ISSN 0971-0388

Aligarh Journal of Statistics
Vol. 44(2024), 25-50

Methodological Approaches Used For the Google Machine Reassignment Problem

Andrew Ishaku Wreford[1], Isah Charles Saidu[2], Umar Muhammad Modibbo[3], Bolaji Asaju La'aro[4]
[Received on September, 2023. Accepted on January, 2024]

ABSTRACT

The Google Machine Reassignment Problem (GMRP) presents a formidable challenge in combinatorial optimization, initially introduced by Google as part of the ROADEF/EURO 2012 Challenge. This problem revolves around the optimal reallocation of a set of processes to a designated set of machines, adhering to specified soft and hard constraints. These constraints encompass considerations like load balancing, efficient resource utilization, and the minimization of communication overhead. This survey comprehensively examines the extensive literature dedicated to addressing the intricacies of the GMRP. It emphasizes the multitude of algorithms and techniques proposed to grapple with this intricate problem. The survey elucidated the problem's statement and its significance, particularly in contemporary computing landscapes marked by extensive data processing and the prevalence of cloud computing infrastructure. It navigates through the numerous challenges posed by the GMRP, highlighting the inherent complexity arising from the interplay of factors such as machine capacities, process dependencies, and communication constraints. The systematic categorization of diverse approaches is a key feature of the survey. It encompasses a spectrum of optimization techniques: mathematical methods, local search algorithms, population-based algorithms, hyperheuristics, and hybrid metaheuristics. The survey reveals that population-based algorithms emerge as a more viable solution among the various approaches investigated. This conclusion is drawn based on the findings obtained by the authors while conducting the survey.

1. Introduction

In the realm of distributed computing, recent research endeavours have predominantly revolved around the concept of cloud computing. Cloud computing, a contemporary paradigm, has evolved from its predecessors, including grid computing and utility computing (Zhang and Geng, 2022). The term "cloud computing" alludes to an architectural framework for computing and data dissemination, leveraging a distributed network of remote servers (Praveen, Ghasempoor, Shahabi, and Izanloo, 2023). Within this framework, a cloud system manifests as a parallel and distributed system composed of a cluster of Virtual Machines that dispense computational resources per service-level agreements (SLAs), orchestrated between service providers and clients (Liu, 2022). Cloud computing empowers users to access networked hardware, operating systems, and applications on demand, implying a wide spectrum of computing resources encompassing the networks, servers, and databases that can be efficiently provisioned to individuals and businesses at scale via internet-enabled cloud

Corresponding author : Asaju La'aro Bolaji, Computer Science Department, Federal University Wukari, Nigeria.
Email: lbasaju@fuwukari.edu.ng
[1]Computer Science Department, Federal University Wukari, Nigeria.
[2]Department of Computer Science and IT, Baze University, Abuja, Nigeria.
[3]Department of Operations Research, Modibbo Adama University, Yola, Nigeria.

platforms. This, in turn, facilitates cost reduction for businesses (Alduailij, Khan, Tahir, Sardaraz, Alduailij, and Malik, 2022). The cloud resource pool typically comprises numerous hosts, while virtual machines are employed to offer the mentioned resources and services. Cloud computing services are categorized into Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). SaaS offers accessible software applications via the Internet, reducing local installations and maintenance; PaaS provides developers with frameworks for building applications; IaaS offers virtualized computing resources like virtual machines, storage, and networking, enabling flexible infrastructure management (Velliangiri, Karthikeyan, and Vinoth, 2021). Notably, the provisioning of services within the cloud platform entails dynamic adjustments to the available resources of all physical hosts (Balasubramaniam, Vijesh, Sivakumar, Prasanth, Satheesh, Kavitha, and Dhanaraj, 2023). This dynamism precludes a consistent assignment of processes to the host with the available resources. Consequently, task scheduling has emerged as a pivotal challenge in cloud computing, where tasks or processes are allocated to central data sources using scheduling algorithms. However, the diversity of scheduling objectives has prevented the emergence of a singular, definitive scheduling algorithm, leading to the construction of hybrid or amalgamated scheduling strategies. The complexity of a scheduling problem can span seconds, hours, or even years, contingent upon the approach employed, with algorithm efficiency being gauged by its runtime, which typically scales linearly with input complexity (Manikandan, Gobalakrishnan, and Pradeep, 2022). In the context of computational complexity theory, complexity classes group problems with comparable difficulty under a specific context (Guo, 2021).

The effective administration of vast infrastructure is a crucial aspect for cloud computing service providers such as Google and Amazon (Portal, 2012). The primary goal is to obtain and distribute resources to provide high-quality services to clients. Prioritizing optimal resource utilization is of utmost importance since any inefficiencies would require acquiring extra resources to compensate for the disadvantages of suboptimal utilization. Nevertheless, the concept of resource optimization goes beyond the mere maximization of computational capability and includes additional considerations such as system robustness. It is imperative to implement strategies aimed at minimizing the consequences of machine malfunctions, guaranteeing the system's operational effectiveness, even in the event of a total shutdown of the data centre. The increased stress experienced by the system increases the likelihood of malfunctions, which justifies the need for a purposeful effort to enhance the allocation of resources to ensure long-term reliability.

In 2012, Google, a prominent cloud vendor, introduced the Google Machine Re-Assignment Problem (GMRP) as a combinatorial optimization challenge. The GMRP encapsulates key aspects of the aforementioned cloud computing issues. The challenge revolves around a group of machines with distinct available computational resources and pre-assigned processes delivering various services. The central objective entails allocating processes to machines to maximize machine utilization while safeguarding system reliability. Google introduced this formulation of GMRP at the ROADEF/EURO (French Operational Research and Decision Aid Society and the European Operational Research Society) Challenge in 2012 (Roadef/Euro Challenge, 2012). The challenge commenced on June 8th, 2011, with a submission deadline of December 8th, 2011, for the qualification round. Of the 83 teams worldwide that registered and submitted their applications, only 30 teams met the criteria and progressed to the subsequent level.

The main objective or contribution of this study is to provide a complete, if not exhaustive, summary of the algorithm variant applied by several authors for the Google Machine Re-Assignment Problem (GMRP) while providing a fundamental reference for academics who are interested in obtaining a

deeper understanding of the evolutionary path of Google Machine Re-Assignment Problem (GMRP). Furthermore, the report highlights prospective algorithms for further research within the framework of GMRP.

The subsequent sections of this work are structured in the following fashion: Section II provides a comprehensive summary of the hosts of the GMRP competition, together with a detailed description of the data instances utilized. Section III of the document delineates the problem formulation for the GMRP and the constraints that govern the challenge. In Section IV, a thorough examination is performed to examine the suitability of various algorithms utilized by different researchers for the GMRP. Subsequently, Section V undertakes an analysis of the various methodologies employed in the study, leading to the subsequent formulation of the conclusion.

2. ROADEF/EURO and GMRP Problem

The ROADEF/EURO challenge is a collaborative competition jointly organized by the French Operational Research and Decision Aid Society and the European Operational Research Society. Established in 1999, the challenge focuses on enhancing industrial optimization problems (Afsar, Artigues, Bourreau, and Kedad-Sidhoum, 2016). In 2012, Google proposed a subject for the challenge involving the complex task of reallocating processes from one set of computers to another. The goal was to improve machine utilization, minimize relocation costs, and optimize the use of resources like CPU, RAM, and disk. These objectives had to be achieved while adhering to operational constraints. The problem consists of two distinct phases:

i. Qualification phase: During this step, each algorithm was executed using instances of Set A. In this study, a total of 30 highest scores were chosen for each category.

ii. In the last step, the execution of each algorithm was carried out using instances of enormous size; specifically groups B and X. The instances consisted of a range of 5,000 to 50,000 processes and 100 to 5,000 machines.

2.1 Data Instances Description

The GMRP competition, conducted during the ROADEF 2011/2012 event, included 30 instances. Multiple participant groups used these instances to evaluate their methodologies. These instances are available on the challenge's official website and are divided into three groups (ROADEF/EURO Challenge, 2012). The sets comprise cases A, B, and X. The present analysis suggests using examples from sets A, B and X.

i. Set A consists of problem instances that take into account a constraint of 1000 processes.

ii. Set B consists of problem situations characterized by the presence of processes ranging from 5000 to 50,000.

iii. Set X: consists of problem instances with some processes between 5000 and 50,000.

Table 1: Dataset A, B, and X characteristics.

