

Self-Balanced Cart Optimization Algorithm

Jincheng Zhang^{*1}[0009-0005-1860-0009]

¹ Faculty of Science and Technology, Rajabhat Maha Sarakham University, Maha Sarakham
44000, Thailand
zjc1639834588@gmail.com

Abstract. In e-commerce systems, the shopping cart is a concentrated reflection of user decision-making behavior. Its dynamic characteristics reflect the complex relationship between multidimensional information interaction, psychological balance, and economic constraints. Traditional shopping cart optimization models often focus on static combinations or price discount maximization, failing to fully reflect the dynamic equilibrium characteristics of shopping behavior. This paper proposes a novel algorithm based on shopping behavior characteristics: the Self-Balanced Cart Optimization (SBCO) algorithm driven by shopping cart interaction. This algorithm, based on the "shopping force field" and "potential energy balance" mechanisms, remodels the dynamic evolution of the shopping cart from the perspective of a physical system. By defining shopping force, shopping potential energy functions, and interaction forces between items, the algorithm achieves adaptive optimization and stable convergence of the shopping cart state. This paper systematically derives the complete theoretical structure of the algorithm from a mathematical perspective, establishing analyzable dynamic equations, potential energy functions, and optimization constraints. Results demonstrate that the algorithm can theoretically and naturally describe the self-organizing characteristics of shopping systems, providing a new algorithmic foundation for intelligent decision-making and dynamic recommendation in e-commerce.

Keywords: Shopping cart optimization; heuristic algorithm; potential energy model; shopping force field; self-balancing mechanism.

I. Introduction

In modern e-commerce platforms, the shopping cart is a key node connecting consumers and the product system. The operations users perform in the shopping cart, such as adding, deleting, and modifying quantities, are actually a dynamic optimization process based on psychological trade-offs and economic constraints. Traditional optimization algorithms, such as greedy strategies, knapsack models, and dynamic programming, all assume that the system is static and rational, but real-world user decisions are often nonlinear, volatile, and feedback-based. Shopping behavior is not only affected by price and discounts, but also by inventory, time, preferences, and the relationship between products. Therefore, building an optimization algorithm that can dynamically respond to changes in multiple factors and automatically reach a

state of equilibrium has become an important research direction for intelligent e-commerce algorithms [1-46].

Based on the above motivation, this paper proposes an optimization model with the concept of "shopping force field" as its core. Its concept is derived from the principle of force field and potential energy balance in physics: in a system, multiple forces act on each object, and the system continuously adjusts its state to reach the equilibrium point with the minimum potential energy. This paper introduces this concept into shopping cart optimization, treating each product as a force-bearing body subject to the combined effects of attraction and repulsion. The shopping cart state is optimal when the total potential energy of the system is minimized.

The innovation of this paper lies in: by introducing mathematical definitions such as shopping force, interaction force, and cart potential energy, a self-organizing and self-regulating optimization framework is constructed. The algorithm does not require preset fixed weights, but instead automatically adjusts system variables through a dynamic update mechanism, allowing the shopping cart to naturally evolve to the optimal equilibrium state.

II. Algorithm Modeling Concept

Shopping behavior is a multi-factor interaction process. The purchase quantity of each item is influenced by price, discounts, user preferences, inventory, and other items. If the set S represents all candidate items, the shopping cart state can be represented as a vector $Q = \{Q_{\text{subscript}1}, Q_{\text{subscript}2}, \dots, Q_{\text{subscript}n}\}$, where $Q_{\text{subscript}i}$ represents the purchase quantity of item i .

The goal of a shopping system is to find the optimal combination Q that maximizes the user's overall satisfaction or utility while satisfying budget and inventory constraints. However, in reality, users do not make choices through precise calculations, but are driven by a series of psychological "forces" that lead to continuous adjustments and trade-offs. Therefore, the algorithm proposed in this paper is centered on the "Shopping Force Field Model," arguing that the shopping cart optimization process can be viewed as a self-balancing process within a physical system.

In this model, each item (s , subscript i) is subject to the combined forces of multiple factors. The ultimate goal of the system is to achieve an equilibrium state for the "shopping force" of all items through multiple rounds of dynamic adjustments, thereby achieving overall optimization.

