# The algorithm for labor resource allocation in IT project implementation based on stochastic gradient descent

Oleksandr Kuchanskyi[1,2,†], Myroslava Gladka[1,3,*,†], Yaroslav Hladkyi[3,†], Volodymir Druzhynin[3,†] and Hanna Tereshchuk[3,†]

[1] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Prosp. Peremohy, 37, 01601 Kyiv, Ukraine

[2] Astana IT University, Mangilik El avenue, Business center EXPO 55/11, block C1, 010000 Astana, Kazakhstan

[3] Taras Shevchenko National University of Kyiv, Volodymyrska Street 60, 01601 Kyiv, Ukraine

**Abstract**

This article delves into a fresh, innovative strategy for allocating human resources during IT project execution. We introduce a groundbreaking algorithm leveraging Stochastic Gradient Descent (SGD), which allows for the effective and continuous optimization of task assignment across development teams. We place particular focus on the distinct challenges of the IT ecosystem: the wide array of technologies and skill sets that demand precise matching, the rapid and ever-changing nature of project requirements, and the need for adaptive team leadership that carefully balances the specific qualifications and current workload of every specialist (be they a developer, tester, or designer). Our proposed algorithm is designed to aggressively minimize overall project costs and completion time. It achieves this by factoring in more than just individual output; it also considers the synergistic benefits between team members and the opportunity for skill enhancement. In stark contrast to older, often inflexible, static methodologies, this new approach is iterative and highly scalable, making it an ideal fit for the fast-paced environment of Agile projects. We thoroughly evaluated the algorithm's performance by conducting a detailed comparative analysis against established heuristic methods using simulated data sets. The findings conclusively show a substantial uplift in critical performance metrics, solidifying the case for using SGD as a powerful tool for labor resource management in IT. This capability not only ensures peak operational effectiveness for the team but also actively cultivates motivation and professional advancement among its members.

**Keywords**

IT-system, algorithm, resource allocation, IT-project, stochastic gradient descent (SGD), optimization, project management, machine learning, Agile, labor resources, cost and time optimization.

## 1. Introduction

The rapid advancement of information technologies and the continuous increase in the IT-projects complexity demand from managers not only technical expertise but also effective tools for resource management [1]. One of the most critical and, at the same time, the most challenging tasks is the optimal allocation of labor resources [2, 3]. The success and the economic viability of a project largely depend on how efficiently tasks are distributed among team members. Improper allocation may lead to a range of adverse consequences, including missed deadlines, budget overruns, reduced quality of products [4], and no less importantly, emotional burnout and team demotivation.

Traditional management methods, such as the Gantt chart and the Critical Path Method (CPM), have proven effective for projects with fixed requirements and predictable processes [6, 7]. However, within the context of contemporary agile methodologies and the high dynamism of the IT environment – where project requirements may change during implementation – these methods often lack sufficient flexibility [8, 9]. They fail to account for key factors such as the unique skill of

*AIT&AIS'2025: International Scientific Workshop on Applied Information Technologies and Artificial Intelligence Systems, December 18–19 2025, Chernivtsi, Ukraine

1,* Corresponding author.

† These authors contributed equally.

kuchansky.a@gmail.com (O. Kuchanskyi); miragladka@gmail.com (M. Gladka); hladkky@gmail.com (Y. Hladkyi); volodymir.druzhynin@knu.ua (V. Druzhynin); ganna.tereschuk@knu.ua (H. Tereshchuk)

0000-0003-1277-8031 (O. Kuchanskyi); 0000-0001-5233-2021 (M. Gladka); 0000-0003-2914-8685 (Y. Hladkyi)); 0009-0009-5049-0099 (V. Druzhynin); 0000-0001-7573-9748 (H. Tereshchuk)

each developer, tester, or designer; the synergy within team; the potential for learning and professional development; and the necessity of maintaining balanced workload to prevent project "bottlenecks" [10, 11].

The answer to this tricky problem needs new ideas that can change with the conditions.

This piece of writing suggests a fresh method called Stochastic Gradient Descent (SGD) – a way often used in learning machines – which helps find the best match by making a tough loss, time spent, and team tasks more even [12].

The suggested method is easy to grow, it makes it good for small-startups or big company plans. The goal of this study is to create and support a plan for using work resources in IT tasks based on SGD.

