Fair Bracket Generation Through a Modified Fisher–Yates Shuffle: Implementation and Initial Evaluation in the GOHit Platform

Varis Anggito [1], Koko Wahyu Prasetyo [2*], Go Frendi Gunawan [3]

[1,2,3] *Faculty of Science and Technology, Universitas Bhinneka Nusantara, Malang, Indonesia*

Keywords
fisher-yates shuffle; randomization algorithm; tournament bracket generation; weighted randomization; ranking-based pairing;

Correspondence Email:
koko@ubhinus.ac.id

Abstract

The fairness and transparency of tournament brackets play a critical role in digital competition platforms, yet many systems still rely on manual or deterministic ordering that can introduce bias. This study implements and evaluates a weighted adaptation of the Fisher–Yates Shuffle algorithm in the GOHit web-based competition management system to address these limitations. The weighted approach divides participants into ranking-based groups prior to shuffling to achieve a balance between randomness and competitive fairness. The algorithm was integrated into a PHP CodeIgniter 4 environment and tested using three ideal bracket sizes (8, 16, and 32 participants) across 30 independent trials each. Results show that the algorithm consistently produced consistent and unique bracket permutations, with no duplication or recurring patterns. Visual and comparative analysis further demonstrate substantial displacement of participant positions when compared to manual sequencing, reducing predictability while maintaining structural fairness. These findings indicate that the weighted Fisher–Yates Shuffle offers an effective, lightweight, and reproducible solution for fair bracket generation in small- to medium-scale digital competitions. The approach also provides a practical foundation for future enhancements in fairness-sensitive scheduling systems.

1. Introduction

Digital competitions or tournaments depend heavily on transparent and fair scheduling mechanisms. When brackets are arranged manually or generated through fixed ordering, participants may experience unbalanced match-ups, predictable patterns, or even unintentional bias. Earlier rounds often determine competitive momentum, making a fair bracket not simply a technical requirement but a determinant of participant satisfaction and trust. As online events grow across universities and organizations, the need for automated, impartial, and auditable bracket generation becomes increasingly pressing.

Despite this need, many locally developed competition platforms still rely on sequential or partially randomized ordering. Such approaches fail to provide guarantees of fairness, especially when tournaments involve ranking systems. Completely random pairing can also be problematic because it may position top-

ranked participants against each other prematurely. Thus, real-world competition systems require a mechanism that achieves balanced unpredictability: randomness that does not neglect competitive structure.

The Fisher–Yates Shuffle is widely recognized as a lightweight, uniform, and unbiased randomization algorithm for generating permutations. However, its traditional form does not incorporate fairness logic required for competitive pairing. This creates a practical gap between theoretical randomization models and the needs of digital tournament management. To address this gap, this study implements a weighted adaptation of the Fisher–Yates Shuffle within the GOHit competition management platform. The adaptation introduces rank-based grouping prior to shuffling, aiming to balance fairness and randomness simultaneously.

The contribution of this work lies in demonstrating an implementation that is simple enough for small-scale academic platforms yet robust enough to produce consistent, fair bracket outcomes. By evaluating the algorithm through repeated trials, the study provides empirical evidence on its suitability for real-time competition environments and its potential to replace manual bracketing practices.

1.1 Literature Review

Randomization is a fundamental principle in digital systems that aim to ensure fairness, transparency, and unpredictability. The Fisher–Yates Shuffle (FYS) algorithm is widely regarded as one of the most efficient methods for producing unbiased permutations. Its theoretical correctness and uniform probability distribution have been formally proven, confirming its reliability for practical use in random ordering tasks (Eberl, 2016).

In applied computing, FYS has demonstrated strong performance in maintaining non-repetitive randomization across various domains. It has been successfully implemented in computer-based examinations to generate unique question sequences and prevent duplication, improving both system performance and perceived fairness (Febriani et al., 2021; Wijaya & Chang, 2023). Similarly, its use in interactive educational media has yielded effective randomization without pattern repetition, as shown in Android-based puzzle and language-learning applications (Irfan et al., 2020; Kannabi & Norhikmah, 2022).

Several Indonesian studies have explored FYS in competition or testing contexts. Rizky et al. (2021) applied it to automate tournament seeding in pencak silat competitions, replacing manual draws and demonstrating improved fairness and efficiency (Rizky et al., 2021). Comparative analyses confirm its simplicity and computational advantages over other random generators such as the Linear Congruent method (Saputro, 2024). Moreover, formal verification studies have validated its unbiased behavior at the algorithmic level (Zetzsche et al., 2025).