III. Mathematical Definition of the Shopping Force Field

3.1 Definition of Shopping Force

Shopping force, denoted as F_i , represents the degree to which item i attracts or repels purchase decisions. It combines four dimensions: user preference, price, discount, and inventory, and is defined as follows:

$$F_i = \alpha \times r_{ui} - \beta \times (P_i / B) + \gamma \times f_i(Q_i) - \delta \times (Q_i / S_i)$$

Where:

α , β , γ , and δ are weighting factors for interest, price, discount, and inventory, respectively, satisfying $\alpha + \beta + \gamma + \delta = 1$.

r_{ui} represents the user's interest score for item i (between 0 and 1).

P_i represents the item price.

B represents the budget.

$f_i(Q_i)$ represents the item discount function, which is related to the quantity Q_i .

S_i represents the inventory quantity.

The four effects correspond to:

$\alpha \times r_{ui}$: attraction due to user interest;

$-\beta \times (P_i / B)$: economic repulsion due to price;

$\gamma \times f_i(Q_i)$: The purchasing appeal of the discount;

$-\delta \times (Q_i / S_i)$: The inhibitory effect of inventory shortages.

3.2 Potential Energy Function Definition

The overall state of the shopping cart can be represented by the potential energy function $U(C)$, which has the form:

$$U(C) = \sum (1/2 \times k \times (F_i - F_{target})^2)$$

Where:

k is the system regulation constant, used to control the degree of potential energy smoothing;

F_{target} represents the ideal purchasing power (i.e., the user's psychological equilibrium point, generally set to 0);

When the system reaches equilibrium, $F(\text{subscript } i)$ approaches F_{target} , minimizing $U(C)$.

The mathematical goal of shopping cart optimization is:

Minimize $U(C)$

Constraints: $\sum (P_{\text{subscript } i} \times Q_{\text{subscript } i}) \leq B$ and $0 \leq Q_{\text{subscript } i} \leq \text{Stock}_{\text{subscript } i}$

3.3 Dynamic Equilibrium Principle

The shopping cart optimization process is not a one-time solution, but a continuous, dynamic process.

Define the state of the shopping cart at time step t as $Q_{\text{subscript } i}(t)$, and its change as $\Delta Q_{\text{subscript } i}$. The update of the item quantity follows the following dynamic equation:

$$\Delta Q_{\text{subscript } i} = \lambda \times F_{\text{subscript } i} - \theta \times \partial U(C) / \partial Q_{\text{subscript } i}$$

Where:

λ is the shopping momentum coefficient, representing the strength of the user's decision response;

θ is the potential energy damping coefficient, used to balance system oscillations;

$\partial U(C) / \partial Q_{\text{subscript } i}$ is the partial derivative of the potential energy with respect to the item quantity.

From the definition of $U(C)$, we obtain:

$$\partial U(C) / \partial Q_{\text{subscript } i} = k \times (F_{\text{subscript } i} - F_{\text{target}}) \times (\partial F_{\text{subscript } i} / \partial Q_{\text{subscript } i})$$

Since $F_{\text{subscript } i}$ contains $f_{\text{subscript } i}(Q_{\text{subscript } i})$, its partial derivative is:

$$\partial F_{\text{subscript } i} / \partial Q_{\text{subscript } i} = \gamma \times \partial f_{\text{subscript } i}(Q_{\text{subscript } i}) / \partial Q_{\text{subscript } i} - \delta / \text{Stock}_{\text{subscript } i}$$

Thus, the complete dynamic update equation is:

$$\Delta Q_{\text{subscript } i} = \lambda \times F_{\text{subscript } i} - \theta \times k \times (F_{\text{subscript } i} - F_{\text{target}}) \times (\gamma \times \partial f_{\text{subscript } i}(Q_{\text{subscript } i}) / \partial Q_{\text{subscript } i} - \delta / \text{Stock}_{\text{subscript } i})$$

After the update, execute:

$Q_{\text{subscript } i} \leftarrow Q_{\text{subscript } i} + \Delta Q_{\text{subscript } i}$
 If $Q_{\text{subscript } i} < 0$, set 0
 If $Q_{\text{subscript } i} > \text{Stock}_{\text{subscript } i}$, then set it to $\text{Stock}_{\text{subscript } i}$.

