Proceedings of the Creative Construction Conference (2025)
Edited by: Miroslaw J. Skibniewski & Miklós Hajdu & Žiga Turk
https://doi.org/10.22260/CCC2025/0073

OPTIMUM SOLUTION OF RESOURCE LEVELING PROBLEM BY COMPLETE ENUMERATION

Önder Halis Bettemir[1]
1 İnönü University, Malatya, Türkiye, onder.bettemir@inonu.edu.tr

Abstract

Solution of resource leveling problem is conducted with meta-heuristic algorithms, mathematical programming, and branch and bound algorithms. However, none of the algorithms can guarantee providing the exact solution of the problem when the number of activities is more than 60. In this study, solution of resource leveling with complete evaluation is implemented and successful results are obtained. The automated complete evaluation process requires automated formation of activity-on-arc type project network; detection of parallel paths and float durations of the activities; formation of schedule combinations by considering the paths and float durations. Activity-on-arc type diagram is formed by implementing the connection of each predecessor with dummy activities and deleting the unnecessary dummies by complete analysis of the network. Activity-on-arc diagram requires less logical checks than activity-on-node type diagram to detect activity clashes due to activity delays. Each path of the network and the critical paths are detected by chaining the nodes of the diagram. The noncritical activities are delayed by nested for loops which are generated automatically according to the number of noncritical activities and the total float durations of the activities. The automated for loops scan the whole search domain of the problem and provide the guaranteed optimum solution. The memory requirement of complete evaluation is within one megabyte since each variable is integer and the whole parameters can be stored at the buffer memory of the central processing unit. This property significantly shortens the computational time for one schedule evaluation compared with population based meta-heuristic algorithms are the buffer memory cannot be adequate and random access memory, which has longer access time, is utilized. Consequently, the complete evaluation process is completed within shorter computation time even tough more schedule evaluations are executed. Moreover, complete evaluation guarantees obtaining the best solution since it evaluates all of the feasible solutions.

Keywords: activity on arc diagram, optimization, resource leveling, scheduling.

Peer-review under responsibility of the scientific committee of the Creative Construction Conference 2025.

1. Introduction

1.1. Resource Levelling

Preliminary construction schedule which is formed by the early start times of the activities may not provide a proper resource allocation. The initial resource distribution may fluctuate and have high peak resource demand which increases the construction cost. More appropriate resource distribution is searched by delaying the noncritical activities without causing elongation in the project completion time. Aforementioned endeavour is called resource levelling. Proper resource distribution prevents the hiring and firing of workers on a short-term basis, ensuring stable employment and assisting in distinguishing and retaining proper workers [1]. It also provides suitable working conditions for the proper learning curves of construction workers and prevents low resource demands [2]. Furthermore, it prevents idle workers [3], unproductive and overcrowded labour at the construction site [4], and decreases management overhead of the resource profile [5]. The cited factors have made research on obtaining the optimum solution of the resource levelling problem to become an important field of study in construction management and industrial engineering.

Corresponding author email address:

1.2. Literature review on the solution of resource levelling problem

Solution of the resource levelling problem is mainly done by heuristic algorithms, meta-heuristic algorithms, and mathematical programming methods.

Harris presented the well-known heuristic rule, Packing Method (PACK) [6]. Packing method is enhanced by introducing modified minimum moment method [7]. Gordon and Tulip proposed heuristic rules for resource levelling [8]. Hiyassat modified the minimum moment to reduce the computational demand [9, 10]. A hyper-heuristic method, containing several heuristic rules for the solution of combined resource allocation and levelling problem is developed [11]. Better results with heuristic network analysis algorithms are obtained [12]. Heuristic methods are based on certain rules or sets of constant algorithms aiming to obtain a reasonable solution. For this reason, they provide proper solutions for only particular problems and heuristic algorithms cannot guarantee obtaining global optimum for every project.

Mathematical Integer Programming is also utilized however; the whole 60 Activity sets problems had not been solved in 2015 almost 30 years after the formation of the hypothetic problems [13]. Branch-price-and-cut algorithm is implemented to solve resource levelling problem [13]. The initial attempts to overcome this difficulty included developing heuristic rules to narrow the search domain. A heuristic approach which focuses on the high peak regions to prevent excessive increase in the number of variables is utilized to solve RLP by Linear Programming (LP) [14].