Beyond educational uses, several studies propose domain-specific modifications to enhance fairness and randomness. Aishwarya and Beny (2015) introduced an iterative variant of FYS to improve data security and distribution balance (Aishwarya & Beny, 2015). Such approaches illustrate the algorithm's flexibility in addressing fairness constraints. This idea directly relevant to competitive tournament systems like GOHit.

Overall, existing literature consistently supports the Fisher–Yates Shuffle as a lightweight, reproducible, and bias-free randomization mechanism. However, most implementations prioritize uniform randomness rather than rank-based fairness. The present study addresses this gap by adapting FYS into a weighted fairness-oriented variant, bridging theoretical uniformity with practical fairness in digital competition management.

2. Research Methods

This study employed an experimental software implementation approach to evaluate the Weighted Fisher–Yates Shuffle algorithm within the GOHit competition management platform. The research was designed to observe algorithmic fairness and consistency across repeated randomization trials using controlled data inputs. Empirical software engineering principles guided the procedure by implementing, executing, measuring, and validating the algorithmic behavior under equivalent conditions (Sarmah & Chakrabarty, 2015)

The implementation was coded in PHP (CodeIgniter 4) and integrated into the Competition Controller of GOHit using the Model–View–Controller (MVC) architecture. The algorithm adopted the classical Fisher–Yates logic,

which iteratively swaps array elements at uniformly distributed random indices to produce unbiased permutations (Eberl, 2016).

To achieve fairness at the tournament level, a weighted modification was introduced by dividing participants into two ranked groups (upper and lower halves) before shuffling. Each subgroup was randomized independently and paired afterward, ensuring balanced match-ups between stronger and weaker players. This variant maintained the algorithm's uniform randomness while incorporating domain-specific fairness constraints, a technique consistent with recent adaptive shuffling designs (Yadav et al., 2017).

Testing involved three ideal bracket sizes (8, 16, and 32 participants), each executed in 30 independent trials. This trial count was selected to provide statistically meaningful variation while maintaining computational feasibility. It is commonly applied in empirical algorithm evaluations to balance sample reliability with runtime cost (Nurhayati et al., 2021). Across 90 total runs, each shuffle instance was logged for output uniqueness and pairing repetition.

Output correctness was validated using black-box testing, where the system's outputs were compared without accessing internal algorithm states. This verification method focuses on the functional equivalence between expected and actual outcomes which are a standard in software testing for randomized systems (Zetzsche et al., 2025).

All experiments were conducted on a Windows environment using Apache 2.4, PHP 8.2, and MySQL 8.0. Emphasis was placed on correctness, reproducibility, and fairness which are aligned with prior Fisher–Yates implementations in web-based educational systems (Febriani et al., 2021).

The dataset consisted of fictional participant identifiers and point rankings, ensuring ethical compliance and reproducibility. The algorithm and validation procedure were designed to generate three core empirical outputs:

a) system-generated bracket outcomes
b) quantitative accuracy data for ideal brackets
c) comparative results between manual and algorithmic pairing

3. Result and Discussion

The weighted Fisher–Yates shuffle was successfully implemented in the GOHit competition management platform. Figure 1 presents the PHP code segment that performs randomization through two main functions: fisherYatesShuffle() and ACAK(). The former executes an in-place swap of array elements to ensure unbiased randomness, while the latter groups participants by ranking points before applying the shuffle process. This modification introduces proportional fairness without altering the algorithm's original uniform distribution property.

The system generated stable and non-repetitive results across repeated runs. Figure 2 shows examples of automatically generated tournament brackets for 8- and 16-participant configurations. The random order of participants differed in each execution, confirming the absence of deterministic patterns. The even distribution of higher- and lower-ranked participants across both halves of the bracket illustrates the intended balance between fairness and randomness in competitive pairing.

```php
public function ACAK($pointData, $typeSport)
{
if (count($pointData) < 2) {
return []; // Kosong jika tidak cukup peserta
}
// Urutkan berdasarkan nilai tertinggi dari typeSport
usort($pointData, function ($a, $b) use ($typeSport) {
return $b[$typeSport] <=> $a[$typeSport];
});
$midPoint = ceil(count($pointData) / 2);
$topRanked = array_slice($pointData, 0, $midPoint);
$others = array_slice($pointData, $midPoint);
// Acak masing-masing kelompok74
$this->fisherYatesShuffle($topRanked);
$this->fisherYatesShuffle($others);
$pairs = [];
$pairCount = min(count($topRanked), count($others));
// Buat pasangan dari dua kelompok
for ($i = 0; $i < $pairCount; $i++) {
$pair = [
$topRanked[$i]['id_user'],
$others[$i]['id_user']
];
// Random urutan pasangan
if (rand(0, 1) === 1) {
$pair = array_reverse($pair);
}
$pairs[] = $pair;
}
// Tambahkan peserta sisa (tanpa pasangan)
$remaining = array_slice($topRanked, $pairCount);
$remaining = array_merge($remaining, array_slice($others, $pairCount));
foreach ($remaining as $peserta) {
$pairs[] = [$peserta['id_user']]; // pasangan tunggal (bye)
}
return $pairs;
}
```

```php
private function fisherYatesShuffle(&$array)
{
$count = count($array);
for ($i = $count - 1; $i > 0; $i--) {
$j = rand(0, $i);
$temp = $array[$i];
$array[$i] = $array[$j];
$array[$j] = $temp;
}
}
```