IV. Inter-item Interaction Mechanism

In a shopping cart system, items often complement or substitute with each other. For example, buying a certain brand of mobile phone may increase the probability of purchasing a phone case, while buying a certain food may reduce the choice of substitutes. Therefore, the purchasing power of a single item is not isolated but is influenced by other items.

Define the inter-item interaction force $G_{\text{subscript } ij}$ to describe the association effect between items $s_{\text{subscript } i}$ and $s_{\text{subscript } j}$:

$$G_{\text{subscript } ij} = \rho \times S_{\text{subscript } ij} \times (F_{\text{subscript } i} - F_{\text{subscript } j})$$

Where:

ρ is the interaction influence coefficient;

$S_{\text{subscript } ij}$ represents the semantic or purchase association between items (negative for complementarity and positive for substitution).

The total potential energy function of the system expands to:

$$U(C) = \sum (1/2 \times k \times (F_{\text{subscript } i} - F_{\text{target}})^2) + \sum \sum (1/2 \times \rho \times S_{\text{subscript } ij} \times (F_{\text{subscript } i} - F_{\text{subscript } j})^2)$$

The optimization objective remains to minimize $U(C)$.

At this point, each item in the system is no longer independent but interacts with other items in a "shopping force field." This interaction force forces the items to collectively adjust toward lower potential energy, thereby achieving self-organization and self-balancing of the shopping cart as a whole.

V. Algorithm Iteration Mechanism

The shopping cart system performs the following operations at each time step:

Calculate the shopping force $F_{\text{subscript } i}$:

Using the above definition, calculate the shopping force of each item based on the current $Q_{\text{subscript } i}$.

Calculate the interaction force $G_{\text{subscript } ij}$:

For each pair of items i and j , calculate the interaction term between them.

Update quantity $Q_{\text{subscript } i}$:

According to the dynamic equation

$$Q_{\text{subscript } i} \leftarrow Q_{\text{subscript } i} + \lambda \times F_{\text{subscript } i} - \theta \times k \times (F_{\text{subscript } i} - F_{\text{target}}) \times (\gamma \times \partial f_{\text{subscript } i} / \partial Q_{\text{subscript } i} - \delta / \text{Stock}_{\text{subscript } i}) + \sum G_{\text{subscript } ij}$$

Constraint modification:

If $\sum (P_{\text{subscript } i} \times Q_{\text{subscript } i}) > B$, prioritize reducing the quantity of the item with the smallest $|F_{\text{subscript } i}|$ until the budget is met.

Convergence determination:

If the absolute value of all $\Delta Q_{\text{subscript } i}$ values is less than the threshold ε , the algorithm terminates and the shopping cart reaches a self-balancing state.

VI. Self-regulation Mechanism

During user interaction, the shopping power weight parameters α , β , γ , and δ are not fixed but automatically adjust based on user behavior.

For example, if a user frequently increases the quantity of a certain category of items, the system automatically increases the interest weight α for that category; when inventory is low, the system increases δ to enhance the suppression effect.

The weight update mechanism is defined as:

$$\alpha \leftarrow \alpha + \eta \times (\text{Preference Contribution} - \alpha)$$

$$\beta \leftarrow \beta + \eta \times (\text{Price Contribution} - \beta)$$

$$\gamma \leftarrow \gamma + \eta \times (\text{Discount Contribution} - \gamma)$$

$$\delta \leftarrow \delta + \eta \times (\text{Inventory Contribution} - \delta)$$

where η is the learning rate, which controls the adjustment amplitude.

Through this mechanism, the system can continuously self-learn during user interaction, making the purchasing power model more closely aligned with the user's true psychological dynamics.

VII. Algorithm Convergence Analysis

From a mathematical perspective, the algorithm's iterative process is equivalent to gradient descent in a multidimensional potential energy space.

The potential energy function $U(C)$ is squared with respect to F (subscript i), and the weight parameters k and ρ are both positive. Therefore, $U(C)$ is always non-negative and has a global minimum.