Instance	**|r|**	**|m|**	**|s|**	**|p|**	**|l|**	**|n|**
A 1-1	2	4	79	100	4	1
A 1-2	4	100	980	1,000	4	2
A 1-3	3	100	216	1,000	25	5
A 1-4	3	50	142	1,000	50	50

A 1-5	4	12	981	1,000	4	4
A 2-1	3	100	1,000	1,000	1	1
A 2-2	12	100	170	1,000	25	5
A 2-3	12	100	129	1,000	25	5
A 2-4	12	50	180	1,000	25	5
A 2-5	12	50	153	1,000	25	5
B1	12	100	2,512	5,000	10	5
B2	12	100	2,462	5,000	10	5
B3	6	100	15,025	20,000	10	5
B4	6	500	1,732	20,000	50	5
B5	6	100	35,082	40,000	10	5
B6	6	200	14,680	40,000	50	5
B7	6	4,000	15,050	40,000	50	5
B8	3	100	45,030	50,000	10	5
B9	3	1,000	4,609	50,000	100	5
B10	3	5,000	4,896	50,000	100	5
X1	12	100	2,529	5,000	10	5
X2	12	100	2,484	5,000	10	5
X3	6	100	14,928	20,000	10	5
X4	6	500	1,190	20,000	50	5
X5	6	100	34,872	40,000	10	5
X6	6	200	14,504	40,000	50	5
X7	6	4,000	15,273	40,000	50	5
X8	3	100	44,950	50,000	10	5
X9	3	1,000	4,871	50,000	100	5
X10	3	5,000	4,615	50,000	100	5

Table 1 shows the key attributes of datasets A, B, and X. The GMRP Problem considers:

i. Processes |p|, which identifies processes needing reassignment based on resource requirements.
ii. Services |s| represents refined process dependency relationships, forming problem constraints.
iii. Machines |m|, encompassing reassignable processes' supporting machines, each with specific resource capacities.
iv. Location |l|, defining disjoint machine sets; it specifies minimum locations (Spreadmin) for each service's process reassignment.

v. Neighbourhood |n| involves machines with reliable service relationships and assigns processes to machines within the same neighbourhood. All neighbourhoods are distinct sets.
vi. Resource |r|: define the set of demandable resources

3. Problem Formulation

3.1 Problem Notation

The notation used in the GMRP problem is summarized as

i. $M -$ set of machines.
ii. $P -$ set of processes.
iii. $R -$ set of resources.
iv. $B -$ balance of triples.
v. $M(p) -$ machine process P is assigned to.
vi. $M_0(p) -$ initial machine process P is assigned to.
vii. $C(m,r) -$ the capacity of resource $r \in R$ for machine $m \in M$ and $\mathcal{R}(p,r)$.
viii. $SC(m,r) -$ the safety capacity of a resource $r \in R$ on a machine $m \in M$.
ix. $R(p,r) -$ the requirement of resource $r \in R$ for process $p \in P$.
x. $U(m,r) -$ the usage U for a machine m for a resource r.

3.2 Decision Variable for GMRP

Let M be the set of machines, and P the set of processes. A solution is an assignment of each process $p \in P$ to one and only one machine $m \in M$ with the assignment mapped as $M(p) = m$. where the initial assignment of process p is denoted as $M_0(p)$. For instance, if $M = \{m_1, m_2\}$ and $P = \{p_1, p_2, p_3\}$, then $M(p_1) = m_1$, $M(p_2) = m_1$, $M(p_3) = m_2$ means processes p_1 and p_2 run on machine m_1 and process p_3 runs on machine m_2. The problem's goal is to minimize costs linked to reassignment, considering two types of constraints: hard and soft. Hard constraints must never be broken; a viable solution respects these, allocating all processes to machines. The problem has stringent limits, and while violating soft constraints is acceptable, their satisfaction is valuable. Achieving optimal satisfaction for soft constraints improves solution quality. Thus, the problem aims to minimize soft constraint violations while meeting all hard constraints.

3.3 Hard Constraints

Capacity constraint: refers to the limitation on the total amount of resources that an instance process can demand. It is essential to ensure that the sum of the demandable resources does not exceed the capacity of the machine resources. Assume R to be the set of resources that is common to all the machines, $C(m,r)$ the capacity of resource $r \in R$ for machine $m \in M$ and $\mathcal{R}(p,r)$ the requirement of resource $r \in R$ for process $p \in P$. Thus, for a given assignment M, the usage U of a machine m for a resource r can be defined as:

$$U(m,r) = \sum_{p \in P} \mathcal{R}(p,r) \quad (1)$$

where $M(p) = m$

Conflict constraints: states that processes belonging to the same service must not be assigned to the same machine. This constraint ensures the system's resilience in the face of potential machine failures.

Thus, processes are normally partitioned into services. Suppose S represents the set of services. A service $s^a \in S$ is a set of processes that must run on distinct machines (note, all services are disjoint). Thus, this constraint is defined by equation 3.4:

$$\forall\, s^a \in S, (p_{i,}\ p_j) \in s^a, p_{i,} \neq p_j \Rightarrow M(P_i) \neq\ M(P_j) \quad (2)$$

$$p^a \text{ and } p^b \text{ are instances of processs}$$

$$s^a \text{ and } s^b \text{ are instances of a services}$$

Spread constraints: state that processes of the same service must be distributed to machines in certain dispersed locations. Suppose L represents the set of locations, and a location $l \in L$ is a set of machines. For each $s^a \in S$, let $spreadMin(s^a) \in N$ be the minimum number of distinct locations where at least one process of service s should run. Thus, this constraint is defined by equation 3.5:

$$\forall\, s^a \in S, \sum_{l \in L} \min(1, |\{p \in s^a | M(p) \in l\}|) \geq spreadMin(s^a) \quad (3)$$

Dependency constraints: state that if two services are interdependent, the collection of machines is divided into neighbourhoods so that the dependent process is assigned to a machine close to the independent process, and vice versa. Suppose N represents the set of neighbourhoods, where a neighbourhood $n^a \in N$ is a set of machines. Assume service s^adepends on service s^b, then each process of s^a should run in the neighbourhood n^a of a s^b process:

$$\forall\, p^a \in s^a, \exists\, p^b \in s^b and\ n^a \in N :\ M(p^a) \in n^a\ and\ M(p^b) \in\ n^a \quad (4)$$

$$n^a \text{ is an instance of a neighbourhood}$$

Transient usage constraints: Assuming a process is migrated from one computer to another, transient usage necessitates the need for a sufficient amount of capacity on both machines (the ability to retain resources during the migration of processes), raising the issue of transient consumption limits. Essentially, when transferring a process p from a machine m_1 to machine m_0, certain resources are duplicated in consumption. One such example is the utilization of disc space, where during the copying process from the machine m_1 to m_0, disc space becomes unavailable on the machine m_0. It is imperative that machine m_0 possesses sufficient disc space to accommodate the copy operation. Let $TR \subseteq R$ denote the subset of resources that necessitate transient utilization, meaning they require capacity on both the initial assignment $M_0(p)$)and the current assignment $M(p)$. Hence, a transient usage can be mathematically expressed as:

$$\forall m \in M, r \in TR, \sum_{p \in P:\, M_0(p)=m \vee M\ M(p)=m} R(p,r) \leq C(m,r) \quad (5)$$

3.4 Soft Constraints

Since the problem aims to optimize the utilization of some set of machines. Hence, to achieve this, a total objective cost is proposed by combining a load cost, a balance cost, and several move costs. Let $SC(m,r)$ be the safety capacity of a resource $r \in R$ on a machine $m \in M$.

Cost 1 (Load): corresponds to the used resource capacity above the safety capacity. It is a metric that quantifies the safety capacity of a resource on a machine. Thus, the load cost is defined per resource and is mathematically represented as:

$$loadCost(r) = \sum_{m \in M} \max(0, U(m,r) - SC(m,r)) \quad (6)$$

Assume B to be some set of triples defined $R^2 \times \mathbb{N}$.

Cost 2 (Balance): refers to the equilibrium between resource availability. For a given triple

$b = \{r_1, r_1, target\} \in$ B,

$$balanceCost(b) = \sum_{m \in M} \max(0, target \times A(m, r_1) - A(m, r_2)) \quad (7)$$

$$with\ A(m, r) = C(m, r) - U(m, r)$$

Suppose $PMC(p)$ is the cost associated with relocating process p from its initial machine $M_0(p)$.

