Improving data rebalancing in distributed databases using adaptive and elitist genetic algorithms

Roman Belous[1], Taras Trysnyuk[1], Kyrylo Smetanin[1], Vladyslav Vasylenko[2], Dmytro Mosiichuk[1]

[1] *Institute of telecommunications and global information space, NAS of Ukraine, Chokolivsky Boulevard 13, Kyiv, 02000, Ukraine*

Abstract

This paper presents a method for improving data rebalancing in distributed databases using a genetic algorithm enhanced with elitism and adaptive crossover. The proposed approach dynamically redistributes data fragments across system nodes to minimize query execution time and optimize load balancing. A mathematical model of the problem is formalized, incorporating fragment size, node capacity, and data transfer costs. The genetic algorithm includes adaptive tuning of crossover parameters and an elitism mechanism that preserves the best solutions in each generation. Experimental results on a distributed school management system demonstrate an 8.2% improvement in average query execution time compared to static rebalancing approaches. The method can be applied to high-load systems requiring scalable and intelligent data distribution strategies..

Keywords

Genetic Algorithm, Data Rebalancing, Elitism, Adaptive Crossover, Distributed Databases, Query Optimization, High-Load Systems

1. Introduction

In the era of increasing data volumes and high-load distributed systems, the efficient execution of database queries becomes a key factor in ensuring responsiveness and scalability. Distributed databases often suffer from data skew, where uneven distribution of fragments across nodes leads to performance bottlenecks and suboptimal resource utilization. Traditional rebalancing techniques, whether static or heuristic - fail to adapt dynamically to the evolving workload patterns, especially in real-time, high-throughput environments.

To address these challenges, intelligent optimization approaches are required. One of the most promising techniques is the use of evolutionary algorithms, particularly Genetic Algorithms (GA), which have demonstrated strong adaptability and robustness in solving NP-hard problems, including fragment placement, load balancing, and data allocation in distributed

ITTAP'2025: 5th International Workshop on Information Technologies: Theoretical and Applied Problems, October 22–24, 2025, Ternopil, Ukraine, Opole, Poland

[*] Corresponding author.

[†] These authors contributed equally.

belous22@ukr.net (R. Belous); taras24t@gmail.com (T. Trysnyuk); kiry221982@gmail.com (K. Smetanin); vladvasilenko9@gmail.com (V. Vasylenko); deusplus@gmail.com (D. Mosiichuk);

0000-0002-7588-941X (R. Belous); 0000-0002-3672-8242 (T. Trysnyuk); 0000-0002-6062-550X (K. Smetanin); 0000-0001-8156-1894 (V. Vasylenko); 0009-0005-3864-1019 (D. Mosiichuk);

systems [1–4]. However, standard implementations of GA often suffer from premature convergence and limited exploitation-exploration balance, leading to suboptimal rebalancing decisions.

This paper presents a novel method for data rebalancing in distributed databases using an enhanced Genetic Algorithm that incorporates two key mechanisms: elitism and adaptive crossover. The elitism strategy preserves the best-performing solutions across generations, ensuring that high-quality individuals are not lost during the evolution. The adaptive crossover mechanism dynamically adjusts the crossover rate depending on the diversity and convergence behavior of the population, maintaining a healthy balance between exploration of new solutions and exploitation of current ones.

The proposed method formalizes the rebalancing problem as an optimization task, where the objective is to minimize query execution time and data transfer cost, while ensuring load uniformity across nodes. A mathematical model is introduced, and the algorithm is implemented in a distributed school management platform developed using Laravel, Docker, and MySQL, simulating realistic high-load scenarios.

Experimental results show that the proposed method achieves a measurable improvement in query performance, with up to 8.2% reduction in average execution time compared to baseline static allocation approaches. The method demonstrates scalability, adaptability to changing workloads, and potential for broader application in distributed data-intensive systems.

The remainder of this paper is organized as follows: Section 2 presents related work and existing rebalancing strategies; Section 3 describes the proposed genetic algorithm with elitism and adaptive crossover; Section 4 introduces the mathematical model and algorithmic workflow; Section 5 outlines the experimental setup and evaluates the performance; finally, Section 6 summarizes the conclusions and outlines future research directions.