Karaa and Nasr proposed a linear-programming based resource-leveling algorithm, which provides the least leasing cost [15]. Takamoto et al. implemented quadratic programming approach [16]. RLP is also solved by integer-linear programming [17, 18]. Mixed-integer programming is also implemented [19-22]. Rieck et al. obtained exact solutions up to 50-Activity projects [19]. Branch-and-bound algorithm is also employed for the solution of RLP [23-26]. Bandelloni et al. developed a non-serial dynamic programming algorithm [27]. However, RLP is NP-hard and the increment of the number of variables causes significant expansion of the search domain [19, 28]. The number of feasible solutions that a serial path produces is computed by Eq. 1 [29]. The factorial relationship causes tremendous expansion of the search domain when the number of activities of a serial path increases.

$$\frac{1}{m!}\prod_{i=1}^{m}\left(n+i\right) \tag{1}$$

In Eq. 1 *m* is the number of activities, and *n* is the total float duration. Exact solution of large projects is not feasible due to high memory allocation and long computation time; therefore exact algorithms can present optimum solution for small and medium-size projects.

Memory and CPU demands of meta-heuristic algorithms are reasonable for optimum or near-optimum solutions for large problems. Consequently, meta-heuristic algorithms are also utilized for the optimum or near-optimum solution of the RLP. Simulated Annealing (SA) [30]; Genetic Algorithm (GA) [31]; Differential Evolution [32]; Memetic Algorithm [33] are applied for the solution of RLP. Ant Colony Optimization (ACO) is implemented [34]. Modified symbiotic organisms search algorithm provided the best solution among JADE, SaDE, SOS, and RLDE meta-heuristic algorithms [35]. The test problem was 42-Activity single-resource project adapted from [36] whose global optimum is not known. GA is implemented to optimize resource levelling of a 7-activity linear project in which two activities have float durations [37]. The search domain was stated as 3000 while, GA executed 50,000 schedule runs.

1.3. Research goals and objections

Literature review revealed that successes of heuristic algorithms are problem specific. Exact algorithms are not suitable for the large construction projects since they can guarantee obtaining global optimum at most 50 activity problems due to computational demand and memory allocation problems [19]. Therefore, complete evaluation is implemented for the solution of RLP with 5-Activity multi-resource levelling problem [38]. Search domain is computed as 8 for the problem. Erzurum and Bettemir solved RLP by complete enumeration [29, 39].

Complete enumeration has a potential to be a solution alternative of the resource levelling problem as the aforementioned technique can guarantee obtaining the optimum solution and pareto front solution set. Moreover, it does not require generation of a population so that the whole variables can be stored at the buffer memory of the CPU. This avoids memory transfer between the RAM and CPU and the computations are completed in a shorter time period. However, complete enumeration requires determination of the search domain of the problem which is a time consuming and tedious task especially for complex network diagrams. Moreover, drawing the network diagram and detecting the paths of the network as well as preparation of for loops to scan all the feasible schedule alternative cause difficulties for the implementation of the complete enumeration.

This study aims to develop algorithms and software which can form Activity-on-Arrow (AoA) type network diagram, detect each path in the diagram, construct nested for loops for the delay of each noncritical activity, and provide the optimum solution of the resource levelling problem.

2. Methodology

The automated formation of activity-on-arrow diagram, detection of each path of the network and automated construction of nested for loops are explained in this section.

2.1. Automated formation of AoA diagram

Formation of AoA diagram by considering the logical relationships between the activities is performed by the principles proposed by [2]. The adapted method is briefly explained in bullet point form.

1. Detect the activities with no predecessor activity and draw them on the left side.
2. Detect the activities with no successor activity and draw them on the right side.
3. Draw the remaining activities at the middle portion.
4. Add start event and end event nodes to the all of the activities.
5. Activity relationships are shown by only dummy activities. Draw dummy activities to represent activity relationships.
6. Examine all of the dummy activities and detect their necessity.
    a. If no activity other than a dummy activity starts from the start event, then that dummy is unnecessary and the dummy activity is terminated, terminating the start event.
    b. If no other activity is connected to the end event of a dummy activity, then that dummy activity is unnecessary. The dummy activity is terminated, terminating the end event.
    c. Determine the end events of activities that do not have a successor activity. If there is more than one end event, merge the end events so that the one with the highest event ID number is retained.