Cost 3 (Process Move)

$$processMoveCost = \sum_{p \in P: M(p) \neq M_0(p)} PMC(p) \quad (8)$$

Cost 4 (Service Move)

$$serviceMoveCost = max_{s \in S}(|\{p \in s | M(p) \neq M_0(p)\}|) \quad (9)$$

Suppose $MMC(m_{source}, m_{destination})$ is the cost associated with relocating a process p from the machine M_{source} to machine $M_{destination}$

Cost 5 (Machine move)

$$machineMoveCost = \sum_{p \in P} MMC(M_0(p), MCP) \quad (10)$$

Hence, the total objective cost

$$totalCost = \sum_{r \in R} weight_{loadcost}(r).loadCost(r) + \sum_{b \in B} weight_{balanceCost}(b).balanceCost(b) + weight_{processMoveCost}.processMoveCost + weight_{serviceMoveCost}.serviceMoveCost + weight_{machineMoveCost}.machineMoveCost \quad (11)$$

4. Methodology Approaches for GMRP

This section provides an overview of the current advancements in algorithms utilized for solving the GMRP. The categorization of existing algorithms was undertaken based on one or more of the following approaches:

i. **Mathematical Optimization algorithms.** Mathematical techniques, such as Integer Programming and Constraint Programming, operate on the assumption that both the objective function and the constraints exhibit linearity (Tan, Goh, Kendall, and Sabar, 2021). Additionally, these algorithms impose restrictions on one or more of the problem variables, requiring them to take on integer values.

ii. **Meta-heuristic algorithms** are computational techniques that are designed to solve complex optimization problems. These algorithms are inspired by natural phenomena or human behavior and aim to find near-optimal solutions Meta-heuristic algorithms are extensively applicable problem-solving techniques that are specifically devised to identify a satisfactory solution within a computationally feasible timeframe (Sharma and Tripathi, 2022). Examples of the meta-heuristic algorithms are
    a. **Multi-Point Solution-Based Method:** The Multi-Point Solution-Based Method also known as population-based algorithms involves maintaining and manipulating a set of candidate solutions, where each answer represents a point in the search space of the problem.
    b. **Single solution-based Method**: involves the maintenance of a solitary candidate solution, with the utilization of a moving operator to explore the vicinity around the existing solution.

iii. **Hyper-heuristic approaches:** refer to a class of problem-solving methodologies that aim to automate the process of selecting or generating heuristics for solving complex problems (Bozorgi, Yazdani, Golsorkhtabaramiri, and Adabi, 2023). Essentially, the hyper-heuristic

techniques employ a collection of heuristics and a selection mechanism to automate the process of determining which heuristics should be utilized.

iv. **Hybrid approaches**. hybrid methodologies integrate the advantageous aspects of multiple meta-heuristic algorithms within a cohesive framework.

Figure 1 shows the paradigms of the methodological approaches applied by distinct authors for the GMRP problem.

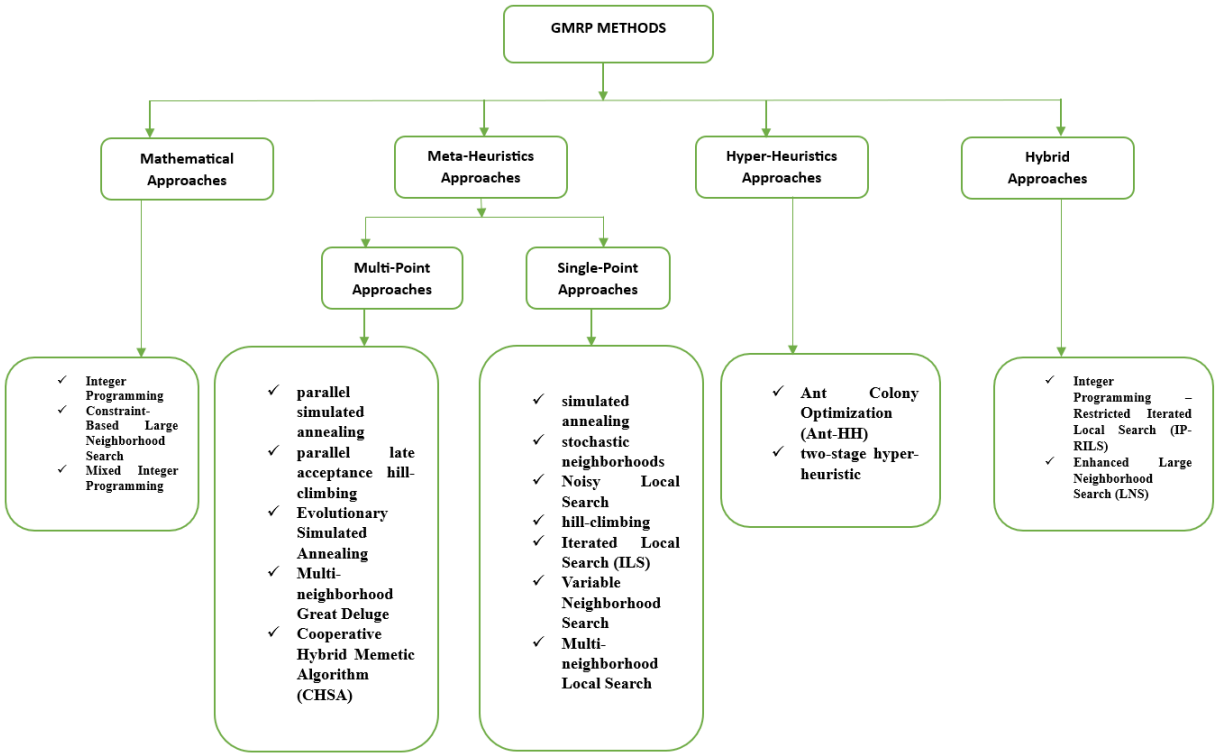

Figure 1: GMRP Methods Paradigm.

4.1 Mathematical Approaches

By iteratively switching processes, Chiraphadhanakul and Figueroa (Team J21) were able to optimize costs and workload distribution across machines using a decomposition strategy (Afsar *et al*., 2016). To identify lower prices, they begin by solving a linear relaxation of the original, unconstrained problem. They use these lower costs to calculate estimated costs for assigning processes to individual machines. Tierney, Delgado, Pacino, and Malitsky (Team J5) proposed an enhanced parallel method that combines a mixed integer model with a destruction/reconstruction approach. This technique involves an exhaustive neighbourhood search where the existing solution undergoes iterations of relaxed randomly selected variables. The local search uses Mixed Integer Programming (MIP) to optimize these relaxed variables, and the process continues until a predefined termination condition is met.

Brandt, Speck, and Völker (2016) as team J25 introduced a Constraint-Based Large Neighborhood Search (CBLNS) approach for the GMRP. This method employs a Large Neighborhood Search (LNS) iteratively, optimizing subsets of processes through Constraint Programming using depth-first search. Strategies include random and process neighbourhood reassignment, targeting underutilized machines, and undo-move reversals. The approach's competitiveness was assessed against top solutions in the ROADEF/EURO competition, demonstrating comparable performance for cases in sets A, B, and X. Hence in seven out of ten instances in the X dataset, the difference from the BKS is

less than 1.6%. However, in certain cases such as X3 and X5, their values deviate significantly from the well-recognized optimal answer of 1883.36 and 324.66% respectively.

Mehdi *et al*. (2015) team 38 proposed an optimization-based heuristic approach to address the GMRP problem. They decomposed it into smaller instances, which were solved via mixed integer programming. Four improvement procedures were introduced: inter-neighbourhood procedure, reassigning processes from different neighbourhood-loaded machines; intra-neighbourhood procedure, reassigning from closely located neighbourhood-loaded machines; single process transfer across machines; and intra-service procedure, reassigning processes covered by a service in the current solution using linear assignment. These methods aimed to enhance solutions and address the GMRP challenge.

Mrad, Gharbi, and Haouari (2016) as team 25 introduced OBH-S27, an optimization-based heuristic technique. This approach addresses load imbalance issues in specific devices through division into smaller problems, solved iteratively with a Mixed Integer Programming solver. A sub-machine generates sub-problems, considering devices with varying load factors. Four assignment augmentation methods were employed: Inter-neighborhood, Intra-neighborhood, Swap, and Intra-service procedures. C-PLEX 12.4 software was used for solving. Incorporating relaxation of the target function focused on the load cost component. Experimental trials on sets A, B, and X showed OBH-S27 effectively optimized solutions in set A. Hence, for the author's study, the maximum difference detected between the Best-Known Solution (BKS) and the solutions derived from seven instances of the X dataset is determined to be below 3.06%.

Team S11, consisting of Clautiaux, Liefooghe, Legillon, and Talbi in 2012, proposed a two-phase approach for workload optimization. The first phase employs an iterated local search algorithm, followed by a heuristic strategy that redistributes overload costs among computers. They use an integer programming solver to improve the existing solution locally.

Jaskwosi, Szubert, and Gawron (2015), as team J12 introduced a hybrid algorithm named HC-LNSHC, merging the Hill Climbing (HC) and Large Neighborhood Search (LNS) techniques. Comprising First Improvement Hill Climbing (FI-HC) and LNS with Mixed Integer Programming, their method aimed to optimize process allocation in instances of the GMRP. FI-HC enhanced solutions via a Shift operator, reallocating processes before local optima, prioritizing cost reduction. Integrated Tabu List considered machines for improved solutions, while LNS generated sub-problems focusing on subsets of machines. Results indicated FI-HC's effectiveness with larger instances and emphasized simultaneous process reassignment's importance. Notably, HC-LNSHC was ranked third most effective in addressing the problem, successfully converging to a single optimal solution in X instances.