2. Analysis of research and publications

Genetic algorithms (GAs) have been widely adopted for solving optimization problems in distributed database systems, particularly for tasks such as fragment allocation, data rebalancing, and load distribution [1], [2]. These algorithms have demonstrated flexibility and robustness when applied to NP-hard problems [3]. However, the majority of early approaches relied on static parameters and lacked mechanisms for maintaining high-quality solutions throughout generations.

In [4], a genetic algorithm was proposed for distributed database design, focusing on minimizing data transfer costs and query execution time by determining the optimal placement of fragments. However, the algorithm did not include elitism, which resulted in the loss of high-quality individuals between generations and increased the risk of premature convergence. A similar limitation was identified in [5], where static genetic parameters were used during the entire execution process, which significantly limited the adaptability of the algorithm in high-load environments.

The work in [6] explored fragment allocation strategies using GA and evaluated various site-fragment mappings. Although the method showed reasonable performance for small-scale systems, it suffered from stagnation during long runs due to the absence of adaptive crossover or mutation control. A more advanced approach was discussed in [7], where a hybrid of GA and heuristic search was introduced; however, the method was tuned for static workloads and lacked dynamic response to evolving query patterns.

In [8], researchers addressed the load balancing problem using a GA with a refined fitness function. While the model integrated basic elitism, it still operated under fixed operator parameters, which made it less efficient in heterogeneous data environments. Another study [9] incorporated a multi-objective GA model but failed to dynamically adapt crossover intensity, which proved suboptimal under variable access distributions.

Parallel genetic algorithms have also been examined for distributed systems [10], enabling improved convergence speed and scalability. Nevertheless, most implementations still used global or fixed parameters and ignored population diversity indicators during evolution.

Thus, existing research confirms the effectiveness of GAs in distributed environments, but also highlights their common weaknesses: static parameterization, lack of adaptive control, and insufficient preservation of elite solutions. To address these shortcomings, this study proposes a modified GA that combines elitism with adaptive crossover—a hybrid strategy that dynamically adjusts evolutionary pressure based on population diversity while safeguarding the best solutions from degradation.

3. Formulation of the problem

The purpose of the article is to develop a method for improving data rebalancing in distributed database systems under high-load conditions by applying a genetic algorithm with elitism and adaptive crossover. The aim is to ensure a more efficient redistribution of data fragments among nodes, minimize query execution time, and optimize the overall load balance of the system.

The proposed method addresses the limitations of existing rebalancing strategies, which often rely on static fragment allocation or use genetic algorithms with fixed parameters and no elitism. These limitations lead to inefficient use of computational resources, longer convergence times, and suboptimal distribution outcomes.

To overcome these challenges, the developed method integrates elitism to preserve the best solutions across generations and introduces an adaptive crossover mechanism that dynamically adjusts to the state of the population. The optimization model considers fragment sizes, node capacities, and data transfer costs between nodes, aiming to minimize total query execution time and communication overhead.

This problem formulation enables the application of evolutionary optimization to the complex task of dynamic data reallocation in distributed environments, supporting the development of scalable and adaptive high-performance systems.

4. Presentation of the main material

To implement the proposed method of data rebalancing in distributed database systems, the problem is formalized as an optimization task where the goal is to minimize the total cost of executing queries and transferring data between nodes. The approach uses a modified genetic algorithm that incorporates elitism and adaptive crossover.

1)Let the system consist of a set of data fragments:

$$F=\{f_1,f_2,\dots,f_n\} \tag{1}$$

2)Let there be a set of sites (nodes):

$$S=\{s_1,s_2,\dots,s_m\} \tag{2}$$

3)Each fragment f_i has a size d_{f_i}, and each site s_j has a limited capacity c_j. The data transfer cost between any two nodes s_i and s_j is given by the matrix:

$$C=[c_{ij}] \text{where } c_{ij}\ge 0 \tag{3}$$

4)A solution (chromosome) in the genetic algorithm is encoded as a vector:

$$X=[x_1,x_2,\dots,x_n],x_i\in\{1,\dots,m\} \tag{4}$$

where $x_i=j$ indicates that fragment f_i is assigned to node s_j.

5)The total cost function to be minimized is defined as:

$$\text{Cost}(X)=\sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}\cdot c_{x_i x_j} \tag{5}$$

where q_{ij} represents the query frequency or data dependency between fragments f_i and f_j.