The listed items are coded in C++ programming software which can form AoA diagram. The network diagram is only form and it is not drawn since visual programming functions were not utilized. The start and end nodes of the activities and the dummy activities are listed.

2.2. Detection of the paths of the network

In order to detect the existing paths of the network containing critical and noncritical activities is conducted by the developed algorithm which is presented as bullet point form.

1. Identify activities with starting event number of zero. Record all identified activities as the first activity in a separate path. Set the activity index values of all paths equal to one.
2. Write the end event ID numbers of the activities to the (index +1) element of the path index. Increment the index value by one.
3. Find the activities that have the same starting index number as the ending index number of the last added activities in the activity list for all the incomplete paths registered in the path directory.

a. If there is one activity match: This means the path is straight. Add the end event of the matching activity to the list as an (index + 1st) element of the corresponding path.

b. If there are more than one activity match: This situation indicates that the path has forked.

c. Increase the number of paths by the number of forking paths. Copy the path by the number of forking paths. Add the ending indices of the matching activities to the (index +1st) element of the path. Increase the index value by one.

4. Repeat item 3 until the end event ID number of the final activity is equal to the finish event ID.

Presented algorithm is suitable for the AoA type networks. The formed path chain includes the ID numbers of the event starting from the start event and ending at the final event. The paths also include the dummy activities if they exist. The paths are necessary to detect the interactions between the noncritical activities that affect each other due to the activity delay durations.

2.3. Formation of automated nested loops

Complete enumeration is based on the visiting all of the feasible scheduling combinations which are obtained by delaying the noncritical activities. In order to try all of the possible schedule combination each noncritical activity should be delayed from zero to its total float value. Number of noncritical activity is not a constant parameter, therefore a predefined number of nested for loops cannot be used for a solution. A recursive function is written in C++ which forms for loops and this function is recalled recursively until the “for loops” of the all noncritical activities are called. The important steps of the recursive “for loop” algorithm is presented in terms of bullet point form.

1. Determine the EET and LET time interval values of non-critical activities.
2. Values determined in the first list item will be used for the initial value and the stopping value of the for loop.
3. Create for loops as many times as the number of non-critical activities.
   a. The loop counter assigns how many days the non-critical activity is delayed.
   b. Determine the finish time of the latest completed activity among the predecessor activities.
   c. If the start time of the corresponding activity is earlier than the time detected in item 3.b, the loop counter is set so that the start day of the activity is not later than the specified predecessor end day.
   d. The start and end times of all activities are determined according to the values assigned within the “for loop”.
   e. Daily resource distribution is computed.
   f. Implement the resource distribution metric and increment the evaluation counter. Update the current best solution if a better solution is obtained.
   g. Continue “for loop” until the stopping criteria of the all loops are met

Given algorithms are programmed in C++ language and they are implemented on resource levelling problems derived from the literature.

3. Case Study Problems

The proposed algorithms are tested on two resource levelling problems derived from the literature.

3.1. Case study problem 1

In Table 1 activity and predecessor list of the first problem is given.

Table 1. First case study problem [17].

Activity	Predecessor	Duration	Resource
A	-	2	10
B	A	3	6
C	-	1	2
D	A	1	4
E	C, D	1	2

First test problem consists of five activities. The attributes given in Table 1 are written to a text file which is read by the developed software. The first case study problem is relatively easy as it has only 11 feasible schedule alternatives. The formation of AoA diagram, paths, and complete enumeration are completed in 0.063 seconds. The best solution is obtained as 428 when minimum moment resource distribution metric is implemented.

In order to test the success of the proposed method the same problem is solved by Genetic Algorithm with 50 schedule evaluations. The optimization process is repeated 10 times and in the average 431.2 objective function value is obtained. The optimization process is performed by conducting 250 schedule evaluations and 429.6 is obtained which is worse than the result obtained by complete evaluation.

3.2. Case study problem 2

Second case study problem consists of nine activities which end up with 272 feasible schedule evaluations. The attributes of the project is given in Table 2. The formation of AoA diagram, paths, and complete enumeration are completed in 0.406 seconds. The best solution is obtained as 2700 when minimum moment resource distribution metric is implemented.

Table 2. Second case study problem [34].