The new idea here is using machine learning ways to solve an old project management issue, which makes things more effective and adaptable [13]. To check if the plan works well, a test study was done where the suggested method was looked at against other methods on made-up data sets.

## 2. Problem statement

Labor resources management in IT projects represents one of the most complex challenges, requiring a delicate balance among numerous interdependent factors [14]. Traditional approaches, which often rely on static models, prove to be ineffective within the context of the modern IT environment [15]. The core issue lies in the inability of these methods to adequately account for the unique characteristics inherent to IT projects (Table 1).

**Table 1**
Unique characteristics of IT projects

| Characteristic | Description |
|---|---|
| Dynamism and Uncertainty | Project requirements may change at any time, necessitating continuous adaptation and task reallocation – something that static models struggle to handle effectively |
| Competencies Specificity | Each team member (devProgram Evaluation and Review Technique, PERT eloper, tester, devops-engineer) has a unique skill set and varying levels of qualifications. Improper task assignment may lead to reduced quality or significant increases in execution time |
| Task interdependence | Many tasks in IT projects are either sequential or parallel in nature, forming a complex web of dependencies |
| Multi-criteria optimization | There is a need to simultaneously minimize project cost, total execution time, and workload imbalance within the team. Such criteria often conflict with one another |

Thus, there emerges a pressing need for the development of a novel, flexible, and scalable algorithm capable of addressing this multifactor optimization problem, adapting to changing conditions, and ensuring the most efficient allocation of labor resources within dynamic IT environment circumstances. The absence of such a tool results in significant financial and temporal losses, while also exerting a detrimental effect on team morale and overall productivity [6, 16].

## 3. Literature review and analysis of existing approaches

Resource management in projects is a long-standing problem, with its origins tracing back to the era of the Industrial Revolution. Over time, numerous approaches have been developed, which can be broadly categorized into classical and modern methods.

Classical methods primarily focus on the sequential execution of tasks and rigid planning structures. These approaches are well-suited for projects with clearly defined requirements and predictable processes, where the environment remains relatively stable throughout the project lifecycle [17, 18] (Table 2).

**Table 2**
Classical methods

| Method name | Description |
|---|---|
| Gantt Chart | This tool visualizes the project plan by displaying tasks along a timeline. While it assists in tracking progress, it does not account for task dependencies, resource workload, or potential risks associated with resource reallocation. |
| Critical Path Method | This method identifies the longest path within a network of dependent tasks, which determines the minimum project duration. The Critical Path Method (CPM) aids in focusing on key tasks; however, it does not consider resource constraints and lacks flexibility in response to changes. |
| Program Evaluation and Review Technique, PERT | Unlike CPM, the Program Evaluation and Review Technique (PERT) accounts for uncertainty in task durations by incorporating three estimates: optimistic, pessimistic, and most likely. This allows for a more realistic assessment of project timelines. However, similar to CPM, PERT does not address the issue of optimizing resource allocation. |

These methods are foundational. However, their inherent static nature renders them poorly suited to the dynamic IT environment, where change is the norm rather than the exception. The main gap is that classical methods only provide high-level time planning, but do not offer a detailed mechanism for micro-task assignment taking into account multi-factor criteria such as qualification matching and total cost optimization.

Contemporary methodologies seek to overcome the limitations of classical approaches by employing more flexible and computationally advanced tools [16, 19] (Table 3).

Although SGD is not a traditional project management tool, it is particularly well-suited for addressing the resource allocation problem. SGD operates with a loss function that measures the "suboptimality" of the current solution. It iteratively adjusts model parameters (in this case—the task distribution), moving in the direction that minimizes this function [20].

In contrast to full gradient descent, SGD updates parameters after processing only a single data point (or a small batch), making it computationally efficient and enabling adaptation to project changes without requiring a complete recalculation. This property is especially valuable in agile environments, where new tasks and shifting priorities arise continuously [21].

Based on the analysis of existing approaches, it becomes evident that traditional methods are outdated for the dynamic IT environment. While modern heuristic and genetic algorithms can be effective, they are often complex to configure [22]. The application of SGD, by contrast, enables the

development of a flexible and scalable system capable of adapting efficiently to changes and optimizing resource allocation in real time.