Each update to Q (subscript i) either decreases or remains constant, so the algorithm theoretically converges to a stable equilibrium point.

Furthermore, if the values of θ and λ satisfy the following conditions:

$$0 < \theta < 1$$

$$0 < \lambda < 1$$

then the system update can be viewed as a damped oscillation process, eventually approaching an equilibrium point exponentially.

This convergence is consistent with the psychological model of shopping behavior: after repeated trade-offs, users eventually reach a state of satisfaction and stop frequently modifying their shopping carts.

VIII. Algorithm Significance and Application Prospects

This algorithm provides a new modeling paradigm:

Converting shopping cart behavior into a "force field equilibrium system." This perspective not only describes users' immediate choices but also simulates their psychological dynamics, impulsivity, and stability. Through the interplay of shopping force and interaction force, the system can automatically identify incompatible combinations of items in the shopping cart and gradually optimize to an equilibrium state.

This algorithm has a wide range of applications:

Shopping recommendation systems on e-commerce platforms can be used to optimize recommendation results;

Dynamic discount strategy adjustment systems can be used to predict shopping cart trends;

User behavior modeling can be used to analyze shopping habits and decision-making patterns;

Interactive intelligent marketing can be used to predict users' psychological critical points.

Because this algorithm is based on dynamic mechanics, does not rely on historical sample training, and can adaptively optimize even with incomplete data, it is of great value in real-time e-commerce decision-making, recommendation, and user modeling.

IX. Mathematical Summary

(1) Definition of shopping power:

$$F_{\text{subscript}i} = \alpha \times r_{\text{subscript}i} - \beta \times (P_{\text{subscript}i} / B) + \gamma \times f_{\text{subscript}i}(Q_{\text{subscript}i}) - \delta \times (Q_{\text{subscript}i} / \text{Stock}_{\text{subscript}i})$$

(2) Interaction force between commodities:

$$G_{\text{subscript}ij} = \rho \times S_{\text{subscript}ij} \times (F_{\text{subscript}i} - F_{\text{subscript}j})$$

(3) Total potential energy function:

$$U(C) = \sum (1/2 \times k \times (F_{\text{subscript}i} - F_{\text{target}})^2) + \sum \sum (1/2 \times \rho \times S_{\text{subscript}ij} \times (F_{\text{subscript}i} - F_{\text{subscript}j})^2)$$

(4) Dynamic update equation:

$$\Delta Q_{\text{subscript}i} = \lambda \times F_{\text{subscript}i} - \theta \times k \times (F_{\text{subscript}i} - F_{\text{target}}) \times (\gamma \times \partial f_{\text{subscript}i}(Q_{\text{subscript}i}) / \partial Q_{\text{subscript}i} - \delta / \text{Stock}_{\text{subscript}i}) + \sum G_{\text{subscript}ij}$$

(5) Quantity update:

$$Q_{\text{subscript}i} \leftarrow Q_{\text{subscript}i} + \Delta Q_{\text{subscript}i}$$

And make budget constraint corrections.

X. Conclusion

The shopping cart interaction-driven self-balancing optimization algorithm proposed in this paper transforms the traditional discrete decision-making problem into a continuous force field optimization problem.

By introducing the mathematical modeling of shopping power, interaction force and potential energy function, the algorithm can naturally explain the dynamic characteristics of user shopping behavior based on the principle of physical balance. Its core idea is to let the shopping cart evolve automatically under the influence of multiple factors until the system potential energy reaches the minimum and a stable optimal configuration is obtained.

This method breaks through the limitations of static solution and fixed weight in traditional optimization algorithms, and enables the shopping system to have the ability of self-organization, self-learning and self-regulation.

In the fields of e-commerce, recommendation systems and behavioral economics, this algorithm provides new theoretical tools and computational frameworks for the modeling and optimization of intelligent shopping behavior.

