Yugoslav Journal of Operations Research
xx (20xx), Number xx, xxx–xxx
DOI: https://doi.org/10.2298/YJOR240815030S

Research Article

COMPLETE LATTICE USING LEXICOGRAPHICAL ORDER AND ITS APPLICATION IN NON-COOPERATIVE ORDERED GAME

Rubono SETIAWAN*
Mathematics Education Department, Universitas Sebelas Maret
rubono.matematika@staff.uns.ac.id, ORCID:0000-0001-7049-5764

Nughthoh Arfawi KURDHI
Mathematics Department, Universitas Sebelas Maret
arfa@staff.uns.ac.id, ORCID:0000-0001-9274-1807

Received: August 2024 / Accepted: July 2025

Abstract: One application of lattices in optimization is defining the equilibrium set of ordered games, typically using the usual (coordinate-wise) order, which is incomplete in $\mathbb{R}^n$. This incompleteness makes some strategies incomparable, requiring a special game concept. Using a complete order, like the lexicographic order, results in a complete lattice. This study explores the properties of a complete lattice with lexicographic order for non-cooperative games and provides a Python algorithm to determine the Nash equilibrium of a supermodular game.

Keywords: Lattice, lexicographical order, equilibrium set, non-cooperative game.

MSC: 06B20, 91A10, 91A26, 90-04.

1. INTRODUCTION

Game theory has been used in optimization analysis across various fields, primarily in economics and computing. The concept of Nash equilibrium has also been applied in modeling involving computational processes and computer science. Several related results have been provided by [1], [2], [3], [4], [5], [6], and [7].

*Corresponding author

A lattice is a partially ordered set equipped with two binary operations, join and meet. Lattice is an algebraic topic. In the literature on the development of lattices, the commonly used order is the standard order in $\mathbb{R}^n$ The coordinate-wise order is an incomplete order. In some applications, this results in pairs of strategies whose preferences cannot be compared. Besides incomplete orders, there are also complete orders. A lattice equipped with a complete order becomes a complete lattice, having properties different from lattices with incomplete orders. One complete order on a lattice, which is a subset of $\mathbb{R}^n$, is the lexicographic order. With the use of a complete order, in this case, the lexicographic order, every element in the lattice can be uniquely and consistently ordered, with no elements left incomparable. Lattices have been used as a discussion domain in applications in many fields of economics and industry, particularly in game theory, as seen in [8], [9], [10], [11], [12], and [13]. However, the use of lexicographic order has not been thoroughly explained. Yet, using a lattice with a complete order as a domain would enable a more comprehensive analysis of all strategies or preferences employed by players or actors in the fields of economics and industry.

One application of lattices is lattice-based optimization, which can be clearly illustrated in the determination of Nash equilibrium from ordered non-cooperative games. In ordered games, the order of strategies used by each player is considered as the basis of each player's preferences. One specific condition that arises due to the order in strategies is complementary strategies. The set of strategies and the set of Nash equilibria for games with complementary strategies are generally defined in a lattice. Examples of games with complementary strategies include supermodular and submodular games. These games were first described by Topkis [8] and have since been applied in the fields of economics [9], [14], [11], [15], [16] and industry ([10], [12], and [13]. In the literature related to supermodular games, the specific impact of using a complete lattice compared to using an arbitrary lattice (which is generally incomplete) has not been thoroughly discussed. A complete lattice is built with a complete order, such as the lexicographic order. In lattices with incomplete order, there is often the problem that some strategies cannot be compared. Although this issue does not occur in complementary games, using a complete order will certainly allow the discovery of new properties of the strategy sets and their Nash equilibrium sets. Based on the explanation above, the research questions (RQ) are as follows:

1. (RQ1) What are the properties of lattices with lexicographic order related to the use of lattices in the strategy sets of ordered non-cooperative games ?
2. (RQ2) What are the equilibrium properties of ordered non-cooperative games whose Nash equilibrium is defined on lattices with lexicographic order?