Activity	Predecessor	Duration	Resource
A	-	7	5
B	-	3	6
C	A, B	7	7
D	A	5	8
E	A	4	6
F	C, D	2	2
G	D	3	4
H	D, E	6	3
I	F, G, H	5	4

The same problem is solved by GA by repeating the optimization process 10 times with 150 schedule evaluations at each trial. The average of the optimization process is obtained as 2704. The same problem is solved by 500 schedule evaluation and the average of the obtained results is 2702.8. The complete enumeration provided better results with less schedule enumeration for small problems.

4. Discussion of Results

Solution of resource levelling problem by complete enumeration technique provided better results than Genetic Algorithm in shorter computation time. Minimum Moment (MM) metric is utilized in order to evaluate the resource distribution. MM may provide the same penalty value for different resource distribution and causes several global optimum solutions [40]. Even though the aforementioned situation, GA could not find the optimum solution.

The reason of this outcome is due to the population based nature of the meta-heuristic algorithm. Formation of population requires larger memory requirement than the complete enumeration which increases the amount of data transfer between the CPU and the RAM so it slows down the execution of the computation. Moreover, cache memory of the CPU is accessed faster than RAM and data transfer significantly slows down if the data is stored on RAM instead of cache memory of the CPU. Therefore, one schedule evaluation is executed in a shorter time with complete enumeration.

The developed software forms the AoA diagram automatically. Moreover, the developed algorithm successfully scanned the whole search domain with recursive for loops. The aforementioned automation significantly reduces the human endeavour to detect the paths and the boundaries of the loops. The obtained solutions are compared with the hand solutions and manual search domain computations and checked that both the obtained solutions and the visited search domains are correct. However, the utilized test problems are not complex enough to prove the success of the developed solution this is because the test problems are small thus the search domain is not large enough; activities of the test problem have at most two predecessors and complex activity relations do not exist. In order to achieve a complete validation of the developed algorithms larger projects with more complex activity relationships should be tested.

Computation time of the complete enumeration process can be shortened by parallel computing. The enumeration process can be divided into smaller but equal sized portions by analysing the paths of the network. The divided portions can be solved at different cores of the CPU and important time savings can be obtained. Moreover, preparation of the resource histogram can be modified in order to reduce the computational demand. In complete enumeration, “for loop” increments the delay duration of an activity by one day at each loop. This situation only changes position of one activity. Consequently, instead of preparation of the whole resource diagram from the beginning, only the days that the delayed activity had left and had arrived can be considered to calculate the objective function. The aforementioned modification can reduce the computation time significantly in long projects.

References

[1] R. B. Harris, “Precedence and arrow networking techniques for construction”, 1st Ed., John Wiley and Sons, New York, N.Y, 1978, "the referenced item does not yet have a DOI number."

[2] J. D. Stevens, “Techniques for Construction Network Scheduling”, McGraw-Hill, New York, 1990, "the referenced item does not yet have a DOI number."

[3] D. H. Jun and K. El-Rayes, “Multi-Objective Optimization of Resource Scheduling in Construction Projects,” Construction Research Congress 2009, pp. 806–815, Apr. 2009, doi: 10.1061/41020(339)82.

[4] J. L. Ponz-Tienda, A. Salcedo-Bernal, and E. Pellicer, “A Parallel Branch and Bound Algorithm for the Resource Leveling Problem with Minimal Lags,” Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 6, pp. 474–498, Nov. 2016, doi: 10.1111/mice.12233

[5] G. Waligora, “Genetic Algorithm for Resource Leveling Problems under Various Objective Functions,” 2022 26th International Conference on Methods and Models in Automation and Robotics (MMAR), Aug. 2022, doi: 10.1109/mmar55195.2022.9874286.

[6] R. B. Harris, “Packing Method for Resource Leveling (Pack),” Journal of Construction Engineering and Management, vol. 116, no. 2, pp. 331–350, Jun. 1990, doi: 10.1061/(asce)0733-9364(1990)116:2(331)

[7] J. Martinez and P. Ioannou, “Resource Leveling Based on the Modified Minimum Moment Heuristic.” Comp. Civ. Bld. Eng., 287-294, 1993, "the referenced item does not yet have a DOI number."

[8] J. Gordon and A. Tulip, "Resource scheduling." Int. J. Proj. Manage., 15(6), 359-370, 1997, "the referenced item does not yet have a DOI number."

[9] M. A. S. Hiyassat, “Modification of Minimum Moment Approach in Resource Leveling,” Journal of Construction Engineering and Management, vol. 126, no. 4, pp. 278–284, Jul. 2000, doi: 10.1061/(asce)0733-9364(2000)126:4(278).