References

1. Çam, Z. G., Çimen, S., & Yıldırım, T. (2015, January). Learning parameter optimization of multi-layer perceptron using artificial bee colony, genetic algorithm and particle swarm optimization. In 2015 IEEE 13th International Symposium on Applied Machine Intelligence and Informatics (SAMI) (pp. 329-332). IEEE.
2. Askarzadeh, A., & dos Santos Coelho, L. (2015). Determination of photovoltaic modules parameters at different operating conditions using a novel bird mating optimizer approach. *Energy Conversion and Management*, 89, 608-614.
3. Wu, J., & Frazier, P. (2016). The parallel knowledge gradient method for batch Bayesian optimization. *Advances in neural information processing systems*, 29.
4. Guedria, N. B. (2016). Improved accelerated PSO algorithm for mechanical engineering optimization problems. *Applied Soft Computing*, 40, 455-467.
5. Moosavi, S. H. S., & Bardsiri, V. K. (2017). Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation. *Engineering Applications of Artificial Intelligence*, 60, 1-15.
6. Dhiman, G., & Kumar, V. (2018). Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. *Knowledge-Based Systems*, 150, 175-197.
7. Kaveh, A., & Ghazaan, M. I. (2017). Enhanced whale optimization algorithm for sizing optimization of skeletal structures. *Mechanics Based design of structures and Machines*, 45(3), 345-362.
8. Khalil, E., Dai, H., Zhang, Y., Dilkina, B., & Song, L. (2017). Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30.
9. Yıldız, B. S., Mehta, P., Panagant, N., Mirjalili, S., & Yıldız, A. R. (2022). A novel chaotic Runge Kutta optimization algorithm for solving constrained engineering problems. *Journal of Computational Design and Engineering*, 9(6), 2452-2465.
10. Seyyedabbasi, A. (2022). WOASCALF: A new hybrid whale optimization algorithm based on sine cosine algorithm and levy flight to solve global optimization problems. *Advances in Engineering Software*, 173, 103272.
11. Marini, F., & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. *Chemometrics and intelligent laboratory systems*, 149, 153-165.
12. Posa, M., Kuindersma, S., & Tedrake, R. (2016, May). Optimization and stabilization of trajectories for constrained dynamical systems. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1366-1373). IEEE.
13. Zhao, T., Wang, Z., & Liu, H. (2015). A nonconvex optimization framework for low rank matrix estimation. *Advances in Neural Information Processing Systems*, 28.
14. Akdemir, D., Sanchez, J. I., & Jannink, J. L. (2015). Optimization of genomic selection training populations with a genetic algorithm. *Genetics Selection Evolution*, 47(1), 38.
15. Park, J., & Law, K. H. (2015). Layout optimization for maximizing wind farm power production using sequential convex programming. *Applied energy*, 151, 320-334.
16. Nie, F., Wang, X., Jordan, M., & Huang, H. (2016, March). The constrained laplacian rank algorithm for graph-based clustering. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 30, No. 1).