Teypaz (Team S14) proposed the integration of a precise algorithm and a metaheuristic approach for the 2012 GRMP problem. The applied algorithm is a Tabu search method with two phases, utilizing matching moves. The well-known Edmonds' bloom shrinking method was used to solve a maximum weight matching problem and assess solution effectiveness. The Tabu search involves process reassignment and swapping, using the LEMON library to find a single Best-Known Solution (BKS) within dataset X.

Pécot (Team S34) proposes an additional tabu search algorithm featuring two distinct neighbourhoods, namely insert and switch. Instead of utilizing entire neighbourhoods, only randomly selected subsets are employed. The aforementioned straightforward approach demonstrates high efficacy by successfully identifying two BKS within the X instances and three BKS within the B instances (Portal, Ritt, Borba, and Buriol, 2016).

Table 1 presents the strengths and weaknesses of the mathematical approach utilized by several authors for the GMRP algorithm.

Table 1: Mathematical approach strengths and weakness.

Authors	**Strength**	**Weakness**
Chiraphadhanakul and Figueroa (Team J21)	The decomposition strategy optimizes costs and workload distribution iteratively.	The reliance on a linear relaxation technique may lead to suboptimal solutions.
Tierney *et al.,* (Team J5)	The enhanced parallel method combines a mixed-integer model with a destruction/reconstruction approach for improved optimization.	The exhaustive neighbourhood search may be computationally expensive.
Brandt *et al.,* (Team J25)	The Constraint-Based Large Neighborhood Search (CBLNS) approach employs a depth-first search for optimization.	Deviations in performance, especially in cases like X3 and X5, indicate variability in solution quality.
Mehdi *et al*., (Team 38)	Optimization-based heuristic approach with decomposition into smaller instances.	The effectiveness of improvement procedures may vary across different problem instance.
Mrad *et al.,* (Team 25)	OBH-S27 addresses load imbalance through iterative division and Mixed Integer Programming.	The effectiveness is case-dependent, with differences in solutions for various datasets.
Clautiaux *et al.,* (Team S11)	Two-phase approach with an iterated local search and heuristic strategy	The reliance on an integer programming solver may limit scalability.
Jaskwosi *et al.,* (Team J12)	HC-LNSHC combines Hill Climbing and Large Neighborhood Search Techniques	The chosen parameters influence the effectiveness, and it may not guarantee a global optimal solution
Teypaz (Team S14)	The combination of enhanced Tabu search and Edmonds' bloom shrinking method may lead to faster convergence to optimal or near-optimal solutions. This is especially beneficial in scenarios where computational resources are limited and quick decision-making is required.	The effectiveness depends on the quality of the chosen matching moves.
Pécot (Team S34)	Additional Tabu search algorithm with distinct insert and switch neighbourhoods	The effectiveness of Tabu Search is highly dependent on properly tuning its parameters, such as the tabu list size, aspiration criteria, and neighbourhood structures. Selecting inappropriate parameter values can lead to suboptimal or inefficient search processes.

4.2 Single-Point Solution-Based Approaches

Gabriel, Marcus, Luciana, Leonardo, and Alexandre (2012) proposed a Simulated Annealing approach to tackle the GMRP Problem. Their method incorporates two neighbourhoods, each utilizing efficient data structures to expedite core operations. The first neighbourhood relocates tasks between machines, while the second swaps tasks on different machines. To generate viable solutions, the authors introduced a probability function with p and $1 - p$ probabilities (with p set at 0.7) to choose between neighbourhoods. Their strategy freezes the solution when no improvement occurs for 20 iterations and accepted moves are below 0.1%. In such cases, the temperature is increased by dividing the initial temperature by 100, aiming to escape local minima by perturbing the current solution.

The Simulated Annealing (SA) approach was also adapted by Ritt, Buriol, Borba, Portal, and Benavides (Team S23) in 2012 and was enhanced to integrate two stochastic neighbourhoods (random process reassignment and swap). The authors proposed a particular data format that facilitates the assessment of moves and implementation of updates in constant time. In six occurrences within the X dataset, the disparity from the BKS (benchmark standard) is seen to be below 0.2%. Additionally, it is important to highlight that the maximum deviation from the Best-Known Solution (BKS) is less than 50%, with a particular case (X_5) demonstrating a deviation of 48.7%.

Larose and Posta (Team J10) developed a parallelized iteration of the LS algorithm, running simultaneously on two threads of the test machine. This method involves generating feasible solutions and refining them via path-relinking, which includes selecting initial and guided solutions from a pool. The algorithm generates a neighbourhood by transferring processes between machines (Afsar *et al.*, 2016).

Gabay and Zaourar (2016) as team J19 introduced an innovative approach involving vector bin packing (VBP) with heterogeneous bins to address the GMRP challenge. They extended the VBP problem to accommodate bins of varying sizes, a framework suitable for addressing virtual machine placement issues that extend beyond GMRP. The researchers proposed diverse families of effective greedy heuristics, demonstrating their feasibility and adaptability to varying constraints. By analyzing the structural facets of GMRP, they partitioned the problem into subproblems and customized their algorithms accordingly. The evaluation involved academic benchmarks, including the GMRP problem and a heterogeneous bin variant of a randomly generated VBP problem.

Gavranovic and Buljubasic (2016) team S41, introduced the Noisy Local Search (NLS) algorithm, employing a multi-start local search strategy with Swap, Shift, and BPR operators. Processes were arranged by resource requirements before applying NLS, prioritizing larger ones and adaptively resizing a subset. The BPR operator enhanced solution quality in specific cases but increased execution time (3-7 times). It satisfied constraints except for capacity limits, triggering perturbations for improved solutions. A noise method and adjusted objective functions countered local optima. NLS addressed random seed concerns by reallocating procedures based on various seeds, yielding notable results across instance sets A and X, albeit with no solutions for set B. Gavranovic and Buljubasic's (2016) numerical findings surpassed others, prevailing as the superior outcome.

Portal *et al.* (2016) addressed the GMRP problem by presenting a heuristic algorithm based on simulated annealing. Their approach employs two neighbourhoods: one for relocating a process between machines and another for swapping processes between machines. They utilize an objective function, with a complexity of $O(r + d)$ where r represents the resources and d indicates the dependencies of the service a process belongs to. The algorithm operates concurrently with two distinct parameter sets until a convergence criterion is met.

Team J6, comprising Dudebout, Masson, Michallet, Petrucci, Subramanian, and Vidal in 2012, presented an innovative strategy blending wide neighbourhood and local search techniques (Afsar *et al.*, 2016). The local search involves two key operations, relocating and swapping processes on

computers. The broader neighbourhood search entails iteratively dismantling and reconstructing substantial solution segments, which is achieved by executing a mixed integer program.

Vancroonenburg and Wauters (Team J17) proposed a metaheuristic approach using the late acceptance hill-climbing algorithm with two efficient neighbourhood functions (Afsar *et al.*, 2016). The approach evaluates new solutions based on their quality relative to the current solution within the last L iterations. Strategies used by other teams included process reassignment and switch machine neighbourhood. Higher L values result in slower convergence but better final solutions. Maximum deviation from the benchmark solution (BKS) was about 1.5% in six instances during the analysis of set X, noting that some instances, like X5, exhibit larger variations.

Multi-Start Iterated Local Search (MS-ILS), introduced by Masson *et al.* (2013), is a local search-based approach addressing Multi-Capacity Bin Packing (MCBPP) and GMRP. It utilizes Iterated Local Search (ILS) incorporating Swap and Shift operators. The method begins by optimizing Load and Balance costs in a machine subset m^{-}; then, individual operators are applied to the chosen machine m^{1}, accepting the one improving solution quality. The local search was applied to either the best-found solution or the current one iteratively to balance exploitation and exploration. MS-ILS combats local optima using a multi-start component. The study employs shaking motions, home relocation, and K-swap operators. Home relocation restores k processes to their initial machine, while K-swap swaps groups of 3, 4, or 5 processes between k pairs of machines. MS-ILS achieved one of the best-known solutions for set A.