After formulating the research questions, the objectives of this research can be determined: to formulate the properties of lattices with lexicographic order and then to formulate the existence of equilibrium sets of ordered non-cooperative games whose Nash equilibrium is defined on lattices with lexicographic order. The rest of this paper is organized as follows. The research methodology is detailed in Chapter 2. The explanation of the properties of complete lattices with lexicographic order and the theorems explaining the Nash equilibrium of ordered non-cooperative games are presented in Chapter 3, which covers results and discussion. Finally, the research conclusions are presented in Chapter 4, Conclusion.

2. METHODS

This research is theoretical research, thus the methodology employed is mathematical theoretical research through literature review and concept development. The first step is to study supporting references, followed by determining a research theme that has not been previously discussed and has novelty value. Then the researcher conducts theoretical analysis to obtain mathematical properties in the form of definitions and theorems. The necessary information includes references related to orders, lattices, and ordered non-cooperative games. The detailed theoretical steps used are as follows:

1. Study references related to orders, partially ordered sets, lattices, ordered games, and Nash equilibrium.
2. Conduct literature review and conclude that the concept of lattice with a complete order, in this case, the lexicographic order, has not yet been explained in terms of its properties. Furthermore, this complete lattice can be used for advanced properties of Nash equilibrium for ordered non-cooperative games.
3. Explain the properties of lattices with lexicographic order related to the use of lattices in the strategy sets of ordered non-cooperative games ?
4. Explain the theorems related to Nash equilibrium in ordered non-cooperative games defined on lattices with lexicographic order.

3. Lattice with Lexicographical Order and Its Application to Supermodular Game

A lattice (with coordinatewise order) is a partially ordered set $(S, \leq)$ where every pair of elements has a unique supremum (least upper bound) and infimum (greatest lower bound). Mathematically, for any two elements x and y in S, the supremum $x \vee y$ is the least element z such that $x \leq z$ and $y \leq z$, while the infimum $x \wedge y$ is the greatest element w such that $w \leq x$ and $w \leq y$. This structure ensures that we can always find a common upper and lower bound for any pair of elements in the set. In addition to the standard order we know in $\mathbb{R}$, which is the coordinate-wise order $\leq$, there are many other order definitions used in various algebraic topics and applications. One of these is the lexicographic order.

In this study, we focus on the space $\mathbb{R}^n$. The choice of $\mathbb{R}^n$ is based on its geometrically simple structure and the ease of ordering its elements. This is important because strategies in non-cooperative games often represent real-world objects, making their illustration and ordering more intuitive when the domain is $\mathbb{R}^n$. Furthermore, the theory of supermodular games, which has been extensively developed by [11], also works well in $\mathbb{R}^n$, where the elements of this space can be fully ordered using the lexicographical order.

Topologically, the space $\mathbb{R}^n$ is a Hausdorff space, meaning that any two distinct points in this space can be separated by disjoint open neighborhoods. This structure ensures that any subspace of $\mathbb{R}^n$ maintains the properties of separability and orderability, which are crucial in the analysis of supermodular games. Suppose S_i is a subset of $\mathbb{R}^n$, then the strategy set $S = S_1 \times S_2 \times \cdots \times S_n$, where each S_i follows a lexicographical order, will also retain a consistent lexicographical ordering.

Moreover, a lattice-based approach in $\mathbb{R}^n$ is sufficient to guarantee the properties of monotonicity and supermodularity in the analyzed game. The lattice structure in this space meets the requirements for analyzing players' strategies without the need for additional, more complex topological structures. Therefore, selecting $\mathbb{R}^n$ as the domain in this study is considered appropriate for the needs of supermodular games.

The lexicographic order, which denoted as $\leq_{lex}$ is a way of ordering pairs (or tuples) of elements. Given two pairs (x_1, y_1) and (x_2, y_2), we say that (x_1, y_1) is less than (x_2, y_2) if $x_1 < x_2$, or if $x_1 = x_2$ and $y_1 < y_2$. When applied to a lattice, this order helps in comparing and arranging pairs of strategies. For example, in the strategy set $\{1,2,3\}$, the supremum of the pairs (1, 1) and (2, 2) is (2, 2), and the infimum is (1, 1). By using the lexicographic order, we can systematically compare pairs of strategies, ensuring a structured approach to finding Nash equilibria in supermodular games.