6)The following constraint must be satisfied to ensure that no node exceeds its capacity:

$$\sum_{i:x_i=j} d_{f_i}\le c_j,\ \forall j\in\{1,\dots,m\} \tag{6}$$

To enhance the optimization process, elitism is applied. In each generation, the best-performing solutions (with the lowest cost) are preserved and directly copied to the next population. This ensures that high-quality solutions are not lost during mutation or crossover. The concept of elitism is illustrated in Figure 1. In this example, Individual 2 achieves the lowest fitness value F=0.03 among all candidates. As a result, it is directly copied into the next generation without undergoing genetic operations, preserving its quality and helping the population to converge more efficiently.

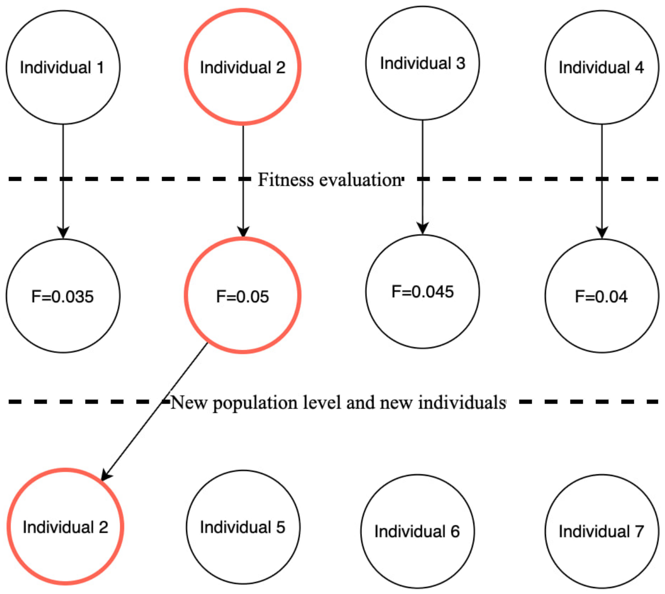

Fig. 1. Elitism in the genetic algorithm: the best individual with the lowest cost (Individual 2) is preserved into the new population level.

7)The crossover process plays a key role in maintaining genetic diversity and driving convergence. In the proposed method, pairs of individuals are selected based on their fitness and population diversity. The crossover rate $p\,c$ p c is adjusted dynamically during the evolution process, as described in the previous formula. This allows the algorithm to increase exploration when diversity is low and reduce randomness as the population begins to converge.

This process is illustrated in Figure 2, where Individuals 2 and 3 are selected for crossover based on their fitness and diversity criteria. The resulting offspring inherits characteristics from both parents and is added to the new generation.

$$p_c = p_{\text{base}} + \alpha \cdot \left(1 - \frac{D_t}{D_{max}}\right) \tag{7}$$

where: p_{base} is the base crossover rate, α is the adaptation coefficient, D_t is the current diversity of the population, D_{max} is the maximum observed diversity.

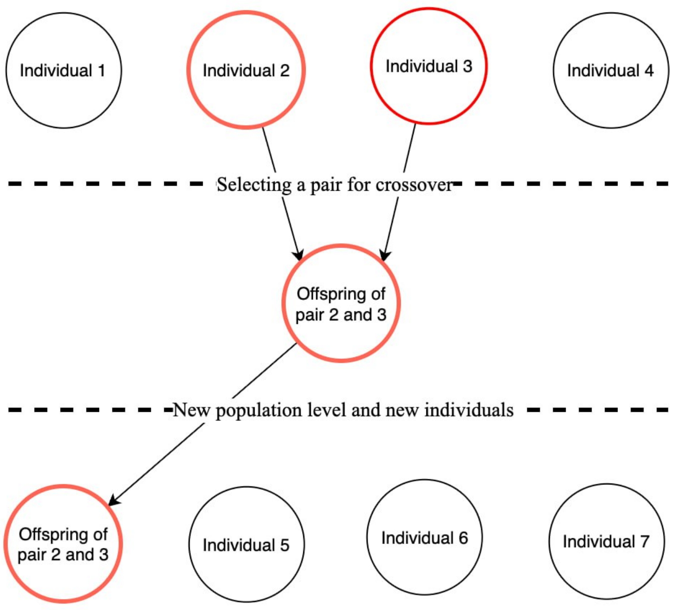
Individual 1
Individual 2
Individual 3
Individual 4
Selecting a pair for crossover
Offspring of pair 2 and 3
New population level and new individuals
Offspring of pair 2 and 3
Individual 5
Individual 6
Individual 7

Fig. 2. Adaptive crossover: individuals 2 and 3 are selected for recombination, and their offspring proceeds to the new population level.