[10] M. A. S. Hiyassat, “Applying Modified Minimum Moment Method to Multiple Resource Leveling,” Journal of Construction Engineering and Management, vol. 127, no. 3, pp. 192–198, Jun. 2001, doi: 10.1061/(asce)0733-9364(2001)127:3(192).

[11] G. K. Koulinas and K. P. Anagnostopoulos, “Construction Resource Allocation and Leveling Using a Threshold Accepting–Based Hyperheuristic Algorithm,” Journal of Construction Engineering and Management, vol. 138, no. 7, pp. 854–863, Jul. 2012, doi: 10.1061/(asce)co.1943-7862.0000492.

[12] F. Ballestín, C. Schwindt and J. Zimmermann, „Resource leveling in make-to-order production: modeling and heuristic solution method”, International Journal of Operations Research, 4(1), 50-62, 2007. "the referenced item does not yet have a DOI number."

[13] E. T. Coughlan, M. E. Lübbecke, and J. Schulz, “A branch-price-and-cut algorithm for multi-mode resource leveling,” European Journal of Operational Research, vol. 245, no. 1, pp. 70–80, Aug. 2015, doi: 10.1016/j.ejor.2015.02.043

[14] R. C. Leachman, “Multiple resource leveling in construction systems through variation of activity intensities,” Naval Research Logistics Quarterly, vol. 30, no. 2, pp. 187–198, Jun. 1983, doi: 10.1002/nav.3800300202.

[15] F. A. Karaa, and A. Y. Nasr, "Resource management in construction." J. Constr. Eng. Manage., 112(3), 346-357, 1986, "the referenced item does not yet have a DOI number."