Team S5 (Gogos, Valouxis, Alefragis, and Housos) introduces an innovative approach integrating integer programming with a late acceptance metaheuristic, utilizing a large variable neighbourhood search. The cooperative process continues until the set time limit is reached, achieving a maximum separation of less than 234.94% from the BKS even if no BKS is found. Team S27 (De Oliveira, Lopes, de Noronha, de Morais, and de Souza) integrates integer programming and a metaheuristic iterated local search with a perturbation technique. Their approach yields an average improvement of 63.97% from the initial solution. Team S37 (Lu and Whang) refines the iterated local search algorithm by exploring three neighbourhoods: process reassignment, 1-1 process swap, and 1-2 process swap. The stochastic process introduces perturbations when no solution change occurs. They achieve differences from the benchmark (BKS) of less than 4.83% for seven instances of set X. Team J38 (Catusse) presents a hill climbing heuristic employing "Move," "Swap," and "2-1 Swap" operations across distinct neighbourhoods. Novel solutions are accepted for improvement over the current solution, sometimes favouring suboptimal solutions for faster convergence. Team S6, consisting of Benoist, Gardi, Estellon, Darlay, Megel, and Nouioua, tackled the GMRP by transforming it into a binary model. They employ a local solver utilizing adaptive simulated annealing, for default search. This involves adjusting specific binary variables to enhance the objective function while adhering to constraints. Changes that breach constraints are rejected and alternative actions are assessed based on their impact on the objective function. In their study, Jarbou and Mladenovic (Team S40) proposed a Variable Neighborhood Search (VNS) algorithm that operates at three levels. This algorithm effectively decomposes the problem into smaller subproblems. The primary framework, known as skewed variable neighbourhood search, can incorporate marginally inferior alternatives, provided that they exhibit sufficient dissimilarity from the current solution. This approach identifies two best-known solutions (BKS) on the B instances.

Wang, Lü, and Ye (2016) proposed a Multi-neighborhood Local Search (MNLS) approach for the GMRP. Their algorithm incorporates three primary neighbourhood structures, an auxiliary neighbourhood, a partition mechanism for neighbourhoods, and a dynamic perturbation operator. This approach enables exploration beyond feasibility, accelerates search through partitioning, and avoids local optima. The algorithm's holistic strategy aims to enhance solution quality by diversifying search and escaping local optima.

Table 2 presents the strengths and weaknesses of the single-point methods applied by distinct authors for the GMRP algorithm.

Table 2: Single-point solution-based approaches strengths and weakness.

Author	**Strength**	**Weakness**
Gabriel *et al.,*	Utilizes Simulated Annealing with two efficient neighborhoods, to enhance solution exploration. Also incorporates a probability function for selecting neighbourhoods, adding adaptability. Freezes the solution to escape local minima, contributing to convergence.	Relies on a fixed probability value, limiting adaptability. Temperature adjustment strategy might impact convergence speed.
Ritt *et al.,* (Team S23)	Integrates two stochastic neighbourhoods, improving solution diversity. Proposes a data format facilitating efficient assessment and updates. Achieves low disparity from the benchmark standard in six instances.	Reports a deviation of 48.7% in one case, indicating variability. Specifics on the stochastic nature of neighbourhoods are not detailed.
Larose and Posta (Team J10)	Implements a parallelized version of the LS algorithm for efficiency. Involves path-relinking to refine solutions, enhancing optimization. Utilizes a pool for selecting initial and guided solutions, promoting diversity.	Limited details on the specifics of path-relinking. Efficiency gains from parallelization are not quantified.
Gabay and Zaourar (Team J19)	Introduces vector bin packing with heterogeneous bins, expanding applicability. Proposes diverse greedy heuristics tailored to varying constraints. Customizes algorithms based on the structural facets of GMRP, enhancing effectiveness.	The variability applied can result in not being near-optimal.
Gavranovic and Buljubasic (Team S41)	Introduces Noisy Local Search with diverse operators, enhancing solution quality. Addresses random seed concerns, contributing to result stability. Numerical findings surpass other algorithms, establishing superiority.	Increased execution time with the Big Process Re-assignment operator may impact efficiency.
Portal *et al.* (2016)	Utilizes simulated annealing with two effective	The effectiveness of Search is highly dependent on the proper

	neighbourhoods. Incorporates an objective function with manageable complexity. Operates with two parameter sets concurrently, enhancing flexibility.	tuning of its heating parameters.
Dudebout *et al.* (Team J6)	Integrates wide neighbourhood and local search techniques for comprehensive exploration. Applies mixed integer programming in local search, improving solution accuracy. Involves dismantling and reconstructing solution segments, promoting diversity.	The cost of applying mixed integer programming in local search can increase the computational overhead.
Vancroonenburg and Wauters (Team J17)	Utilizes late acceptance hill-climbing with efficient neighbourhood functions. Adapts to solution quality within the last L iterations, enhancing convergence. Achieves a maximum deviation of about 1.5% from the benchmark solution in some instances.	Higher L values might result in slower convergence, impacting efficiency.
Gogos *et al.*, (Team S5)	Integrates integer programming with late acceptance meta-heuristic, combining precision and flexibility. Achieves notable results even without finding the best-known solution. Utilizes large variable neighbourhood search for comprehensive exploration.	-
De Oliveira *et al.*, (Team S27)	Integrates integer programming and iterated local search with a perturbation technique. Yields an average improvement of 63.97% from the initial solution, indicating effectiveness. Combines precision from integer programming with adaptability from iterated local search.	If not carefully implemented, the perturbation technique can lead to increased execution time.

Lu and Whang (Team S37)	Achieves differences from the benchmark (BKS) of less than 4.83% for seven instances in set X.	
Jarbou and Mladenovic (Team S40)	Proposes a Variable Neighborhood Search (VNS) algorithm operating at three levels. Decomposes the problem into smaller subproblems, promoting efficiency. Identifies two best-known solutions (BKS) on the B instances, demonstrating effectiveness.	Variability can result in increased computational overhead.
Wang *et al.* (2016)	Introduces a Multi-neighborhood Local Search (MNLS) approach for GMRP. Implements a dynamic perturbation operator to counter local optima.	The dynamic perturbation operators might also result in increased computational cost.

4.3 Multi-Point Solution Based Method

Sansoterra, Ferruci, Calcaveccia, and Sironi (Team J33) proposed an approach that integrates parallel simulated annealing with variable neighbourhood search methods for the GMRP intending to exploit the computing capabilities of several CPU cores (Afsar *et al.*, 2016). Information is transmitted across heuristics through a shared repository of possible solutions. The solution repository provides a secure method for accessing two separate collections of solutions, each distinguished by either high quality or great diversity. The path relinking technique probabilistically selects one solution from each set with a 50% chance and examines the intermediate solutions. One solution that is widely recognized as the best-known solution (BKS) is identified from dataset X, whereas another BKS is identified from dataset B.

To solve the GMRP, (Turky, Sabar, Sattar, and Song, 2016) developed a parallel late acceptance hill-climbing algorithm (P-LAHC) based on evolutionary principles. Their research was driven by a desire to improve efficiency by breaking free of local optima. Instead of relying on a single solution, P-LAHC uses a set of possible solutions. Each possible answer represents one LAHC solution. The LAHC processes all run simultaneously to boost the quality of the search results. The LAHC models avoid becoming trapped at local optimums by starting with different beginning individuals (as the LAHC process) and taking alternative search pathways. Ayad *et al.* (2016) also incorporate mutation operators to eliminate search stagnation. This technique has considerably mitigated the likelihood of the search being stuck at a local optimum.

Turky, Sabar, Sattar, and Song (2017) introduced an Evolutionary Simulated Annealing (ESA) algorithm. ESA employs a population of potential solutions, each equipped with its own Simulated Annealing (SA) algorithm instance. These SA procedures run concurrently, starting from unique initial solutions generated by modifying S_0. The objective was to generate distinct solutions with various local optima levels systematically. The authors used mutation operators (Swap, Double Swap, Shift, and Double Shift) to overcome local optima. The study found that ESA's effectiveness was comparable to that of advanced algorithms, particularly noting its highest achievement of knowledge in set A. In 2017, Turky *et al.* (2017) introduced a multi-neighborhood great divide (MNGD). The Great Deluge algorithm is a singular-point method that incorporates acceptance criteria for both enhanced and non-enhanced solutions, considering a predetermined threshold (referred to as the

quality of s_0) and a minor decrement ($\in$) applied during each iteration. Four operators are employed to introduce disturbances to solutions: single swap, double swap, single shift, and double shift operators. The results obtained from the three benchmark instances indicate that MNGD achieved the best-known solution in sets A and B, with one solution discovered in each set.

Turky, Sabar, Sattar, and Song (2018) introduced the Cooperative Hybrid Memetic Algorithm (CHSA), an advanced version of the parallel late acceptance hill-climbing algorithm (P-LAHC), integrated with SA. The algorithm concurrently executes numerous SA algorithms with different configurations involving perturbation operators, beginning temperature, and cooling coefficient. Each iteration's best solution from each SA algorithm is preserved after being evaluated using operators like single swap, double swap, single shift, and double shift. The CHSA algorithm outperforms competing approaches in sets A and B, using a novel mutation operator to improve the optimal solution obtained from all SA approaches. This execution was parallelized using four threads.