**Table 3**
Contemporary approaches

| Method name | Description |
|---|---|
| Linear programming | This mathematical approach makes it possible to determine the optimal solution for a problem with multiple constraints. It can be applied to minimize cost or execution time; however, its efficiency decreases significantly as complexity and the number of variables increase—a common feature of large-scale IT projects. |
| Heuristic algorithms | These methods aim to find a "good enough" solution rather than a perfect one. Examples include ant colony optimization and simulated annealing. They are fast and capable of handling large datasets, yet their performance is highly task-dependent, and they do not always guarantee optimality. |
| Genetic algorithms | This evolutionary approach simulates the process of natural selection. It generates random solutions, evaluates their effectiveness, and “crosses” the best ones to create a new generation. While this is a powerful tool for solving complex optimization problems, it can be computationally expensive and requires careful parameter tuning. |
| Machine learning methods | The application of machine learning to project management represents a relatively new research direction. Algorithms such as neural networks and support vector machines can analyze historical data to forecast timelines and costs. Stochastic Gradient Descent (SGD) is one of the key optimization algorithms underlying many modern machine learning models. Its main advantage lies in the iterative approach to finding an optimum, making it particularly effective for large-scale and dynamic systems. |

## 4. Mathematical model of the resource allocation problem

To formalize the problem of labor resource allocation in IT projects, we define the key components and the interrelationships among them. The model is oriented toward minimizing an objective function that simultaneously incorporates multiple criteria [14].

### 4.1. Problem formalization

Let us consider a set of tasks in the project:

$$T = \{t_1, \ldots, t_i, \ldots, t_n\}, \quad (1)$$

where $n$ denotes the total number of tasks in the project.
Each task $t_i$ is characterized by the following attributes:

1. Estimate duration $d_i$ (in hours or person-days).
2. Required qualification level $S_{req_i}$ (represented as a skill vector that specifies the competencies necessary for the executor) [1, 11].

3. Dependencies: a set of predecessor tasks $P_i$. Task $t_i$ cannot begin until all tasks in $P_i$ are completed.

The set of executors involved in project task implementation can be represented as [4]:

$$E=\{e_1,\dots,e_j,\dots,e_m\}, \tag{2}$$

where $m$ denotes the total number of team members.

Each executor $e_j$ is characterized by:

1. Skill set $S_{e_j}$ (a vector analogous to $S_{req_i}$, describing the list of skills and qualifications of each project team member).
2. Work cost $c_j$ of a labor resource per unit of time (e.g., per hour).

The allocation of resources to project tasks can be represented by a matrix $X$, where each element $X_{ij}$ is a binary variable defined as:

1. $X_{ij}$ = 1, if task $t_i$ is assigned to executor $e_j$.
2. $X_{ij}$ = 0 otherwise.

### 4.2. Constraints

For the solution to be valid, it must satisfy the key conditions imposed on project tasks [23, 24]:

Each task must be assigned to exactly one executor:

$$\sum_{j=1}^{m} X_{ij}=1, \quad \forall i\in\{1,\dots,n\}. \tag{3}$$

The executor $e_j$ can be assigned a task $t_i$ only if their skills meet the minimum requirements of that task:

$$X_{ij}=1\Rightarrow S_{e_j}\geq S_{req_i}, \quad \forall i\in\{1,\dots,n\}, \quad \forall j\in\{1,\dots,m\} \tag{4}$$

(Here, the $\geq$ is understood element-wise, meaning that each component of the executor's skill vector $S_{e_j}$ must be greater than or equal to the corresponding requirement for the task $S_{req_i}$).

The start time of task $t_i$ (denoted as $S_i$) must not be earlier than the completion times of all its predecessor tasks:

$$S_i\geq \max_{t_k\in P_i}(S_k+D_k), \quad \forall i\in\{1,\dots,n\}, \tag{5}$$

where $D_k$ – real execution time of task $t_k$.

### 4.3. Loss Function

Our objective is to find an allocation matrix $X$ that minimizes the composite objective function $L(X)$. This function is the sum of three components representing the principal optimization criteria:

$$L(X)=\alpha*L_{cost}(X)+\beta*L_{time}(X)+\gamma*L_{load}(X), \tag{6}$$

where $\alpha$ ,$\beta$ ,$\gamma$ are weighting coefficients that determine the priority of each criterion (for example, if cost is more important than time, $\alpha$ will be greater).