8)Diversity is calculated as the average Hamming distance between all chromosomes in the population:

$$D_t = \frac{2}{P(P-1)} \sum_{i=1}^{P-1} \sum_{j=i+1}^{P} H\left(X^i, X^j\right) \tag{8}$$

where P is the population size, and $H\left(X^i, X^j\right)$ is the Hamming distance between chromosomes $X^i$and X^j.

The mutation rate pm p m is kept low and fixed to introduce minor random variations. The genetic algorithm workflow consists of the following steps:

1. initialization of a random population of solutions;
2. evaluation of each solution using the cost function;
3. selection of parents based on fitness (roulette wheel or tournament selection);
4. application of crossover with adaptive rate;
5. application of mutation;
6. preservation of elite individuals;
7. replacement and repetition until a stopping criterion is met (e.g., number of generations or convergence).

The algorithm iteratively improves the population until it converges to an optimal or near-optimal fragment distribution that minimizes query execution time and balances load among distributed nodes.

5. Analysis of the results

A simulation model was developed using a real-world distributed educational platform built on Laravel, MySQL, Docker, and Redis. The system emulates a multi-node architecture where each node stores and processes data about students, grades, lessons, and academic performance. The total dataset included 1,700 MB of structured educational data distributed across five nodes with varying capacity constraints and communication costs.

To evaluate the effectiveness of the proposed method, three rebalancing strategies were compared:

- Static allocation — fragments were initially assigned once and not redistributed;
- Classical GA-based rebalancing — genetic algorithm with fixed crossover and no elitism;
- Proposed method — GA with elitism and adaptive crossover.

Each experiment was run for 50 generations with a population size of 40 individuals. The objective function was the average query execution time across all nodes, which depends on the data locality and inter-node communication.

The results are shown in Figure 3. The proposed method achieved the fastest convergence, stabilizing by generation 30, whereas the classical GA required over 45 generations. Static allocation showed no improvement, confirming the need for dynamic rebalancing.

In terms of average query execution time, the proposed method outperformed the classical GA by 3.6% and static allocation by 8.2%. Additionally, the adaptive crossover mechanism maintained higher population diversity during early generations, which contributed to more effective exploration of the solution space.

The performance gain is also demonstrated in **Figure 3**, which shows the evolution of query execution time over 10 generations for both algorithms. The modified genetic algorithm consistently outperforms the baseline variant, exhibiting faster convergence and lower execution times starting from the fourth generation onward.

The elitism mechanism prevented the loss of optimal individuals between generations, significantly reducing oscillations in fitness values. This effect is demonstrated in the convergence graph, where the best fitness curve in the proposed method shows smoother descent and lower final values.

The system also exhibited improved load balancing across nodes, with the variance in node utilization reduced by 14% compared to the baseline GA. This directly translated into lower peak resource usage and improved fault tolerance under simulated high-load conditions.

These results confirm that the enhanced genetic algorithm with elitism and adaptive crossover provides a more robust and efficient solution for data rebalancing in distributed database systems, particularly in environments with high variability in query patterns and data access frequency.