Saber, Gandibleux, Neil, Murphy, and Ventresque, (2019) devised GeNePi, a hybrid optimization model, integrating GRASP, Non-dominated Sorting Genetic Algorithm, and Pareto Local Search Algorithm to address the Google Machine Reassignment Problem (GMRP). They applied this model to 14 dataset instances from categories A and B. In the initial phase, they employed a Greedy Randomized Adaptive Search Procedure to reassign processes with relaxed randomization, based on dependencies sequentially. The non-dominated sorting genetic algorithm used crossover and mutation techniques to refine individual assignments and mix their features. Lastly, the Pareto Local Search applied local search operators like process swaps, 1-exchange, and shifts for improved solutions. The study primarily analyzed the time GeNePi required to attain feasible solutions across instances.

Table 3 presents the strengths and weaknesses of the multi-point solution-based method utilized by several authors for the GMRP algorithm.

Table 3: Multi-point solution-based method strengths and weaknesses.

Authors	**Strength**	**Weakness**
Sansoterra *et al.,* (Team J33)	Integration of parallel simulated annealing with variable neighbourhood search for GMRP. Exploitation of multi-core CPU capabilities for enhanced computing efficiency. Utilization of a shared repository for information exchange among heuristics. Management of two collections of solutions based on high quality and diversity. Implementation of path relinking technique for solution selection and refinement.	Reliance on probabilistic selection for path relinking may introduce variability.
Turky *et al.,* (2016 - P-LAHC)	Introduction of parallel late acceptance hill-climbing algorithm (P-LAHC). Utilization of evolutionary principles to escape local optima. Simultaneous execution of	Inappropriate tuning or selection of mutation operators could potentially hinder convergence or compromise the quality of solutions.

	multiple LAHC processes to enhance search quality. Mitigation of local optima by starting with different initial individuals. Incorporation of mutation operators to prevent search stagnation.	
Turky *et al.* (2017)	Introduction of Evolutionary Simulated Annealing (ESA) algorithm. Utilization of a population of potential solutions with concurrent SA instances. Use of mutation operators to overcome local optima in ESA. Introduction of Multi-neighborhood Great Deluge (MNGD) algorithm. Achievement of best-known solutions in sets A and B with MNGD.	By introducing a population of potential solutions with concurrent Simulated Annealing (SA) instances, the ESA algorithm may lead to increased algorithmic complexity. Managing and coordinating multiple instances simultaneously can demand substantial computational resources.
Turky *et al.* (2018)	Introduction of Cooperative Hybrid Memetic Algorithm (CHSA). Integration of SA with multiple configurations in a parallel execution. Use of perturbation operators and cooling coefficients for SA algorithms. Preservation of the best solution from each SA algorithm in each iteration. Outperformance of competing approaches in sets A and B.	Managing diverse perturbation operators and cooling coefficients may result in additional computational overhead.
Saber *et al.*, (2019)	Development of a hybrid optimization model (GeNePi) using GRASP, NSGA, and Pareto Local Search. Application to 14 dataset instances from categories A and B. Sequential reassignment of processes using Greedy Randomized Adaptive Search Procedure. Utilization of crossover and mutation techniques in the Non-dominated Sorting Genetic Algorithm. Application of local search operators in Pareto Local Search for improved solutions.	The model's success relies on appropriately tuning parameters for each integrated algorithm. Sensitivity to parameter changes may pose challenges in achieving optimal performance across different problem instances.

4.4 Hyper-Heuristic Approaches

Turky (2019) proposed an enhanced version of the bi-level hyperheuristic approach that was developed using Ant Colony Optimization (Ant-HH). This approach refines the concept of HH by incorporating various enhancements. Ant-HH has two levels: the first is dedicated to local search strategies, and the second is to operators. Using the Ant-HH strategy, an Ant Colony Optimization algorithm controls both by utilizing a combination of appropriate local search algorithms and sequences of operators. When searching for a solution, ants use a two-tiered, weighted graph where nodes reflect local search techniques and operators. Connectivity between nodes is valued as shown by these weights. Considering quality and diversity as goals, pheromones are deposited along edges, leading to non-dominated solutions. When it comes to improving algorithm convergence, heuristic knowledge places a premium on sets of two related local search methods or operators. The author's algorithm uses a pool of varied, high-quality answers. Each ant chooses a solution from the pool of possibilities as the process repeats. The final result uses Pheromone and heuristic knowledge to fine-tune local search methods and operator sets. It uses the same distance-based method as HH to update its population. Ant-HH stands out as one of the top algorithms in testing sets A, B, and X. Nine of the most popular solutions in Set A, six in Set B, and eight in Set X were found.

Turky, Sabar, Dunstall, and Song (2020) proposed the utilization of a hyper-heuristic (HH) technique to tackle combinatorial optimization challenges, explicitly emphasizing the GMRP. They used a two-stage hyper-heuristic framework in their research, which required the customization of local search algorithms and internal operators. Choosing a local search strategy like SA, ILS, LAHC, Great Deluge, or Steepest Descent is the first step in the process (referred to as HH-LS). Several local operators, such as the single Swap, double Swap, single Shift, double Shift, Swap-Shift, Shift-Swap, BPR, and Swap-BPR, are used in the subsequent phase. As a result, the innovative adaptive ranking system uses entropy to select suitable neighbourhood structures, thereby continuously updating the pool of available structures. The study includes a statistically sizable subset of the population, which improves the reliability of the results and allows researchers to more accurately gauge the problem's size and the solutions' scope. Updating the population of solutions with a distance-based method ensures differential representation across different regions of the search space, which improves allocation diversification.

Table 4 depicts the strengths and weaknesses of the hyper-heuristic approaches utilized by several authors for the GMRP algorithm.

Table 4: Hyper-heuristic approaches strengths and weakness.

Authors	**Strength**	**Weakness**
Turky (2019)	The approach utilizes Ant Colony Optimization, a metaheuristic algorithm, to control local search strategies and operators. This combination allows for a more comprehensive exploration of the solution space.	The described approaches, especially Ant-HH, might be complex due to the integration of components like pheromone deposition, heuristic knowledge, and two-tiered graphs. This complexity could potentially impact the algorithm's ease of implementation and understanding
Turky *et al.,* (2020)	Introduction of parallel late acceptance hill-climbing algorithm (P-LAHC). Utilization of evolutionary principles to escape local optima. Simultaneous execution of	The performance of these algorithms could be sensitive to the tuning of parameters, and finding optimal parameter values might require additional effort.

	multiple LAHC processes to enhance search quality. Mitigation of local optima by starting with different initial individuals. Incorporation of mutation operators to prevent search stagnation.	

4.5 Hybrid Approaches

Lopes *et al.*, (2015) introduced a hybrid methodology that combines Iterated Local Search and Integer Programming (IP) to enhance solution quality in production scheduling. The approach incorporated specialized Shift and Swap operators for load sorting, focusing on machines and processes. The concept aimed to minimize Load Cost by prioritizing higher load cost processes. A "Swap in Service" operator was introduced to the Swap operator, ensuring perturbations adhere to dependency, spread, and conflict criteria. The methodology, known as Integer Programming – Restricted Iterated Local Search (IP-RILS), employed IP and Branch and Bound techniques to address sub-problems. Experimental results on problem instances B and X demonstrated that IP-RILS achieved optimal solutions, yielding zero best-known answers.

Souza, Grimes, and O'Sullivan (2023) proposed an innovative Large Neighborhood Search (LNS) approach enhanced by a domain-specific heuristic for selecting neighbourhoods. This heuristic capitalizes on resource utilization imbalances among machines to identify promising processes strategically. Unlike conventional heuristics, this approach exploits problem-specific characteristics for more efficient solutions. Two distinct search strategies emerged for optimizing sub-problems in the Machine Reassignment Problem. The first strategy adapts Limited Discrepancy Search (LDS) to handle large-scale instances. The second strategy combines constraint programming and random restart strategies, aiming for diverse and effective search trajectories.

4.6 Ranking Overview Method

The ranking process scheduled during the challenge takes into cognizance the reference solution. The reference solution, where no process is reallocated, is trivial. Both the qualification and final stages employed the same ranking scheme. Team scores were calculated as the difference between their and best costs, divided by the reference solution's cost (Afsar, Artigues, Bourreau, and Kedad-Sidhoum, 2016). The qualification phase aggregated scores across dataset A instances. The final phase's cumulative score came from summating scores over B and X datasets. Qualification required a 100% score. Figure 1 graphically represents scores and qualifying teams, denoted by "S" for Senior and "J" for Junior teams. Junior teams consist exclusively of non-PhD students. No additional Senior teams registered. Table 3 details qualified team composition and outcomes during qualification, providing further insight.