The cost component $L_{cost}$ is calculated as the total project cost:

$$L_{cost}(X)=\sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}*c_j*d_i. \tag{7}$$

This is the total sum of the costs of executing all assigned tasks.

The time component $L_{time}$ is defined as the overall project duration, which corresponds to the completion time of the last task:

$$L_{load}(X) = \max_{i \in n}(S_i + D_i). \quad (8)$$

Time starts $S_i$ and duration $D_i$ are computed based on the allocartion matrix $X$ and the precedence constraints of the tasks. The critical path forms the basis for determining the overall project duration.

The load component $L_{load}$ reflects the imbalance of team workload. This criterion helps prevent situations where one executor is overloaded while others remain underutilized. We define this measue as the variance of workload distribution:

$$L_{load}(X) = \frac{1}{m}\sum_{j=1}^{m}\left(TotalLoad_j - \frac{1}{m}\sum_{k=1}^{m} TotalLoad_k\right), \quad (9)$$

where $TotalLoad_j = \sum_{i=1}^{n} X_{ij} * d_i$ is the total amount of workload assigned to executor $e_j$.

Thus, the problem reduces to finding the allocation matrix $X$, which minimizes the objective $L(X)$ subject to all specified constraints. This model provides flexibility in adjusting priorities (through the weighting coefficients) and serves as the foundation for applying the stochastic gradient descent.

Thanks to its structure, which is geared toward processing small data packets, the SGD algorithm demonstrates high computational efficiency and is a scalable solution that allows you to quickly find an effective distribution of resources even for projects with a large number of variables.

## 5. Resource allocation algorithm based on stochastic gradient descent

Building on the developed mathematical formulations, we propose an algorithm based on the principles of stochastic gradient descent (SGD) [13] for the iterative search of an optimal distribution of tasks among project team members. The algorithm accounts for competence indicators, task prioritization, and the workload of each executor, with the goal of minimizing the composite objective function $L(X)$. The primary advantage of SGD lies in its efficiency and its capacity to handle large-scale data, enabling dynamic adaptation to project changes.

Unlike classical gradient descent, which computes the gradient (the direction of steepest ascent of the function) for the entire dataset, SGD estimates it using a small random subset of data (the so-called mini-batch). This considerably accelerates the optimization process, as each iteration is computationally less expensive [25, 26]. In our case, the "data" correspond to project tasks, while the "parameters" are represented by the allocation matrix $X$.

### 5.1. Algorithm steps

The algorithm is executed iteratively and consists of successive steps (see Fig. 1).

At the initialization stage, an initial allocation matrix $X(0)$ is generated. This can be a random assignment of tasks to executors who satisfy the minimum qualification requirements.

Next, the hyperparameters are defined:

1. Learning rate $\eta$, which controls the step size at each iteration.
2. Number of iterations $K$.
3. Mini-batch size $B$ (the number of tasks processed in each iteration).

The iterative process then proceeds as follows. For k = 1, ..., K:

1. The mini-batch selection is a random subset $B$ of tasks is drawn from the overall task set $T$. This constitutes the “stochastic” element of the algorithm.
2. For each task $t_i$ in the selected mini-batch, the local gradient of the loss function is calculated. The gradient indicates how the assignement (i.e., the values in the allocation

matrix $X$) should be adjusted to locally decrease the objective function. Mathematically, $\nabla L(X)$ is computed for the current mini-batch. In practice, this corresponds to analyzing how reassigning task $t_i$ from executor $e_j$ to executor $e_k$ would affect cost, duration and workload balance.

3. The allocation matrix $X(k)$ is updated based on the computed gradient and the learning rate.

$$X^{(k+1)} = X^{(k)} - \eta * \nabla L\left(X^{(k)}\right). \tag{10}$$

Since $X$ is a binary matrix, the update process is nonlinear. In practice, the values of $X_{ij}$ are not modified fractionally; instead, tasks are reassigned if such a reassignment leads to a reduction in the loss function. For example, if the gradient indicates that reassigning task $t_i$ from executor $e_j$ to executor $e_k$ decreases $L(X)$, then this reassignment is performed.

The stopping criteria for the algorithm are defined as follows:

1. The maximum number of iterations $K$ is reached.
2. The value of the objective function $L(X)$ stabilizes, meaning that further updates no longer yield significant improvements.