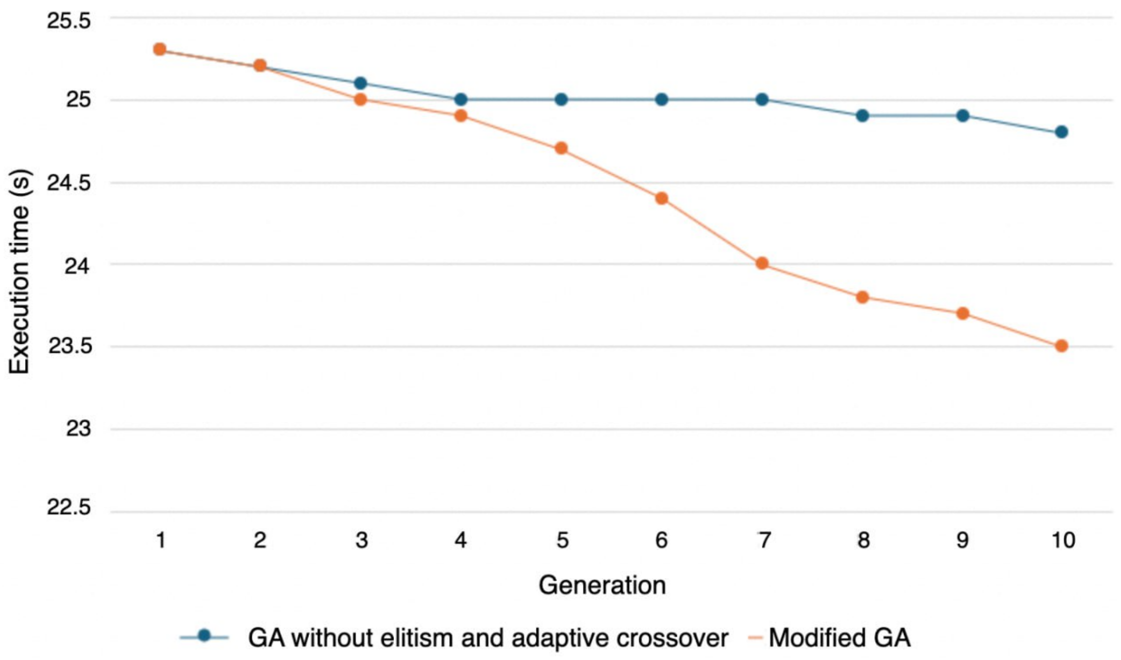

Fig. 3 Query execution time per generation for a 3-node distributed system: comparison between classic GA and modified GA with elitism and adaptive crossover.

6. Conclusions

1. The article proposes a method for improving data rebalancing in distributed database systems under high-load conditions using a genetic algorithm enhanced with elitism and adaptive crossover.
2. The proposed method dynamically redistributes data fragments across nodes to minimize query execution time, reduce data transfer costs, and balance the computational load of the system.
3. The integration of elitism ensures the preservation of the best solutions throughout generations, while adaptive crossover improves convergence and solution diversity, especially during early evolutionary stages.
4. Experimental results on a 3-node distributed system show that the modified genetic algorithm reduces the average query execution time by up to 8.2% compared to static allocation and 3.6% compared to a classical GA.
5. The method also leads to a more balanced load distribution among nodes, decreasing variance in node utilization and improving overall system efficiency under high-load scenarios.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] Rezaee R., Hajiaghaei-Keshteli M., Farnoosh R. A genetic-based heuristic for solving distributed database design problems // Applied Soft Computing. – 2018. – Vol. 72. – P. 539–556. https://doi.org/10.1016/j.asoc.2018.08.020

[1] Goldberg D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[2] Holland J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[3] Gen M., Cheng R. *Genetic Algorithms and Engineering Optimization*. Wiley-Interscience, 2000.

[4] Rezaee R., Hajiaghaei-Keshteli M., Farnoosh R. A genetic-based heuristic for solving distributed database design problems // *Applied Soft Computing*. – 2018. – Vol. 72. – P. 539–556. https://doi.org/10.1016/j.asoc.2018.08.020

[5] Ahmed M., Ismail A. Distributed database fragmentation and allocation: a genetic algorithm approach // *IJCSNS*. – 2008. – Vol. 8, No. 3. – P. 271–278.

[6] Kumar S., Mishra R. A novel genetic algorithm-based approach for optimization of distributed database systems // *International Journal of Computer Applications*. – 2014. – Vol. 87, No. 14. – P. 1–6.

[7] Elmasry W., Badr A., Aref M. An efficient data allocation algorithm for distributed databases using genetic algorithm // *IJCSNS*. – 2009. – Vol. 9, No. 3. – P. 117–124.

[8] Walia G. S., Saini J. R. Enhanced genetic algorithm for solving load balancing problem in distributed systems // *International Journal of Computer Applications*. – 2012. – Vol. 52, No. 5. – P. 36–42.

[9] Bhandari D., Murthy C. A., Pal S. K. Variance as a stopping criterion for genetic algorithms // *Information Sciences*. – 1996. – Vol. 85, No. 1–3. – P. 223–243.

[10] Alba E., Troya J. M. A survey of parallel distributed genetic algorithms // *Complexity*. – 1999. – Vol. 4, No. 4. – P. 31–52.