In the context of subsets of $\mathbb{R}^n$, a lattice with lexicographic order maintains the same principles but extends to n-dimensional vectors. For any two vectors $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ in $\mathbb{R}^n$, the lexicographic order states that $\mathbf{x} \leq_{lex} \mathbf{y}$ if and only if the first non-equal element from the left is less in $\mathbf{x}$ than in $\mathbf{y}$. The supremum and infimum are defined similarly: the supremum $\mathbf{x} \vee \mathbf{y}$ is the vector where each component is the maximum of the corresponding components of $\mathbf{x}$ and $\mathbf{y}$, and the infimum $\mathbf{x} \wedge \mathbf{y}$ is the vector where each component is the minimum of the corresponding components. This ordering is particularly useful in multi-dimensional decision-making processes where strategies are represented as vectors in $\mathbb{R}^n$.

A lattice with lexicographic order is a complete lattice, a property that supports the analysis of Nash equilibrium existence in ordered games, particularly supermodular games. In supermodular games, the presence of unordered strategies does not pose a significant problem. This is because unordered strategies, such as (a, x) and (a, y) (in the case of a two-player game), will not be compared since each player will not choose the same strategy in response to a change in the opponent's strategy. However, in real conditions, there is always the possibility that players might have no choice but to respond with the same strategy. For instance, if initially the second player chooses the strategy (a_1, x) and then raises their strategy to y, with $x \leq y$, to achieve a higher payoff. Due to the principle of complementarity and the usual order of strategies, the first player will also achieve a higher payoff by choosing a higher-ordered strategy, say a_2, with $a_1 \leq a_2$. In the context of the $\leq$ order, strategy a_1 will not be chosen, but analytically, strategy a_1 cannot be compared (exist) due to the incompleteness of the coordinatewise order $\leq$. On the other hand, the completeness of possible strategy pairs becomes important in a more comprehensive strategy analysis. This is because Player 1 might also only be able to choose strategy a_1, which means that a Nash equilibrium is reached. However, in this context, strategies (a_1, x) and (a_1, y) cannot be compared with the coordinate-wise order. This issue can be resolved with the use of a complete order, one of which is the lexicographic order $\leq_{lex}$.

We will define a complete non-cooperative supermodular games. A defined complete non-cooperative supermodular game is a development of the basic supermodular with a particulary development for lexicographic order. The game used is a triple $G_c = (N, S, F_{i \in N})$. That is assumed to consist of a finite number of players denoted as $N \in \{1, \ldots, n\}$. The strategy played by each Player i is x_i. The game is assumed to consist of a finite number of players denoted as $N \in \{1, \ldots, n\}$. The strategy played by each Player i is x_i. The strategy x_i can be either a single value or a vector value with k vector elements ($k \geq 2$). Each player is assumed to use an equal number of strategies (single or vector form with the same number of k elements). The strategy set $S_i \subset \mathbb{R}^k$ is defined such that $x_i \in S_i$. Furthermore, the feasible strategy set $S_i \subset \mathbb{R}^k$ is defined as the finite Cartesian product of S_i from $i = 1$ to $i = n$, that is $S = S_1 \times S_2 \times \cdots \times S_n$ and denoted as $S := \times_{i=1}^{n} S_i$. The set S is a subset of $\times_{i=1}^{n} \mathbb{R}^p$. The selected strategy vector $x = \{x_1, \ldots, x_2\} \in S$ is the combination of the strategies x_i of each Player and will be chosen to be played. For each $i \in \{1, \ldots, n\}$, the payoff function $F_i : S \to \mathbb{R}$ is defined.

If strategy x_i is played, each Player i receives a payoff of $F_i(x)$. Each player is assumed to use the same complete lexicographical order, denoted as $\leq_{lex}$. Due to the condition of complementary strategies, the preference order in the feasible outcome set induces the same preference order on S. For any chosen strategies $x, y \in S$, $x \leq_{lex} y$ if it satisfies $F_i(x) \leq_{lex} F_i(y)$. For each selected strategy in S and any Player i, x_{-i} generally denotes the strategies of all players in N except Player i. Furthermore, there exists $y_i \in \mathbb{R}^k$ such that (y_i, x_{-i}) denotes the strategy vector where the chosen strategy x_i of Player i is replaced with y_i, while the other components of $\mathbf{x}$ remain unchanged.

