A Review of Multi-Armed Bandit Algorithms in Player Modeling and Game Design

Xizhi Zhan[1]

School of Computer Science & Technology, Huazhong University of Science and Technology, Wuhan, China

Abstract. This paper explores the application of multi-armed bandit algorithms (MAB) in game design, focusing on player modeling and game optimization. The effectiveness of multi-armed bandit algorithms in modeling player characteristics such as skill level, play style, and social comparison orientation is investigated. The potential of MAB in optimizing game design elements like difficulty, rewards, and user interface is also explored. The paper presents empirical results from simulations and user studies and concludes by discussing the potential of MAB algorithms in game design and highlighting future research directions.

1 Introduction

The multi-billion dollar video game industry depends on a fine balance between software development and artistic creativity. Despite established templates and best practices, creating a successful game remains a complex endeavor, often requiring a blend of skill, intuition, and a touch of luck. Game designers employ various techniques, including extensive testing, beta releases, player feedback, and data analysis, to refine their games and understand player preferences. However, these methods are time-consuming and may not always provide a comprehensive understanding of the intricacies of player behavior.

Due to the intense competition and dynamic nature of the video game industry, developers are constantly looking for new and creative ways to draw in and keep players. Conventional approaches to game design frequently depend on subjective judgment and scant testing, which can be insufficient to yield a thorough understanding of player behavior and preferences. To address this limitation, MAB algorithms offer a systematic, data-driven framework that strikes a balance between exploration (finding out about player preferences) and exploitation (maximizing reward based on known information).

This is where the concept of MAB algorithms enters the picture. By striking a balance in the trade-off between exploration and exploitation, MAB algorithms present a novel method for game design optimization. Game designers can systematically experiment with various design variations and dynamically modify game parameters in response to real-time player feedback by utilizing the principles of MAB. This approach enhances player experience and

[1] Corresponding author: u202215549@hust.edu.cn

contributes to the overall success and profitability of the game. This paper focuses on two key areas:

1. **Player Modeling**: I investigate the potential of MAB algorithms to model player characteristics such as skill level, play style, and social comparison orientation. This entails developing MAB strategies that adapt to player behavior and dynamically adjust game content to enhance the player experience.
2. **Game Design Optimization**: MAB algorithms are applied to optimize game design elements such as difficulty levels, rewards, and the user interface. This adaptive approach allows real-time modifications based on player feedback and performance, aiming to create a more personalized and engaging gaming environment.

The objective of exploring the application of MAB algorithms in enhancing game design is pursued by focusing on two primary areas: player modeling and game design. By investigating how MAB methods can complement traditional approaches in these fields, insights are sought into how these algorithms can lead to more successful and profitable gaming experiences.

2 Literature Review

The application of MAB algorithms in game design is an emerging area of research that shows great potential. Several studies have explored how MAB can optimize key aspects of game design, particularly player modeling.

Gray, Robert C., Jichen Zhu, and Santiago Ontañón's study explores applying MAB algorithms to player modeling in multiplayer games [1]1. Their key contributions include proposing a framework to handle the complexity of multiplayer MAB modeling and introducing the "Multiplayer Regression Oracle" (MRO) to address the challenge of best-arm estimation in multiplayer environments. Additionally, they propose the "Multiplayer Forced Exploration" (MFE) algorithm to handle exploration strategies in these settings. This highly innovative study extends MAB applications from single-player to multiplayer environments and addresses key challenges in multiplayer game design. Using social comparison theory and exercise games as the research context provides a compelling application scenario for the theoretical framework. However, the paper only addresses two of the three major challenges in multiplayer settings, leaving social fairness concerns for future research, which can be seen as a limitation.

Another study [2]2, which focuses on player modeling using MAB algorithms, may be an earlier or related version of Gray et al.'s work.This study focuses on single-player modeling, potentially laying the groundwork for the multiplayer focus seen in subsequent research.

Several studies offer fresh perspectives on applying MAB to player modeling. Vinogradov and Harrison's study explores how to use MAB algorithms to dynamically update player models in experience-managed environments [3]. The study emphasizes dynamic player modeling, which could have important applications in game development and user experience optimization.

