexey Portal of developers of BaseGroup Labs (circulation date 31 October 17) Retrieved from: https://basegroup.ru/community/articles/very-large-data

[12] Andieva E Yu and Semenova I I 2008 The way to build a psychological profile of the borrower to assess the risks in consumer finance Risk Management 1 56-62

[13] Britkov V B and Bulychev A V 2010 Methods for analyzing large volumes of weakly structured information Information Technologies and Computing Systems Federal Research Center "Informatics and Control" of the Russian Academy of Sciences (Moscow) 1 36-44

[14] Zhigalov I Y 1997 Models of functional blocks for automated hybrid design of nonlinear devices Radio engineering 7 34-9

[15] Kalitkin N N 1978 Numerical Methods (Moscow: Nauka)

[16] Samarkin M E and Tarasov V N 2016 Classification of large data of a telecommunications company using DATA MINING technology Infocommunication technologies 14(3) 258-63

[17] Tyurin A G and Zuev I O 2014 Cluster analysis Methods and algorithms of clustering Russian Technological Journal 2(3) 86-97

[18] Jagadish H V 1990 Linear Clustering of Objects with Multiple Attributes (ACM)

[19] Gruzdev A V 2012 Application of the CART algorithm for banking scoring problems Financial Management Publishing House "Grebennikov" 3 174-89