Thus, strategy $\mathbf{x}$ can be represented as $x = (x_i, x_{-i})$. Given any other players' strategies x_{-i}, it is denoted as $S(x_{-i})$, with $S_i(x_{-i}) = \{y_i \in S_i | (y_i, x_{-i}) \in S\}$. The set $S_i(x_{-i})$ is the section of S at x_{-i}. The feasible strategy set for any player may depend on the strategy choices of other players. For each Player i, $S_{-i} = \{x_{-i} | S_i(x_{-i}) \neq \emptyset\}$ can be defined. This set S_{-i} is the non-empty collection of all strategies except for Player i such that there exists a strategy y_i for Player i with $(y_i, x_{-i}) \in S$. Therefore, in this case, S_{-i} is also the projection of S onto the strategy set of all players except Player i. For each $i \in 1, \ldots, n$ and any $x^{'}, x^{''} \in S_i$, the supremum of x' and x'' is called the join and is denoted as $x^{'} \vee x^{''}$, while the infimum of x' and x'' is called the meet and is denoted as $x^{'} \wedge x^{''}$. The binary operations of join and meet on the feasible outcome set of each player, which is a subset of $\mathbb{R}^k$, are defined respectively as $x^{'} \vee x^{''} = (x_1^{'} \vee x_1^{''}, \ldots, x_n^{'} \vee x_n^{''})$ and $x^{'} \wedge x^{''} = (x_1^{'} \wedge x_1^{''}, \ldots, x_n^{'} \wedge x_n^{''})$, for any $x^{'}, x^{''} \in \mathbb{R}^n$. Specifically, in $\mathbb{R}$, the lexicographic order $\leq_{lex}$ is applied, with the join and meet operations in $\mathbb{R}$ defined respectively as $x^{'} \vee x^{''} = \sup_{\mathbb{R}} x^{'}, x^{''}$ and $x^{''} = \inf_{\mathbb{R}} x^{'}, x^{''}$, for any $x^{'}, x^{''} \in \mathbb{R}$. The definition of join and meet operations in the strategy set $S_i \subset \mathbb{R}^p$ is analogous to that in the feasible outcome set.

Next, the use of a complete lattice in the existence of equilibrium in the supermodular game G_c will be explained through the following theorem.

Theorem 1. *If $G_c = (N, S, F_{i \in N})$ is complete supermodular game, the set S of feasible joint strategies is nonempty and complete under lexicographical order $\leq_{lex}$, and the payoff function $F_i(y_i, x_{-i})$ is upper semicontinuos in y_i on $S(x_{-i})$ for each x_{-i} in S_{-i} for each x_{-i} in S_{-i} and each i, then the set of equilibrium points is a nonempty lattice and a greatest and a least Nash equilibrium exist.*

It can be easily proven that because S is a complete lattice, it implies that S is compact, thus Theorem 1 will directly follow from the Nash equilibrium existence theorem by [11]. Next, a theorem will be provided regarding the existence of Nash equilibrium in supermodular games defined on a lattice with lexicographic order for specific conditions of each player's payoff function.

Theorem 2. *Given a supermodular game G_c. If $S_i \subset \mathbb{R}^n, i = 1, \ldots, n$ is a nonempty compact lattice, $F_i(.,.), i = \{1, \ldots, n\}$ is upper-continuous in yi on $S_i(x_{-i}) \in S_{-i}$, and payoff*

$F_i(x) = y_i.H(x_{-i})$, the equilibrium set $x^* = (x_i^*, \ldots, x_n^*)$ is a nonempty complete lattice. Furthermore, if x_{-i}^b is a greatest lower bound of the set of feasible strategies of all other players such that $H\left(x_{-i}^b\right) = 0$, then there is exists the greatest Nash equilibrium $x^{*\prime}$ and the least Nash equilibrium $x^{*\prime\prime}$.