A closely related work investigates the use of contextual bandits to adapt to changing user preferences over time [4]. The article reveals the focus on dynamic user preferences—a crucial aspect of practical game design. Contextual bandits, as proposed in this research, offer an interesting approach that may be better suited to handling evolving user preferences than standard MAB algorithms.

Another paper is Using Multi-Arm Bandits to Optimize Game Play Metrics and Effective Game Design by Kenny Raharjo and Ramon Lawrence [5].This study explores the application of multi-armed bandit (MAB) algorithms to optimize player engagement metrics by testing various game designs. Raharjo and Lawrence implemented an epsilon-greedy

MAB algorithm on a simple game, "Diamond Hunter," which offers three different background themes. Their study confirmed that MAB algorithms can dynamically and effectively identify the most engaging design by maximizing playtime. The key contribution is a demonstration of MAB's ability to provide rapid convergence to the preferred game variant with minimal external input, making it a valuable tool for real-time design adaptation. A notable limitation, however, is that the study only examined aesthetic themes without exploring gameplay adjustments, which could affect the generalizability of MAB's optimization potential across more complex game elements.

The next work of Kunanusont, Kamolwan, Simon Lucas, and Diego Perez-Liebana presents a different approach by combining evolutionary algorithms with MAB methods to model player experience [6]. Their N-Tuple Bandit Evolutionary Algorithm offers an innovative means of capturing the complexity and dynamics of player experiences. This approach holds promise for future research that seeks to integrate player experience modeling with adaptive game design techniques.

Wang et al. propose an innovative approach to MAB in multi-player settings [7], introducing an upper confidence bound (UCB)-based algorithm for learning in environments with player-specific reward distributions. The study's central advancement is its ε-multi-player multi-armed bandit (ε-MPMAB) model, which allows for collaborative optimization by aggregating data across players with minimal regret. This algorithm demonstrates improved collective regret, particularly under conditions of small distributional discrepancy (ε), achieving almost a factor of M improvement over individual learning. This robust framework is particularly relevant for applications where similar, but not identical, tasks are performed by multiple agents. Despite its strength, the model's performance decreases under high distributional discrepancy, suggesting the need for further exploration into adaptability for broader multiplayer scenarios.

Raharjo examines how MAB algorithms, specifically epsilon-greedy and UCB [8], can be used to optimize various aspects of game design, including user engagement and game completion rates. The study focuses on dynamically adjusting game elements to minimize abandonment and maximize player retention. Through a hypothetical example and analysis of game play data, Raharjo illustrates that MAB algorithms can be effectively used for adaptive game design. This work provides valuable insights into using MAB for player experience optimization in a market where player engagement is crucial. A limitation noted in the study is its theoretical nature; while hypothetical scenarios are useful, actual deployment and empirical data on player interaction with adjusted game elements would further validate the approach's effectiveness.

Amiri and Sekhavat's study proposes a dynamic adjustment of game properties at runtime using MAB [9], applying an epsilon-greedy algorithm in a 3D roll-ball game to adjust game themes according to player preferences. Their work demonstrates that real-time modification of game parameters such as environment color and player speed based on player behavior can significantly enhance the player experience. This study contributes to the growing body of work that seeks to dynamically optimize game design, with real-time adjustments providing insights into player preferences and improving engagement. While the study shows promising results, it primarily focuses on a limited set of parameters, such as environment color and player movement speed, which may limit its generalizability to more complex games.

Kunanusont, Gaina, Liu, Perez-Liebana, and Lucas's work introduces the N-Tuple Bandit Evolutionary Algorithm for automatic game improvement [10], using AI to assist in game design. This algorithm explores a hybrid approach, combining evolutionary algorithms with MAB principles to optimize game parameters. Their study shows that this approach is particularly useful in balancing exploration and exploitation in large search spaces, offering robust results for game parameter tuning. This innovative combination of evolutionary algorithms and bandit methods represents a step forward in automatic game design, providing tools to dynamically evolve game parameters with minimal human intervention.