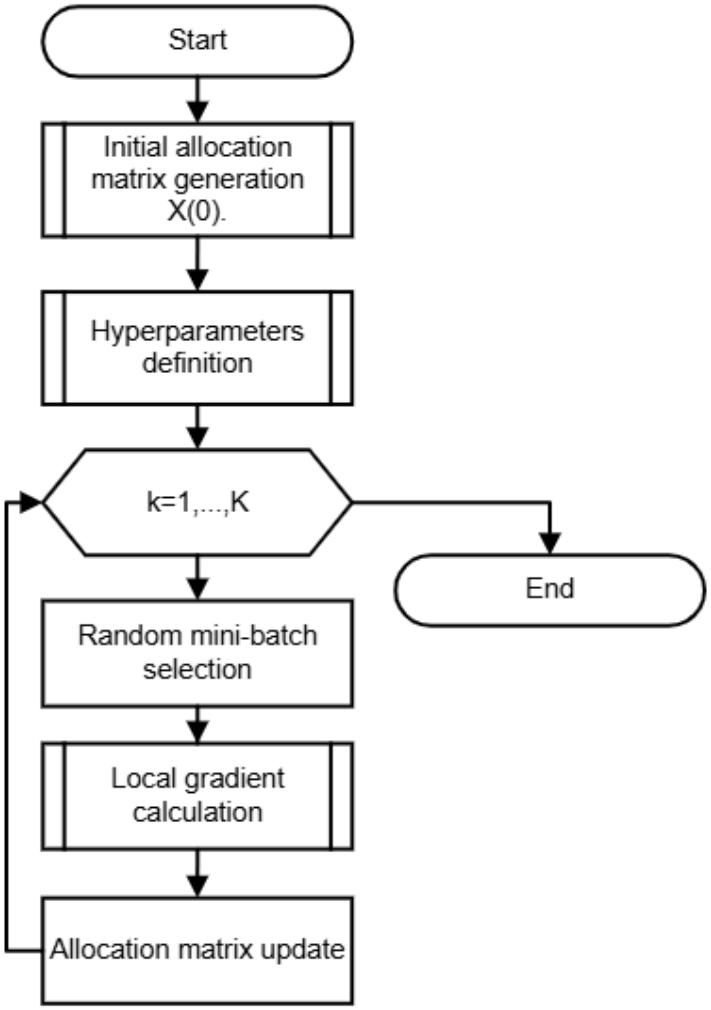


**Figure 1:** Main algorithm steps.

## 5.2. Algorithm implementation

In the general case, the implementation of the proposed algorithm can be illustrated as follows. Suppose the task is to select an executor for project work between two developers: Dev1, who has higher qualifications but also higher cost, and Dev2, who has lower cost but also lower

qualification. The decision concerns the assignment of task T10 (mini-batch with size=1). At the current iteration, the algorithm evaluates the following.

1. The task T10 is initially assigned to Dev1. The cost is high, but the execution time is short.
2. The algorithm computes the gradient by analyzing the potential outcome if T10 were reassigned to Dev2.
3. Reassignment to Dev2 would reduce cost (lower hourly rate) but increase execution time (due to lower qualification).
4. If the overall loss function $L(X)$ considering the weighting coefficients $\alpha$, $\beta$, $\gamma$ decreases after the reassignment, the algorithm performs the change.

Through this iterative process, the algorithm gradually "learns" and discovers increasingly effective allocation strategies, efficiently balancing cost, execution time, and workload distribution within the team.

## 6. Experimental studies and results

To validate the effectiveness of the proposed algorithm based on Stochastic Gradient Descent (SGD), a series of experiments were conducted on simulated datasets. The aim was to compare our approach against two widely used methods:

1. Heuristic algorithm based on a greedy strategy (Greedy Algorithm): this method assigns each task to the executor with the highest qualification and the lowest current workload.
2. Exhaustive search method: suitable for small projects as a benchmark to achieve the ideal solution, although computationally infeasible for large-scale projects due to its exponential complexity.

For the experiment, three datasets were generated to simulate IT projects of different sizes (Table 4).

**Table 4**
Experimental data for validation

| Project type | Number of tasks | Number of executors |
|---|---|---|
| Small project | 20 | 5 |
| Medium project | 50 | 10 |
| Large project | 150 | 25 |

Each task was characterized by its duration, the required skill level, and dependency constraints. Executors were assigned different costs, unique skill sets, and varying initial workloads.