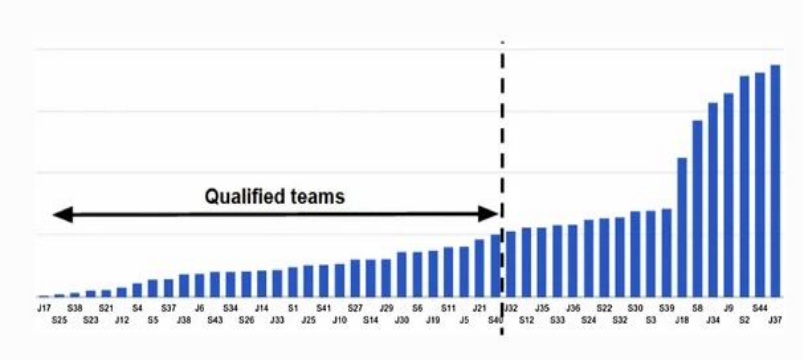

Figure 2: Candidate Qualification Stage Scores.

In the final phase, the eligible participants were required to handle instances of significantly greater magnitude. Out of the teams that met the necessary qualifications, a total of 27 teams submitted their program for evaluation.

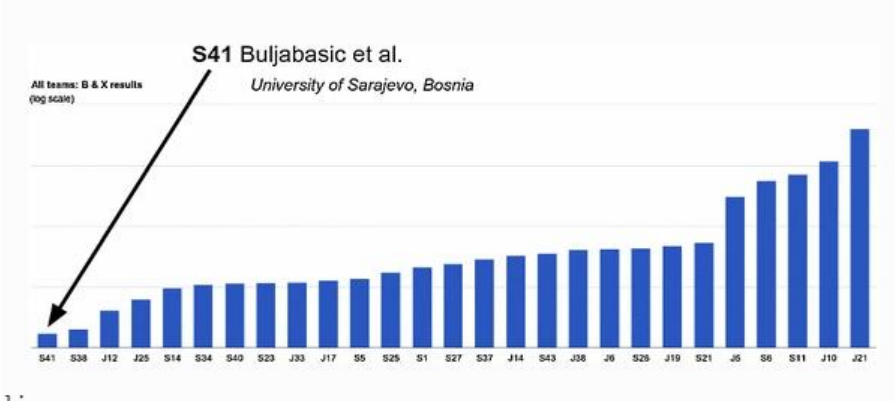

Figure 3: Final phase candidate scores.

Figure 4 is a graph depicting the breakdown of the years in which the various examples of literature were published. Publications per year are plotted along the y-axis, while years make up the x-axis. The graph depicts the fluctuating contribution rate over the years. The number of donations received in a given year is represented by the height of the bar for that year.

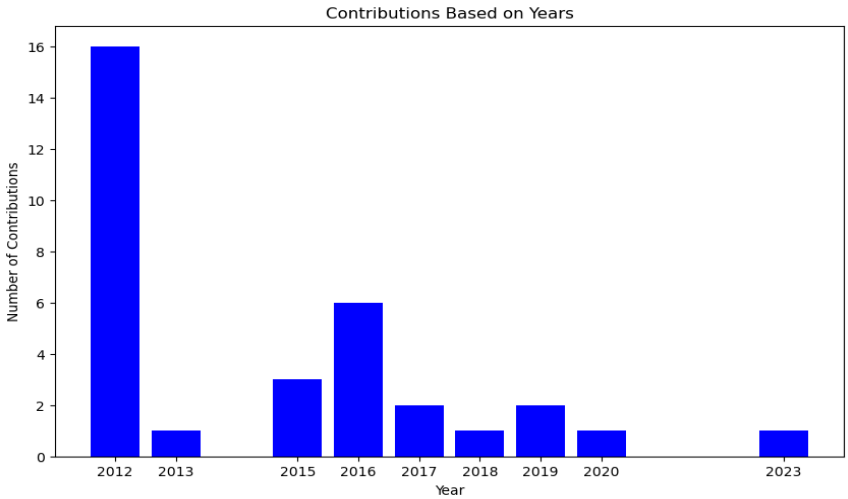

Figure 4: Author contributions on GMRP over the years.

4.7 Summary of Best Solutions

Table 5: Algorithm mapping for the GMRP approaches.

Best-Method	**Mapping**
ANT-HH	M_1
HH	M_2
HC-LNSHC	M_3
VNS	M_4
OBH	M_5
NLS	M_6
CHSA	M_7
MS-ILS	M_8
LNS-CP	M_9
LNS-MIP	M_{10}
MNLS	M_{11}
MNGD	M_{12}
EM-LNSHC	M_{13}
SD-Combined	M_{14}

Table 5 presents a mapping for each best-known algorithm for the GMRP problem. The mappings are denoted by M_{i-N}. Correspondently, Table 6 considers the best-known algorithm mapped from Table 1 as the reason for presenting the most prominent solutions during and after the competition from the literature reviewed.

Table 6: Summary of literature for GMRP.

Instance	**Best Known**	**Mapped Algorithm**
A1-1	4.4306501×10^7	$M_2, M_1, M_4, M_9, M_{10}, M_8, M_{14}, M_{11}, M_5$
A1-2	7.77532177×10^8	M_2, M_1
A1-3	5.83005715×10^8	M_1
A1-4	2.47875200×10^8	M_2, M_1
A1-5	7.27578308×10^8	M_1, M_5
A2-1	161	M_1
A2-2	7.20671511×10^8	M_1
A2-3	1.182260491×10^9	M_6
A2-4	1.680368560×10^9	M_1
A2-5	3.07150821×10^8	M_1
B-1	3.291069365×10^9	M_1
B-2	1.010949451×10^9	M_{13}, M_{12}

B-3	1.56519816×10^8	M_4
B-4	4.677792536×10^9	M_1
B-5	9.22944510×10^8	M_7, M_1
B-6	9.525421332×10^8	M_4
B-7	$1.4834456020 \times 10^{10}$	M_2
B-8	1.214086286×10^9	M_4
B-9	$1.5885437252 \times 10^{10}$	M_1
B-10	$1.8048187105 \times 10^{10}$	M_1
X-1	3.030246091×10^9	$M_4,$
X-2	1.002379317×10^9	M_7, M_1
X-3	7.5154×10^4	M_1
X-4	4.721586142×10^9	M_1
X-5	5.7973×10^4	M_1
X-6	9.546936159×10^9	M_1
X-7	$1.4252476500 \times 10^{10}$	M_1
X-8	3.2014×10^4	M_1
X-9	$1.6125531142 \times 10^{10}$	M_1
X-10	$1.7815045320 \times 10^{10}$	M_4, M_6

The provided results present the outcomes of different algorithms applied to the GMRP for various instances based on the research conducted by this study. The results highlight the best-known solutions achieved by each algorithm for each instance. For instance, A1-1, the best-known solution is 4.4306501×10^7. Multiple algorithms, including HH, $M_2, M_1, M_4, M_9, M_{10}, M_8, M_{14}, M_{11}, and\ M_5$were employed, indicating a variety of approaches used to tackle this instance. Instance A1-2 exhibits a best-known solution of 7.7753217710^8. Here, $M_2\ and\ M_1$algorithms were utilized, with M_1 proving effective in finding competitive solutions. The results for A1-3, A1-4, and A1-5 showcase that M_1consistently performed well in generating solutions for these instances, with best-known solutions of 5.83005715×10^8; 2.47875200×10^8; and 7.27578308×10^8 respectively. Moving on to instance A2-1, the ANT-HH algorithm produced a solution of 161, demonstrating its success in this particular case. In instance A2-2, M_1 achieved the best-known solution of 7.20671511×10^8 indicating its effectiveness in larger instances as well. For instance, in A2-3, NLS achieved a solution of 1.182260491×10^9 suggesting its applicability to more complex cases. Instances A2-4 and A2-5 once again highlight the strength of the M_1 algorithm with solutions of 1.680368560×10^9 and 3.07150821×10^8 respectively. Instances B-1 through B-10 demonstrate varying best-known solutions, with M_1, M_{13}, M_{12}, M_3, and M_7algorithms showing their effectiveness in addressing these instances. Instances X-1 through X-10 further showcase the application of different algorithms, with M_4,M_6,M_7, and M_1 producing competitive results across various instances. Overall, the results illustrate the diversity of algorithms utilized to tackle the GMRP, with some algorithms consistently outperforming others across different instances. This provides insights into the strengths and

weaknesses of different approaches when addressing combinatorial optimization problems like the GMRP.

5. Conclusion

This paper meticulously surveys a spectrum of algorithms employed by numerous researchers over time to address the GMRP. The research landscape unveiled through this survey reflects a rich tapestry of methodologies designed to confront the intricate challenges inherent in cloud machine resource allocation and optimization. Heuristic approaches, notably Simulated Annealing (SA), Local Search (LS), and Evolutionary Algorithms, along with innovative techniques like population-based algorithms, have played pivotal roles in effectively addressing the complexities of GMRP.