While the reviewed studies provide valuable insights into the use of multi-armed bandit (MAB) algorithms in both single-player and multiplayer contexts, this paper highlights several underexplored areas within the existing literature. Specifically, such as social comparison orientation and shifting play styles—dynamics often overlooked in favor of static player attributes. It also emphasizes contextual, personalized design adjustments, where real-time data could dynamically tailor elements like difficulty, rewards, and challenges to the individual player experience. Additionally, this paper considers multi-dimensional player engagement metrics, moving beyond simple retention rates to capture richer indicators such as emotional engagement and social connectivity. Unlike prior research, which frequently relies on static or narrowly dynamic models, this paper underscores the fluid and evolving nature of player preferences in real-time environments, demonstrating a more robust application of contextual bandits. Its innovative contribution lies in presenting a comprehensive review that integrates MAB algorithms into both player modeling and game design optimization. By synthesizing insights on player behavior with adaptive design strategies, this review establishes a cohesive framework for understanding how these algorithms can enhance player experience and drive game profitability. These findings reveal the significant potential of MAB algorithms to advance game design by effectively balancing player engagement, personalization, and business objectives, paving the way for a more adaptive, player-centered gaming experience.

3 Player Modeling with MAB Algorithms

In this chapter, I explore the potential of MAB algorithms to model player characteristics in dynamic gaming environments. Player modeling involves understanding various player traits—such as skill level, play style, and social comparison orientation—which allows game systems to adapt in real-time to enhance the overall gaming experience. This chapter delves into how MAB algorithms can be used to model these aspects effectively, discussing both the theoretical framework and practical implementations.

3.1 Introduction to Player Modeling with MAB

Player modeling is a critical component in modern game design, as it enables games to dynamically adjust to the behaviors, preferences, and performance levels of individual players. Traditional methods often involve static difficulty settings or pre-determined pathways, which fail to account for the real-time variations in player behavior. MAB algorithms, with their ability to balance exploration (trying new strategies) and exploitation (utilizing known successful strategies), offer a robust solution for this adaptive challenge.

MAB approaches allow game systems to continuously learn from player interactions, gradually improving the model of the player's preferences and abilities. This adaptive learning not only enhances the player’s experience by keeping the game challenging and engaging but also prevents frustration due to mismatches between player skill and game difficulty.

3.2 Modeling Skill Level

Player skill level varies significantly across different individuals and can change over time as players improve. A fundamental challenge in game design is to ensure that the game remains neither too easy nor too difficult, providing a smooth progression in difficulty that matches the player's evolving skill.

MAB algorithms can dynamically adjust difficulty levels by evaluating player performance after each round or task. In this context, *arms* represent the various choices available to the player, such as different game configurations. *Regret* measures the potential loss incurred by not selecting the optimal arm, or in gaming terms, the lost opportunity for a more engaging experience. For instance, if a player consistently performs well, the algorithm can increase the difficulty by selecting tougher game configurations (arms). Conversely, if the player struggles, the algorithm can reduce the difficulty. This approach minimizes regret, which in games refers to the lost engagement or enjoyment caused by a mismatch between the player's ability and the game's difficulty.

Several studies have applied MAB algorithms to dynamic difficulty adjustment. In these systems, MAB evaluates the performance of a player and adapts future game challenges accordingly. The epsilon-greedy algorithm or Upper Confidence Bound (UCB) strategies are commonly employed to ensure that the system continues to explore new difficulty levels while exploiting those that are most suitable for the player.

3.3 Play Style Recognition

Another key aspect of player modeling is recognizing a player's preferred play style. Some players might favor an aggressive, high-risk approach, while others might prefer a more defensive, exploratory, or strategic play style. By identifying these preferences, MAB algorithms can adjust the game environment to better suit the player, enhancing their immersion and satisfaction.

MAB can observe the player's choices (arms) across various game scenarios to infer their preferred style. For instance, if a player consistently chooses faster, more aggressive tactics, the algorithm can introduce game elements that encourage this behavior, such as more action-packed levels or rewards for high-speed completion. Conversely, if a player prefers a slower, more methodical approach, the algorithm can present challenges that require careful planning and patience.

In Vinogradov and Harrison's study [3], MAB algorithms were used to dynamically adapt game content based on the player's style. This application of MAB enables real-time content generation that responds to player preferences, such as offering more difficult puzzles for players who show a higher level of skill or providing narrative choices that align with the player's decision-making style.

3.4 Social Comparison Orientation in Multiplayer Games

Social comparison is a psychological factor that plays a significant role in multiplayer gaming environments. Players often measure their success against others, and their enjoyment and engagement can be influenced by how they perceive themselves relative to their peers. Modeling this social comparison orientation is particularly important in competitive or cooperative multiplayer games.