Fig 1. Fisher-Yates Algorithm Implementation

Empirical findings summarized in Table 1 demonstrate that all 30 randomization trials for each bracket size (8, 16, and 32 participants) produced 100% unique outcomes with no duplication. This result indicates that the algorithm consistently meets the theoretical condition of unbiased permutation generation, as explained in previous evaluations of the Fisher–Yates algorithm. The 30-trial setup provided sufficient repetition to verify stability while maintaining computational practicality, which is an appropriate scope for system-level testing.

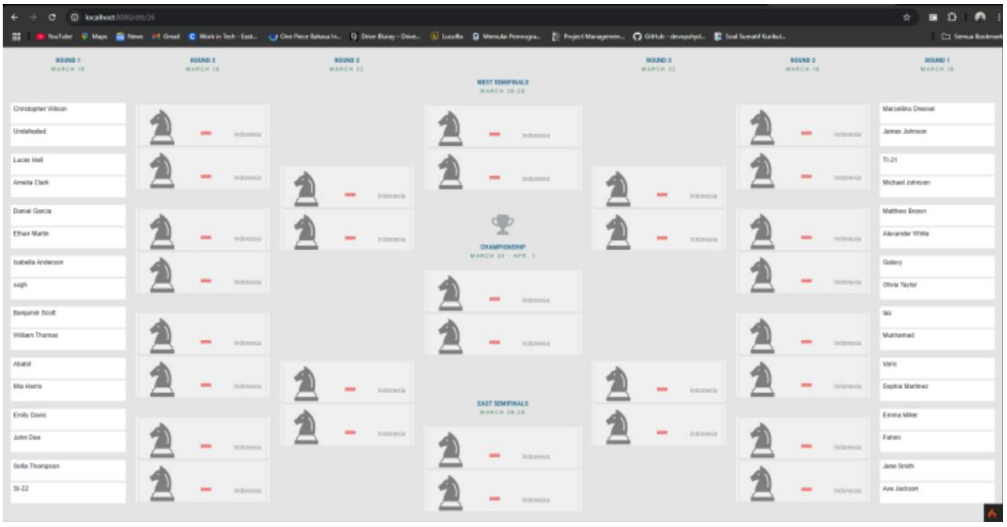

Fig 2. Bracket Randomization Result

Table 1. Randomization Accuracy for Ideal Bracket Sizes

No. of Participants	Trials per Scenario	Unique Outcomes	Duplicate Patterns
8	30	30	0
16	30	30	0
32	30	30	0

Further comparison in Table 2 highlights the difference between manual and algorithmic randomization. Under manual sequencing, participant order remained constant across tournaments, which can introduce bias or predictability. In contrast, the weighted Fisher–Yates algorithm displaced participant positions by an average of approximately 3 to 4 slots, ensuring that no player consistently occupied the same bracket position. This balanced displacement supports the principle of fair competition while maintaining structural integrity. Similar results were observed in educational and assessment systems that applied the same algorithm for randomized item distribution (Nurhayati et al., 2021; Febriani et al., 2021).

Table 2. Comparison Sample of Bracket Ordering: Manual vs Weighted

Participant (fictional dataset)	Manual Bracket Position	Algorithmic Bracket Position	Difference (Δ Position)
Andi Pratama	1	5	4
Budi Santoso	2	8	6
Citra Rahma	3	1	−2
Dwi Kusuma	4	6	2
Eka Putri	5	2	−3
Fajar Hidayat	6	7	1
Galih Saputra	7	3	−4
Hani Wulandari	8	4	−4

Overall, the results confirm that the proposed approach successfully combines fairness, randomness, and computational efficiency. The implementation demonstrates that a lightweight modification to the classical Fisher–Yates shuffle can significantly improve transparency and participant trust in online tournament systems. These findings align with the fairness-oriented randomization models described in prior studies (Yadav et al., 2017; Saputro, 2024), validating the algorithm's relevance beyond academic demonstrations.