Integrating these heuristics with parallel computing and domain-specific insights has yielded notable improvements in convergence rates and solution quality. Population or multi-point algorithms such as the Ant Algorithm, Evolutionary Simulated Annealing, and Hyper Heuristics have demonstrated their effectiveness in enhancing GMRP solutions. A nuanced analysis of contributions over time reveals dynamic trends, showcasing varying research output that reflects the field's responsiveness to emerging challenges.

In summary, the survey underscores the evolving nature of GMRP research, emphasizing the amalgamation of heuristic algorithms, optimization techniques, and domain-specific considerations in tackling the intricacies of resource allocation. As computational capabilities continue to advance, the exploration of GMRP promises to deliver even more refined solutions, thereby enriching optimization and resource management practices. In conclusion, future research endeavours should contemplate the adaptation of population-based algorithms, with a strategic focus on algorithms with fewer parameters to reduce complexity. The Jaya algorithm, for instance, stands out as a viable option for researchers exploring avenues to enhance GMRP solutions.

Reference

Afsar, H., Artigues, C., Bourreau, E., and Kedad-Sidhoum, S. (2016): Machine reassignment problem: the ROADEF/EURO challenge 2012. *Annals of Operational Research*, 242, 1-17.

Alduailij, M., Khan, Q. W., Tahir, M., Sardaraz, M., Alduailij, M., and Malik, F. (2022): Machine-learning-based DDoS attack detection using mutual information and random forest feature importance method. *Symmetry*, 14(6),1095.

Arnaud, T. (2020): The challenge of the hybrid cloud.

http://www.thedigitalsociety.tech/index.php/2020/05/08/the-challenge-of-the-hybrid-cloud/

Ayad, T., Sabar N.R., Sattar A., and Song A. (2016): Parallel Late Acceptance Hill-Climbing Algorithm for the Google Machine Reassignment Problem. https://doi.org/10.1007/978-3-319-50127-7_13.

Ayad, T., Sabar, N.R., Sattar, A., and Song, A. (2017): An evolutionary simulating annealing algorithm for Google machine reassignment problem in Intelligent and Evolutionary Systems. *Springer*, 431-442.

Balasubramaniam, S., Vijesh Joe, C., Sivakumar, T. A., Prasanth, A., Satheesh Kumar, K., Kavitha, V., and Dhanaraj, R. K. (2023): Optimization Enabled Deep Learning-Based DDoS Attack Detection in Cloud Computing. *International Journal of Intelligent Systems, 2023*.

Bozorgi, S. M., Yazdani, S., Golsorkhtabaramiri, M., and Adabi, S. (2023): A hyper-heuristic approach based on an adaptive selection operator and behavioural schema for global optimization. *Soft Computing*, 1-50.

Brandt, F., Speck, J., and Völker, M. (2016): Constraint-based large neighbourhood search for machine reassignment—A solution approach to the ROADEF/EURO challenge 2012. *Ann. Oper. Res.*, 242(1), 63–91.

Gabay, M., and Zaourar, S. (2016): Vector bin packing with heterogeneous bins: application to the machine reassignment problem. *Annals of Operational Research,* 242, 161–194. https://doi.org/10.1007/s10479-015-1973-7.

Gabriel, P., Marcus, R., Luciana, S.B., Leonardo, B., and Alexandre, B. (2012): Simulated Annealing for the Machine Reassignment Problem. https://www.inf.ufrgs.br/~MRPRITT/Publications/R05-euro2012.pdf.

Gavranović, H., and Buljubašić, M. (2016): An efficient local search with a noising strategy for the Google Machine Reassignment problem. *Annals of Operational Research,* 242, 242, 19–31. https://doi.org/10.1007/s10479-014-1686-3.

Guo, X. (2021): Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defence algorithm. *Alexandria Engineering Journal*, 60(60, 5603–5609.

Ja´skowski W., Szubert M., and Gawron, P. (2015): A hybrid MIP-based large neighbourhood search heuristic for solving the machine reassignment problem. *Annals of Operational Research,* 242, 33–62.

Jiang, W., Wang, Y., and Jiang, Y. (2020): Mobile internet mobile agent system dynamic trust model for cloud computing. *Computers, Materials & Continua*, 62(1), 123–136.

Liu, H. (2022): Research on cloud computing adaptive task scheduling based on ant colony algorithm. *Optik*, 258, 168677–168711.

Lopes R., Morais, V.W. C., Noronha, T. F., and Souza, V. A. A. (2015): `Heuristics and metaheuristics for a real-life machine reassignment problem. *International Transactions in Operational Research,* 22(1), 77-95.

Manikandan, N., Gobalakrishnan, N., and Pradeep, K. (2022): Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment. *Computer Communications*, 187, 35–44.

Masson, R., Vidal, T., Michallet, J., Penna, P. H. V., Petrucci, V., Subramanian, A., and Dubedout, H. (2013): 'An iterated local search heuristic for multi-capacity bin packing and machine reassignment problems. *Expert Syst. Appl.*, 40, 13, 5266–5275.

Mehdi, M., Anis, G., Mohammed, H., and Mohamed, K. (2015): An optimization-based heuristic for machine reassignment. https://www.researchgate.net/profile/Mehdi-Mrad/publication/282332585_An_optimization-based_heuristic_for_the_machine_reassignment_problem/links/5af94d81a6fdcc0c03345343/An-optimization-based-heuristic-for-the-machine-reassignment-problem.pdf.

Mrad, M., Gharbi, A., and Haouari, M. (2016): An optimization-based heuristic for the machine reassignment problem. *Annals of Operational Research,* 242, 115–132. https://doi.org/10.1007/s10479-015-2002-6.

National Institute of Standards and Technology. (2011): The NIST Definition of Cloud Computing. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

Portal, G.M., Ritt, M., and Borba, L.M. (2016): Simulated annealing for the machine reassignment problem. *Annals of Operational Research*, 242, 93–114. https://doi.org/10.1007/s10479-014-1771-7

Praveen, S. P., Ghasempoor, H., Shahabi, N., and Izanloo, F. (2023): A hybrid gravitational emulation local search-based algorithm for task scheduling in cloud computing. *Mathematical Problems in Engineering*, *2023*.

ROADEF/EURO Challenge (2012): Google Machine Reassignment Problem. http://challenge.roadef.org/2012/en/

Saber, T., Gandibleux, X., O'Neill, M., Murphy, L.B.E., and Ventresque, A.A. (2019): comparative study of multi-objective machine reassignment algorithms for data centres. *Journal of Heuristics,* 26, 119-150

Sharma, V., and Tripathi, A. K. (2022): A systematic review of meta-heuristic algorithms in IoT-based applications. *Array*, *14*, 100164.

Souza, F., Grimes, D., and O'Sullivan, B. (2023): A large neighbourhood search approach for the data centre machine reassignment problem. *Artificial Intelligence and Cognitive Science, 397–40.*

Tan, J. S., Goh, S. L., Kendall, G., and Sabar, N. R. (2021): A survey of the state-of-the-art of optimization methodologies in school timetabling problems. *Expert Systems with Applications, 165*, 113943.

Turky, A. (2019): ''Bi-level hyper-heuristic approaches for combinatorial optimization problems,'' Ph.D. dissertation, School Sci., RMIT Univ., Melbourne VIC, Australia, 2019

Turky, A., Sabar, N. R., and Song, A. (2017): An evolutionary simulated annealing algorithm for Google machine reassignment problem. *Springer,* 431-422

Turky, A., Sabar, N. R., and Song, A. (2017): Multi-neighbourhood great deluge for Google machine reassignment problem. *Springer,* 10593, 706-715

Turky, A., Sabar, N. R., and Song, A. (2018): Cooperative evolutionary heterogeneous simulated annealing algorithm for Google machine reassignment problem. *Genetic Program. Evolvable Mach.*, 19, 1–2, 183–210.

Turky, A., Sabar, N. R., Dunstall, S., and Song, A. (2020): Hyper-heuristic local search for combinatorial optimization problems. *Knowledge-Based Systems, 205,* 106264.

Velliangiri, S., Karthikeyan, P., and Vinoth Kumar, V. (2021): Detection of distributed denial of service attacks in cloud computing using optimization-based deep networks. *Journal of Experimental and Theoretical Artificial Intelligence,* 33(3), 405–424.

Wang, Z., Lü, Z., and Ye, T. (2016): *Multi-neighbourhood local search optimization for machine reassignment problem. Computers & Operations Research, 68, 16–29.*

Wu, G., Mallipeddi, R., and Suganthan, P. N. (2019). Ensemble strategies for population-based optimization algorithms–A survey. *Swarm and evolutionary computation*, 44, 695-711.

Zhang, Y., and Geng, P. (2022): Multi-Task Assignment Method of the Cloud Computing Platform Based on Artificial Intelligence. *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9584703/*