MAB algorithms can be used in multiplayer settings to model social comparison by tracking the performance of players relative to each other. For example, in Gray et al.'s study [2], the authors proposed a framework for handling the complexity of multiplayer MAB modeling by introducing the "Multiplayer Regression Oracle" (MRO) and "Multiplayer Forced Exploration" (MFE) algorithms. These algorithms adaptively manage exploration and exploitation strategies across multiple players, ensuring that each player receives a balanced challenge that matches their social comparison orientation. This can be especially useful in games that feature ranking systems or leaderboards, as it ensures a fair and competitive environment.

3.5 Dynamic Player Behavior Adaptation

One of the key benefits of using MAB algorithms in player modeling is their ability to adapt to player behavior dynamically. Unlike static models, MAB continuously updates its understanding of the player based on real-time feedback. As the player's behavior evolves, the game's response also evolves, creating a more personalized experience.

For example, if a player starts to experiment with different strategies or shows signs of fatigue or frustration (e.g., failing multiple times in succession), MAB algorithms can detect these changes and adjust the game accordingly. This could mean offering more accessible levels to help the player regain confidence or introducing new gameplay mechanics to re-engage the player.

In games that employ procedural generation or adaptive narratives, MAB can be used to adjust the content dynamically based on player interaction patterns. For instance, if a player frequently chooses exploration over combat, the game could generate more exploration-based challenges, such as puzzle-solving or resource gathering, while downplaying combat-heavy sections.

3.6 Challenges and Limitations

While MAB algorithms offer significant potential in player modeling, there are challenges to consider. One key limitation is the cold-start problem, where the algorithm initially lacks sufficient data to make accurate decisions about player preferences or skill levels. This issue can be mitigated by integrating MAB with other learning algorithms or by employing hybrid models that incorporate rule-based systems during the initial stages.

Additionally, MAB's effectiveness depends on selecting appropriate "arms" or options for the algorithm to choose from. In some cases, defining these arms (such as specific game parameters or difficulty settings) can be complex and may require extensive tuning to ensure that the algorithm balances exploration and exploitation effectively.

4 Game Design Optimization with MAB Algorithms

In this chapter, I focus on how MAB algorithms can be leveraged to optimize game design elements, such as difficulty levels, rewards, and user interface, in real time. The goal is to explore how MAB can dynamically adapt game mechanics and structure based on player feedback and performance, thus creating a personalized, engaging, and evolving gaming experience.

4.1 Introduction to Game Design Optimization

Game design traditionally relies on static systems or manual adjustments to fine-tune elements like difficulty, rewards, and user interface (UI). However, as games become more complex and player preferences more diverse, static models may struggle to provide an optimal experience for all players. MAB algorithms offer a solution by automating the process of game optimization, allowing the game to adapt in real time to a player's behavior, skill, and preferences.

In MAB applications for game design, each game element (e.g., difficulty setting, reward structure, or UI configuration) can be treated as an "arm" of the bandit. The algorithm continually evaluates player interactions with these elements, adjusting them to minimize "regret"—in this case, the loss of engagement, satisfaction, or fun due to suboptimal design choices.

4.2 Adapting Difficulty Levels

One of the most critical elements of game design is difficulty adjustment. If a game is too easy, it can become boring; if it is too hard, it can frustrate players. MAB algorithms are uniquely suited to adjust difficulty dynamically based on player performance, ensuring that the game stays challenging without becoming overly difficult.

In dynamic difficulty adjustment systems, MAB algorithms observe how well the player performs (e.g., win rates, time taken to complete a task, or failure rates) and use this feedback to modify the game's difficulty in subsequent rounds or levels. For instance, if a player consistently overcomes challenges easily, the MAB can select harder configurations (arms) to increase difficulty. Conversely, if the player struggles, the MAB may reduce difficulty by selecting easier configurations.

Several techniques are used for difficulty optimization. For example, the epsilon-greedy algorithm could be applied, allowing the system to experiment with different difficulty settings (exploration) while also favoring those that seem to match the player's skill (exploitation). This continuous adaptation ensures that players remain engaged and challenged at their level.