For each dataset, the three algorithms were applied, and the following performance indicators were measured:

1. Total project cost — the aggregate labor expenses.
2. Total project duration — the completion time of the last task.
3. Team workload imbalance — measured as the standard deviation from the mean workload of all executors. The lower this value, the more evenly the tasks are distributed.

For the SGD-based algorithm, 100 iterations were carried out with a mini-batch size equal to 10% of the total number of tasks and a learning rate of $\eta = 0.01$.

The experimental results are summarized in Table 5.

**Table 5**
Experimental results for resource allocation algorithms

| Project type | Algorithm | Cost ($) | Duration (days) | Workload imbalance (days) |
|---|---|---|---|---|
| Small | SGD | 28 500 | 35 | 4,2 |
| | Heuristic | 30 100 | 41 | 6,5 |
| | Exhaustive | 27 900 | 34 | 3,9 |
| Medium | SGD | 67 200 | 78 | 9,1 |
| | Heuristic | 73 800 | 89 | 15,3 |
| Large | SGD | 215 000 | 185 | 18,7 |
| | Heuristic | 240 500 | 210 | 29,8 |

As the results indicate, the SGD-based algorithm demonstrates significantly better efficiency compared to the greedy heuristic approach:

1. In small projects, the results obtained with SGD were very close to the optimal solution derived from exhaustive search, which confirms the high accuracy of the algorithm.
2. In medium and large projects, the advantage of SGD becomes even more pronounced. On average, cost and time savings reached 10–15%, a factor of critical importance for the success of large-scale projects.
3. Particularly noteworthy is the improvement in workload imbalance. The SGD-based approach enabled a 30–40% more balanced distribution of tasks, preventing the overloading of individual executors and thereby enhancing overall productivity and the team's morale.

The experiments demonstrated that the iterative and adaptive nature of SGD makes it more effective than greedy heuristics. The heuristic algorithm, by assigning tasks locally, often fails to account for the overall project landscape, which leads to suboptimal long-term outcomes (e.g., overloading the most qualified specialists). As the size of the project increases, the computational efficiency and low time complexity of SGD becomes a decisive advantage. In contrast, SGD continuously evaluates the impact of each decision on the global loss function, thereby enabling a more balanced allocation of resources.

As project size increases, the computational efficiency of SGD becomes decisive. Although the heuristic algorithm is computationally fast, its solutions become progressively less optimal. Exhaustive search, on the other hand, is computationally infeasible for large projects, which underscores the necessity of employing modern, scalable optimization methods such as SGD.

## 7. Conclusion

The article proposes and substantiates an innovative approach to optimizing labor resource allocation in IT projects based on the Stochastic Gradient Descent (SGD) algorithm. The developed mathematical model makes it possible to formalize a multifactor optimization problem while accounting for the specific requirements of the modern IT environment, such as unique skill sets, task dependencies, and the necessity of balancing cost, time, and team workload [27, 28].

Experimental studies conducted on simulated projects of different scales confirmed the high effectiveness of the proposed approach. Compared to traditional heuristic methods, the SGD-based algorithm demonstrated significant advantages:

1. Reduction of overall project cost and duration by an average of 10–15%.

2. Improved team workload balance, which is critically important for preventing burnout and maintaining high productivity.
3. The ability to efficiently handle large projects, where computationally intensive methods often fail.

Thus, the proposed algorithm represents a powerful tool for project managers, allowing not only the achievement of target performance indicators but also the optimization of the most valuable resource—human capital. The flexibility of SGD enables it to adapt to constant changes, making it an ideal solution for modern agile projects.

Despite these promising results, several directions for further improvement of the model remain [29, 30]:

1. Integration of dynamic learning, through mechanisms that enable the algorithm to learn from data generated in real time, thereby improving predictive accuracy.
2. Consideration of team synergy, by incorporating parameters that capture how productivity depends on the interaction of specific team members.
3. Inclusion of risk-related factors in the objective function to account for uncertainties associated with task assignment.
4. Extension of the model and adaptation of the algorithm to other domains beyond IT, where efficient resource allocation is equally critical.