[16] M. Takamoto, N. Yamada, Y. Kobayashi, H. Nonaka, and S. Okoshi, “Zero-one quadratic programming algorithm for resource leveling of manufacturing process schedules,” Systems and Computers in Japan, vol. 26, no. 10, pp. 68–76, Jan. 1995, doi: 10.1002/scj.4690261007
[17] S. M. Easa, “Resource Leveling in Construction by Optimization,” Journal of Construction Engineering and Management, vol. 115, no. 2, pp. 302–316, Jun. 1989, doi: 10.1061/(asce)0733-9364(1989)115:2(302)
[18] M. Hariga and S. M. El-Sayegh, “Cost Optimization Model for the Multiresource Leveling Problem with Allowed Activity Splitting,” Journal of Construction Engineering and Management, vol. 137, no. 1, pp. 56–64, Jan. 2011, doi: 10.1061/(asce)co.1943-7862.0000251
[19] J. Rieck, J. Zimmermann, and T. Gather, “Mixed-integer linear programming for resource leveling problems,” European Journal of Operational Research, vol. 221, no. 1, pp. 27–37, Aug. 2012, doi: 10.1016/j.ejor.2012.03.003
[20] S. Kreter, J. Rieck, and J. Zimmermann, “The total adjustment cost problem: Applications, models, and solution algorithms,” Journal of Scheduling, vol. 17, no. 2, pp. 145–160, Aug. 2013, doi: 10.1007/s10951-013-0344-y.
[21] J. Rieck and J. Zimmermann, “Exact Methods for Resource Leveling Problems,” Handbook on Project Management and Scheduling Vol.1, pp. 361–387, Oct. 2014, doi: 10.1007/978-3-319-05443-8_17.
[22] L. Bianco, M. Caramia, and S. Giordani, “Resource levelling in project scheduling with generalized precedence relationships and variable execution intensities,” OR Spectrum, vol. 38, no. 2, pp. 405–425, Feb. 2016, doi: 10.1007/s00291-016-0435-1.
[23] K. Neumann and J. Zimmermann, “Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints,” European Journal of Operational Research, vol. 127, no. 2, pp. 425–443, Dec. 2000, doi: 10.1016/s0377-2217(99)00498-1.
[24] T. Gather, J. Zimmermann, and J.-H. Bartels, “Exact methods for the resource levelling problem,” Journal of Scheduling, vol. 14, no. 6, pp. 557–569, Nov. 2010, doi: 10.1007/s10951-010-0207-8.
[25] K. Neumann and J. Zimmermann, “Resource levelling for projects with schedule-dependent time windows,” European Journal of Operational Research, vol. 117, no. 3, pp. 591–605, Sep. 1999, doi: 10.1016/s0377-2217(98)00272-0.
[26] M. Ç. Mutlu, “A branch and bound algorithm for resource leveling problem,” MSc Thesis, METU, 2010, "the referenced item does not yet have a DOI number."
[27] M. Bandelloni, M. Tucci, and R. Rinaldi, “Optimal resource leveling using non-serial dyanamic programming,” European Journal of Operational Research, vol. 78, no. 2, pp. 162–177, Oct. 1994, doi: 10.1016/0377-2217(94)90380-8.
[28] K. Neumann, C. Schwindt, and J. Zimmermann, “Resource-Constrained Project Scheduling — Minimization of General Objective Functions,” Project Scheduling with Time Windows and Scarce Resources, pp. 195–346, 2003, doi: 10.1007/978-3-540-24800-2_3.
[29] Ö. H. Bettemir and T. Erzurum, “Exact Solution of Resource Leveling Problem by Exhaustive Enumeration with Parallel Programming,” Teknik Dergi, vol. 32, no. 3, pp. 10767–10805, May 2021, doi: 10.18400/tekderg.595238.
[30] J. Son and M. J. Skibniewski, “Multiheuristic Approach for Resource Leveling Problem in Construction Engineering: Hybrid Approach,” Journal of Construction Engineering and Management, vol. 125, no. 1, pp. 23–31, Jan. 1999, doi: 10.1061/(asce)0733-9364(1999)125:1(23).
[31] T. Hegazy, “Optimization of Resource Allocation and Leveling Using Genetic Algorithms,” Journal of Construction Engineering and Management, vol. 125, no. 3, pp. 167–175, Jun. 1999, doi: 10.1061/(asce)0733-9364(1999)125:3(167).
[32] H.-H. Tran and N.-D. Hoang, “A Novel Resource-Leveling Approach for Construction Project Based on Differential Evolution,” Journal of Construction Engineering, vol. 2014, pp. 1–7, May 2014, doi: 10.1155/2014/648938.
[33] M. Iranagh, R. Sonmez, T. Atan, F. Uysal, and Ö. H. Bettemir, “A Memetic Algorithm for the Solution of the Resource Leveling Problem,” Buildings, vol. 13, no. 11, p. 2738, Oct. 2023, doi: 10.3390/buildings13112738.
[34] J. Geng, L. Weng, and S. Liu, “An improved ant colony optimization algorithm for nonlinear resource-leveling problems,” Computers & Mathematics with Applications, vol. 61, no. 8, pp. 2300–2305, Apr. 2011, doi: 10.1016/j.camwa.2010.09.058.
[35] D. Prayogo, M.-Y. Cheng, F. T. Wong, D. Tjandra, and D.-H. Tran, “Optimization model for construction project resource leveling using a novel modified symbiotic organisms search,” Asian Journal of Civil Engineering, vol. 19, no. 5, pp. 625–638, Apr. 2018, doi: 10.1007/s42107-018-0048-x
[36] K. Sears, G. Sears, R.Clough, (). Construction Project Management: A Practical Guide to Field Construction Management, 5th edition, John Wiley and Son, NJ, USA, 2008, "the referenced item does not yet have a DOI number."
[37] A. Damci, D. Arditi, and G. Polat, “Resource Leveling in Line-of-Balance Scheduling,” Computer-Aided Civil and Infrastructure Engineering, vol. 28, no. 9, pp. 679–692, Aug. 2013, doi: 10.1111/mice.12038
[38] M. A. Younis and B. Saad, “Optimal resource leveling of multi-resource projects,” Computers & Industrial Engineering, vol. 31, no. 1–2, pp. 1–4, Oct. 1996, doi: 10.1016/0360-8352(96)00116-7
[39] Ö. H. Bettemir and T. Erzurum, “Optimum Solution of Resource Leveling Problem by Evaluating the Whole Search Domain,” Fırat Üniversitesi Mühendislik Bilimleri Dergisi, vol. 34, no. 2, pp. 779–794, Sep. 2022, doi: 10.35234/fumbd.1105174.
[40] Ö. H. Bettemir, “Analysis of Existing Resource Distribution Metrics and Suggestion of New Metrics,” Journal of Construction Engineering and Management, vol. 151, no. 2, Feb. 2025, doi: 10.1061/jcemd4.coeng-15085.