Amiri and Sekhavat's study on using MAB for real-time theme adjustment in a 3D game demonstrates how difficulty and environment can be dynamically optimized to match player performance [9]. The epsilon-greedy algorithm was used to modify the game environment based on player behavior, ensuring a more personalized and enjoyable experience. This method can easily be extended to adjusting difficulty levels, providing real-time feedback that maintains an optimal challenge for players.

4.3 Optimizing Rewards and Incentive Structures

Another key area where MAB algorithms can be applied is in optimizing the reward system of a game. Rewards play a significant role in maintaining player motivation and engagement. Poorly designed reward structures can lead to disengagement, while well-calibrated rewards can drive player retention and satisfaction.

MAB can be used to dynamically adjust rewards based on player interaction. For instance, rewards such as in-game items, achievements, or experience points can be treated as different arms of a bandit, with the algorithm continuously learning which rewards best keep players engaged. By balancing exploration and exploitation, the MAB can test different reward configurations and identify those that maximize player satisfaction.

In Vinogradov and Harrison's study on dynamic player modeling, MAB algorithms were employed to adapt reward structures based on how players interacted with the game environment 3. This approach ensures that rewards are tailored to individual players, offering a more personalized experience. For example, high-performing players may receive larger, more challenging rewards, while players who are struggling might be given smaller, more frequent rewards to keep them motivated.

4.4 Adapting User Interface (UI) for Player Experience

The user interface (UI) is a critical element of game design, affecting how players interact with the game and their overall experience. A well-designed UI can enhance immersion and usability, while a poorly designed UI can detract from the gaming experience. MAB algorithms can optimize UI elements by continuously adjusting layouts, control schemes, or visual feedback based on player preferences and performance.

MAB algorithms can be used to test and adapt UI configurations in real time. For example, different UI layouts or control schemes can be treated as arms, with the algorithm learning which options provide the best player experience. Players who prefer a streamlined, minimalistic interface can be presented with fewer on-screen elements, while those who prefer more information or control can be shown more complex UIs. This dynamic adaptation allows the game to cater to a wide range of player preferences without requiring manual intervention.

Amiri and Sekhavat's work also highlights how real-time adjustments can be made to game environments based on player interaction [9]. Similarly, this approach can be applied to UI elements, where MAB algorithms can optimize the layout and control systems in real-time based on how players interact with them. For instance, if players struggle with a certain UI configuration, the MAB can switch to an alternative layout that is easier to navigate.

4.5 Balancing Exploration and Exploitation in Game Design

One of the core challenges in applying MAB algorithms to game design is balancing exploration (trying new game design choices) and exploitation (utilizing known successful designs). This trade-off is critical to ensuring that the game continues to evolve and improve while maintaining a high-quality experience for players.

During exploration, the MAB algorithm tries different configurations of game design elements—such as difficulty, rewards, and UI—to learn which ones work best for each player. This is particularly important for new players or players whose preferences are not yet well understood.

In the exploitation phase, the algorithm selects game configurations that have proven successful in the past. For instance, if a particular reward system keeps the player engaged, the MAB will continue using that configuration until a change in player behavior suggests that it is no longer optimal.

To achieve an effective balance, methods like epsilon-greedy, Upper Confidence Bound (UCB), or Thompson Sampling can be used. These algorithms allow the game to experiment with new designs while prioritizing configurations that have already shown success.

Thompson Sampling for Game Design: Kunanusont et al. introduced the N-Tuple Bandit Evolutionary Algorithm [10], which uses a bandit-based approach to optimize game design parameters. This algorithm balances exploration and exploitation by using an evolutionary framework to evolve game parameters dynamically. The system continuously evaluates new configurations and refines the design based on player feedback, providing a robust mechanism for real-time game optimization.

4.6 Challenges and Limitations

While MAB algorithms offer powerful tools for game design optimization, there are several challenges to their implementation:

Cold-start Problem: Similar to player modeling, the cold-start problem arises when there is insufficient initial data to make informed decisions. In game design, this means that MAB algorithms may struggle to optimize design elements early in a player's experience, as they lack historical data on player preferences.

Scalability: As games grow in complexity, the number of design elements (arms) that need to be optimized increases. Managing and optimizing a large number of arms can become computationally expensive, requiring efficient algorithms and careful tuning to ensure that the game adapts in real time without performance issues.