The application of machine learning methods such as SGD opens new horizons for project management, making it more intelligent, flexible, and efficient [31, 32].

## Acknowledgements

The author team expresses its gratitude to the company manager and the head of the implementation department of company "Test" LLC for the opportunity to conduct experimental research on real project teams involved in the implementation of IT projects.

## Declaration on Generative AI

The authors have used the pseudorandom values generators to generate sample values for experimental calculations (description and qualifications of employees, characteristics of tasks and qualification requirement for their performance). Grammarly was used to translate the article text from Ukrainian to English.

## References

[1] A. Biloshchytskyi , M. Gladka, O. Kuchanskyi, Y. Andrashko, A. Faizullin, Devising a method for building an educational project team based on the analysis of competency matrix. Eastern-European Journal of Enterprise Technologies, 6 (2024) 47–57. doi:10.15587/1729-4061.2024.316782.

[2] V. Druzhynin, M. Gladka, I. Borysenko, Y. Hladkyi, R. Lisnevskyi, A Model for allocating labor resources to project work based on task prioritization, in: Proceedings of the XI International Scientific Conference Information Technology and Implementation, IT&I ’2024, CEUR Workshop Proceedings, Aachen, Germany, 2024, pp. 42–54.

[3] J. Storey, P. M. Wright, Strategic human resource management: A research overview, 2nd. ed., Routledge, New York, NY, 2023.

[4] M. Gladka, O. Kuchanskyi, M. Kostikov, R. Lisnevskyi, Method of allocation of labor resources for IT project based on expert assessements of delphi, in: Proceedings of the International Conference on Smart Information Systems and Technologies, SIST ’2023, IEEE, New York, NY, pp. 545–551. doi:10.1109/SIST58284.2023.10223549.

[5] E. W. Larson, C. F. Gray, Project management: The managerial process, 7th. ed., McGraw-Hill Education, New York, NY, 2018.
[6] PMI, A guide to the project management body of knowledge (PMBOK® Guide), 6th. ed., Project Management Institute, Newtown Square, PA, 2016.
[7] HBR guide to project management, HBR Guide Series, Ingram Publisher Services, Boston, MA, 2016.
[8] J. Peng, X. Liu, Labor resource allocation under extremely short construction period based on the inverse optimization method, Eng. Constr. Archit. Manag. 31 (2024) 1254–1271. doi:10.1108/ECAM-06-2022-0604.
[9] H. Golabchi, A. Hammad, Estimating labor resource requirements in construction projects using machine learning, Constr. Innov. 24 (2024) 1048–1065. doi:10.1108/CI-11-2021-0211.
[10] J. Grabis, B. Haidabrus, E. Druzhinin, K. Kolesnikova, Deployment and release management process in agile digital projects, in: Proceedings of the 7th International Conference on Design, Simulation, Manufacturing: The Innovation Exchange, DSMIE '2024, Springer Cham, Cham, Switzerland, 2024, pp. 124–136. doi:10.1007/978-3-031-61797-3_11.
[11] M. Gladka, O. Kravchenko, Ya. Hladkyi, Sh. Borashova, Qualification and appointment of staff for project work in implementing IT systems under conditions of uncertainty, in: Proceedings of the International Conference on Smart Information Systems and Technologies, SIST '2021, IEEE, New York, NY, 2021, pp. 278–282. doi:10.1109/SIST50301.2021.9465897.
[12] P. Toulis, E. Airoldi, Asymptotic and finite-sample properties of estimators based on stochastic gradients, arXiv preprint arXiv:1408.2923 (2014). doi:10.48550/arXiv.1408.2923.
[13] H. Zhu, B.-G. Hwang, J. Ngo, J. P. S. Tan, Applications of smart technologies in construction project management, J. Constr. Eng. Manag. 148 (2022). doi:10.1061/(ASCE)CO.1943-7862.0002260.
[14] M. Gladka, A. Kuchansky, R. Lisnevskyi, Teamsformation for it projects implementation on the basis of the model of limited rationality, Manag. Dev. Complex Syst. 48 (2021) 17–23. doi:10.32347/2412-9933.2021.48.17-23.
[15] S. Morcov, L. Pintelon, R. Kusters, Definitions, characteristics and measures of IT project complexity – a systematic literature review, Int. J. Inf. Syst. Proj. Manag. 8 (2020). doi:10.12821/ijispm080201.
[16] H. Kerzner, Project management: A systems approach to planning, scheduling, and controlling, 14th. ed., John Wiley & Sons, Inc., Hoboken, NJ, 2025.
[17] H. Kerzner, Innovation project management: Methods, case studies, and tools for managing innovation projects, 1st. ed., John Wiley & Sons, Inc., Hoboken, NJ, 2023.
[18] D. Ciric Lalic, B. Lalic, M. Delić, D. Gracanin, D. Stefanovic, How project management approach impact project success? From traditional to agile, Int. J. Manag. Proj. Bus. 15 (2022) 494–521. doi:10.1108/IJMPB-04-2021-0108.
[19] T. Zorić, V. Makitan, E. Brtka, Modern technologies in IT project management, in: Proceedings of the Engineering Management Conference, EMC '2021, Technical Faculty "Mihajlo Pupin", Zrenjanin, Serbia, 2021, pp. 124–132.
[20] Q. Li, C. Tai, E. Weinan, Stochastic modified equations and dynamics of stochastic gradient algorithms I: Mathematical Foundations, arXiv preprint arXiv:1811.01558 (2018). doi:10.48550/arXiv.1811.01558.
[21] B. Gess, S. Kassing, V. Konarovskyi, Stochastic modified flows, mean-field limits and dynamics of stochastic gradient descent, arXiv preprint arXiv:2302.07125 (2023). doi:10.48550/arXiv.2302.07125.
[22] H. Elkholosy, R. Ead, A. Hammad, S. AbouRizk, Data mining for forecasting labor resource requirements: a case study of project management staffing requirements, Int. J. Constr. Manag. 24 (5) (2024) 561–572. doi:10.1080/15623599.2022.2112898.
[23] S. Wawak, Enhancing project quality through effective requirements management, Int. J. Qual. Res. 19 (2024) 151–168. doi:10.24874/IJQR19.01-10.