For future development, several practical opportunities remain. First, the algorithm could be tested with larger participant datasets or multiple concurrent tournaments to observe its stability under higher computational loads. Second, integrating a secure random number generator could further enhance randomness quality and prevent predictability in long-term usage. These follow-up studies would strengthen GOHit's reliability as a digital competition platform and encourage broader application of fair randomization methods in other academic or organizational settings.

4. Conclusions

This study successfully implemented and validated the weighted Fisher–Yates shuffle algorithm in the GOHit competition management platform. The algorithm effectively produced fair and fully randomized tournament brackets while maintaining computational efficiency in a web-based environment.

Empirical results showed completely unique outcomes in all 30 trials for ideal bracket sizes of 8, 16, and 32 participants, confirming the algorithm's uniform randomness and absence of repeated pairings. The randomization process displaced participant positions across runs, increasing fairness and eliminating bias from manual sequencing. The weighted adaptation added a simple fairness constraint by pairing participants

from different ranking levels, ensuring balanced matchups without sacrificing randomness. This combination of fairness, simplicity, and speed demonstrates that the Fisher–Yates shuffle can serve as a reliable foundation for small- to medium-scale digital competitions.

Further improvements could enhance statistical confidence and applicability. Increasing the number of trials beyond 30 would allow more robust verification of randomness distribution and reveal subtle bias patterns that may not appear in limited samples. Additionally, testing on non-ideal bracket sizes (where the number of participants is not a multiple of 8) could validate the algorithm's flexibility for real-world tournaments with uneven registrations. Exploring these directions would strengthen both the practical and analytical robustness of the GOHit system.

Overall, this research contributes a lightweight and reproducible randomization method that balances fairness and efficiency. The findings can guide future system enhancements or academic projects seeking to implement equitable competition scheduling in digital environments.

5. References

Aishwarya, C., & Beny, J. (2015). Novel architecture for data shuffling using Fisher–Yates Shuffle algorithm. International Journal of Scientific Research in Science, Engineering and Technology, 1(1), 387–390. https://doi.org/10.32628/IJSRSET151655

Eberl, M. (2016). Fisher-yates shuffle. Arch. Formal Proofs, 2016, 19.

Febriani, I., Ekawati, R., Supriadi, U., & Abdullah, M. A. M. (2021). Fisher–Yates Shuffle algorithm for randomization math exam on computer-based test. AIP Conference Proceedings, 2331(1), 060015. https://doi.org/10.1063/5.0042534

Irfan, M., Ramdhania, D. R., Nita, I. S., Priatna, T., & Atmadja, A. R. (2020). Design and build an early childhood puzzle educational game using the Fisher–Yates Shuffle algorithm as an Android-based scrambler for snippets. Proceedings of the 2020 6th International Conference on Wireless and Telematics (ICWT), 1–6. https://doi.org/10.1109/ICWT50448.2020.9243628

Kannabi, A. M., & Norhikmah, N. (2022). Implementation of the Fisher–Yates Shuffle game algorithm in learning Hijaiyah letters. SISTEMASI, 11(3), 2053. https://doi.org/10.32520/stmsi.v11i3.2053

Rizky, M. M., Asriyanik, A., & Lelah, L. (2021). Implementasi algoritma Fisher–Yates Shuffle pada bagan undian peserta pertandingan pencak silat. Progresif: Jurnal Ilmiah Komputer, 17(2), 1–6. https://doi.org/10.35889/progresif.v17i2.643

Saputro, N. D. (2024). Comparison of Fisher–Yates Shuffle and Linear Congruent algorithms for question randomization. Journal of Dinda: Data Science, Information Technology, and Data Analytics, 4(2). https://doi.org/10.20895/dinda.v4i2.1584

Wijaya, A., & Chang, F. D. (2023). Information system web design for class English proficiency test using Fisher–Yates Shuffle method. bit-Tech, 5(3). https://doi.org/10.32877/bt.v5i3.724

Yadav, M., Gautam, P. R., Shokeen, V., & Singhal, P. K. (2017). Modern fisher–yates shuffling based random interleaver design for SCFDMA-IDMA systems. Wireless Personal Communications, 97(1), 63-73.

Zetzsche, S., Tristan, J.-B., Lepoint, T., & Mayer, M. (2025). Verifying the Fisher–Yates Shuffle algorithm in Dafny. Journal of Applied Informatics and Computing.