17. Song, J., Wang, J., & Lu, H. (2018). A novel combined model based on advanced optimization algorithm for short-term wind speed forecasting. *Applied energy*, 215, 643-658.
18. Nazari-Heris, M., Mohammadi-Ivatloo, B., & Gharehpetian, G. B. (2018). A comprehensive review of heuristic optimization algorithms for optimal combined heat and power dispatch from economic and environmental perspectives. *Renewable and Sustainable Energy Reviews*, 81, 2128-2143.
19. Abualigah, L. M., & Khader, A. T. (2017). Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. *The Journal of Supercomputing*, 73(11), 4773-4795.
20. Yurek, E. E., & Ozmutlu, H. C. (2018). A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies*, 91, 249-262.
21. Titri, S., Larbes, C., Toumi, K. Y., & Benatchba, K. (2017). A new MPPT controller based on the Ant colony optimization algorithm for Photovoltaic systems under partial shading conditions. *Applied Soft Computing*, 58, 465-479.
22. Zhang, J. Unification-Inspired Optimization Algorithm: A Mathematical Framework Based on the First Egyptian Dynasty. <https://doi.org/10.13140/RG.2.2.36606.04164>
23. Zhang, J. Lunar Halo Optimization. <https://doi.org/10.13140/RG.2.2.17897.15206>
24. Kumar, Y., & Sahoo, G. (2017). An improved cat swarm optimization algorithm based on opposition-based learning and Cauchy operator for clustering. *Journal of Information Processing Systems*, 13(4), 1000-1013.
25. Zamfirache, I. A., Precup, R. E., Roman, R. C., & Petriu, E. M. (2022). Policy iteration reinforcement learning-based control using a grey wolf optimizer algorithm. *Information Sciences*, 585, 162-175.
26. Izci, D., Ekinci, S., & Hussien, A. G. (2024). Efficient parameter extraction of photovoltaic models with a novel enhanced prairie dog optimization algorithm. *Scientific Reports*, 14(1), 7945.
27. Garg, H. (2016). A hybrid PSO-GA algorithm for constrained optimization problems. *Applied Mathematics and Computation*, 274, 292-305.
28. Montes, M. O., Ivvan, S., Pe, V., & Rionda, S. B. (2016). Topology optimization algorithms for the solution of compliance and volume problems in 2D. *Topology*.
29. Bamakan, S. M. H., Wang, H., Yingjie, T., & Shi, Y. (2016). An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. *Neurocomputing*, 199, 90-102.
30. Lacoste-Julien, S., & Jaggi, M. (2015). On the global linear convergence of Frank-Wolfe optimization variants. *Advances in neural information processing systems*, 28.
31. Pradhan, M., Roy, P. K., & Pal, T. (2016). Grey wolf optimization applied to economic load dispatch problems. *International Journal of Electrical Power & Energy Systems*, 83, 325-334.
32. Oliva, D., Abd El Aziz, M., & Hassanien, A. E. (2017). Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. *Applied energy*, 200, 141-154.
33. Mousavirad, S. J., & Ebrahimpour-Komleh, H. (2017). Human mental search: a new population-based metaheuristic optimization algorithm. *Applied Intelligence*, 47(3), 850-887.
34. Zhang, X., Miao, Q., Zhang, H., & Wang, L. (2018). A parameter-adaptive VMD method based on grasshopper optimization algorithm to analyze vibration signals from rotating machinery. *Mechanical Systems and Signal Processing*, 108, 58-72.

35. Abd El Aziz, M., Ewees, A. A., & Hassanien, A. E. (2017). Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation. *Expert Systems with Applications*, 83, 242-256.
36. Chawla, V. K., Chanda, A. K., & Angra, S. (2018). Multi-load AGVs scheduling by application of modified memetic particle swarm optimization algorithm. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 40(9), 436.
37. Gholizadeh, S., Davoudi, H., & Fattahi, F. (2017). Design of steel frames by an enhanced moth-flame optimization algorithm. *Steel Compos Struct*, 24(1), 129-140.
38. Dhiman, G., & Kumar, V. (2018). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowledge-Based Systems*, 159, 20-50.
39. Bansal, J. C. (2018). Particle swarm optimization. In *Evolutionary and swarm intelligence algorithms* (pp. 11-23). Cham: Springer International Publishing.
40. Marzband, M., Yousefnejad, E., Sumper, A., & Domínguez-García, J. L. (2016). Real time experimental implementation of optimum energy management system in standalone microgrid by using multi-layer ant colony optimization. *International Journal of Electrical Power & Energy Systems*, 75, 265-274.
41. Zhang, J. Asteroid Satellite Inspired Optimization. <https://doi.org/10.13140/RG.2.2.28471.38562>
42. Zhang, J. Variable Star Light Curve Optimization. <https://doi.org/10.13140/RG.2.2.14250.07368>
43. Xu, G., Yang, Y. Q., Liu, B. B., Xu, Y. H., & Wu, A. J. (2015). An efficient hybrid multi-objective particle swarm optimization with a multi-objective dichotomy line search. *Journal of computational and applied mathematics*, 280, 310-326.
44. Mo, H., & Xu, L. (2015). Research of biogeography particle swarm optimization for robot path planning. *Neurocomputing*, 148, 91-99.
45. Darwish, A. (2018). Bio-inspired computing: Algorithms review, deep analysis, and the scope of applications. *Future Computing and Informatics Journal*, 3(2), 231-246.
46. Sulaiman, N., Hannan, M. A., Mohamed, A., Ker, P. J., Majlan, E. H., & Daud, W. W. (2018). Optimization of energy management system for fuel-cell hybrid electric vehicles: Issues and recommendations. *Applied energy*, 228, 2061-2079.