[24] J. Leong, K. May Yee, O. Baitsegi, L. Palanisamy, R. K. Ramasamy, Hybrid project management between traditional software development lifecycle and agile based product development for future sustainability, Sustainability 15 (2023) 1121. doi:10.3390/su15021121.
[25] S. Bhatnagar, H. L. Prasad, L. A. Prashanth, Stochastic recursive algorithms for optimization: Simultaneous perturbation methods, Springer, London, 2013.
[26] J. C. Chen, Y.-Y. Chen, T.-L. Chen, Y.-H. Lin, Multi-project scheduling with multi-skilled workforce assignment considering uncertainty and learning effect for large-scale equipment manufacturer, Comput. Ind. Eng. 169 (2022). doi:10.1016/j.cie.2022.108240.
[27] Y. Shastri, R. Hoda, R. Amor, The role of the project manager in agile software development projects, J. Syst. Softw. 173 (2021). doi:10.1016/j.jss.2020.110871.
[28] Y. Andrashko, O. Kuchanskyi, A. Biloshchytskyi, O. Pohoriliak, M. Gladka, G. Slyvka-Tylyshchak, D. Khlaponin, A. Chychkan, A method for assessing the productivity trends of collective scientific subjects based on the modified PageRank algorithm, East.-Eur. J. Enterp. Technol. 1 (2023) 41–47. doi:10.15587/1729-4061.2023.273929.
[29] A. Neftissov, A. Sarinova, I. Kazambaev, L. Kirichenko, R. Lisnevskyi, L. Rzayeva, Development of the smart office concept, in: Proceedings of the International Conference on Smart Information Systems and Technologies, SIST ’2023, IEEE, New York, NY, 2023, pp. 382–386. doi:10.1109/SIST58284.2023.10223512.
[30] G. Liang, L. Xu, L. Chen, Optimization of enterprise labor resource allocation based on quality optimization model, Complexity 2021 (2021). doi:10.1155/2021/5551762.
[31] J. L. Brewer, K. C. Dittman, Methods of IT project management, 4th. ed., Purdue University Press, West Lafayette, IN, 2018.
[32] V. Minni, Making the invisible hand visible: Managers and the the allocation of workers to jobs, IZA Discussion Paper No. 16853, University of Chicago, CEPR and IZA, 2025. URL: https://www.jstor.org/stable/pdf/resrep72224.pdf.