Player Fatigue: Frequent changes in game design elements (such as UI or difficulty) can potentially lead to player fatigue if not managed carefully. MAB algorithms must strike a balance between making necessary adjustments and avoiding overwhelming players with too many changes too quickly.

5 Conclusion

In conclusion, MAB (multi-armed bandit) algorithms provide a robust framework for real-time modeling of player characteristics. By adapting to individual player behavior, MAB enhances player engagement, satisfaction, and the overall gaming experience. These algorithms can be applied not only to model players but also to optimize the game design itself, offering a flexible, dynamic approach to game design. This enables real-time adjustments to elements like difficulty levels, rewards, and the user interface based on player feedback. By continuously learning from player behavior, MAB algorithms create a personalized and evolving experience that keeps pace with the player. To advance the application of MAB algorithms in game design, the next steps for research and implementation involve addressing the cold-start problem, improving scalability, and refining techniques to ensure seamless player experience throughout game evolution.

The cold-start problem occurs when MAB algorithms lack sufficient initial data to make effective decisions, particularly in gaming where new players or content are introduced without enough behavioral information for individualized experiences. If the algorithm fails to adapt quickly, it can result in a suboptimal first impression, diminishing player engagement and retention. Solutions to this challenge include using historical data from similar players to predict new players' preferences and incorporating random exploration early on to gather diverse data across game elements like difficulty and rewards. As player numbers and game content grow, the complexity and computational demands on MAB algorithms increase, especially in large-scale multiplayer or open-world games that require rapid responses to varied player actions. To improve scalability, techniques such as parallel computing, hierarchical structuring of game elements, and clustering algorithms for grouping players with similar behaviors can be employed. Furthermore, maintaining a consistent player experience as games evolve is crucial; if the algorithm cannot promptly adjust to changes in player behavior or new content, abrupt gameplay shifts may occur, disrupting immersion. Implementing incremental updates allows real-time monitoring of player behavior and gradual adjustments to difficulty and rewards, while hybrid algorithms can balance short-term responsiveness with long-term consistency. Additionally, integrating feedback control mechanisms enables automatic adjustments to game parameters, ensuring a stable player experience amid performance fluctuations.

MAB algorithms offer a flexible, adaptive framework for player modeling and game optimization in real-time. By addressing challenges such as the cold-start problem, scalability, and ensuring a seamless experience, MAB algorithms can enable more personalized, dynamic, and engaging gaming experiences. Future research will focus on refining these aspects to ensure that MAB-driven gameplay remains consistent, enjoyable, and responsive as the game evolves.

References

1. Gray, Robert C., Jichen Zhu, and Santiago Ontañón. "Multiplayer modeling via multi-armed bandits." 2021 IEEE Conference on Games (CoG). IEEE, 2021.
2. Gray, Robert C., et al. "Player modeling via multi-armed bandits." Proceedings of the 15th international conference on the foundations of digital games. 2020.

3. Vinogradov, Anton, and Brent Harrison. "Using multi-armed bandits to dynamically update player models in an experience managed environment." Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. Vol. 18. No. 1. 2022.
4. Rao, Dattaraj. "Contextual Bandits for adapting to changing User preferences over time." arXiv preprint arXiv:2009.10073 (2020).
5. Raharjo, Kenny, and Ramon Lawrence. "Using multi-arm bandits to optimize game play metrics and effective game design." (2016).
6. Kunanusont, Kamolwan, Simon Lucas, and Diego Perez-Liebana. "Modeling player experience with the n-tuple bandit evolutionary algorithm." Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. Vol. 14. No. 1. 2018.
7. Wang, Zhi, et al. "Stochastic multi-player bandit learning from player-dependent feedback." ICML Workshop on Real World Experiment Design and Active Learning. 2020.
8. Raharjo, Kenny. Applying Multi-Armed Bandit on Game Development. Diss. PhD thesis, PhD thesis, University of British Columbia, Okanagan, 2016.
9. Amiri, Zahra, and Yoones A. Sekhavat. "Intelligent adjustment of game properties at run time using multi-armed bandits." The Computer Games Journal 8.3 (2019): 143-156.
10. Kunanusont, Kamolwan, et al. "The n-tuple bandit evolutionary algorithm for automatic game improvement." 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2017.