Proof. By following the proof by Topkis [11], it can be verified that G_c has two equilibria, that is the largest Nash equilibrium and the smallest Nash equilibrium. Because of $S_i \subset \mathbb{R}^n$ $, i = 1, \ldots, n$ is a nonempty compact lattice, then $S = S_1 \times \cdots \times S_n$ is a nonempty compact lattice. For each $i = 1, \ldots, n, F_i(., x_{-i})$ is both upper semicontinuous and lower semicontinuous at the same time. There is exist $\bar{y}_i \in S_i(\bar{x}_{-i})$ such that $F_i(., x_{-i})$ reach a maximum at point at $\bar{y}_i$ and holds that $F_i(\mathbf{x}, \bar{y}_i) = arg\max_{y \in S} F_i(., x_{-i})$. Moreover, there is $\underline{y}_i$ such that $F_i(., x_{-i})$ reach a minimum point at $\underline{y}_i$ and holds that $F_i\left(\mathbf{x}, \underline{y}_i\right) = \arg\min_{y \in S} F_i(., x_{-i})$. Since $F_i(., x_{-i}), i = 1, \ldots, n$ supermodular in y_i on S_i for each $x_{-i} \in S_{-i}$, then $F_i(., x_{-i}), i = 1, \ldots, n$ have an increasing differences in (y_i, x_{-i}) on $S_i \times S_{-i}$. Furthermore, for each $x_{-i} \leq_{lex} x'_{-i}$ and y'_i,

$$\phi_i\left(y_i, x'_{-i}\right) - \phi_i\left(y_i, x_{-i}\right) \leq_{lex} \phi_i\left(y'_i, x'_{-i}\right) - \phi_i\left(y'_i, x_{-i}\right). \tag{1}$$

Based on Inequality (1), if other players can obtain a higher payoff according to their choice of strategy, then player i can obtain a higher payoff by doing the same thing. If all opposing players ($n-1$ player) of player i do not choose a strategy $x_{-i} = x_i^b$, or there is exist at most l opposing players ($l < n-2$) which play strategy $x_{-i} = x_{-i}^b$, then player i will choose $y_i = x_i^a, x_i \leq_{lex} x_a, x_i \in S_i$ as an optimum response to obtain the higher payoff. Hence, $\left(x_i^a, x_{-i}^a\right)$ is the greatest Nash equilibrium. Based on Inequality (1), if other players can obtain a higher payoff according to their choice of strategy, then player i can obtain a higher payoff by doing the same thing. If all opposing players ($n-1$ player) of player i do not choose a strategy $x_{-i} = x_i^b$, or there is exist at most l opposing players ($l < n-2$) which play strategy $x_{-i} = x_{-i}^b$, then player i will choose $y_i = x_i^a, x_i \leq_{lex} x_a, x_i \in S_i$ as an optimum response to obtain the higher payoff. Hence, $\left(x_i^a, x_{-i}^a\right)$ is the greatest Nash equilibrium. □

4. NUMERICAL COMPUTATIONS

In this subsection, the algorithm to determine the Nash equilibrium of a supermodular game defined in a lattice with lexicographic order will be explained. The Algorithm 1 is written in the Python programming language and executed on the Spyder platform. In the algorithm, we facilitate input in LaTeX equation mode and have validated it with examples of supermodular games. However, the algorithm can only identify one possible Nash equilibrium (generally the largest). It should be noted that in supermodular games, there are two equilibria: the largest and the smallest Nash equilibrium. In many cases, the smallest Nash equilibrium can be identified from the lower bound of the strategy set.

The process begins by requesting the input of each player's payoff function, namely $U_A(q_A, q_B)$ for player A and $U_B(q_B, q_A)$ for player B, where q_A and q_B represent the available strategies for each player. The algorithm then defines the strategy set available to both players, which is assumed to be a discrete set ordered lexicographically. Once the strategies and payoff functions are established, the algorithm evaluates all possible strategy pairs by checking whether each strategy is a best response to the opponent's strategy. For each strategy q_A chosen by player A, the algorithm compares the resulting payoff with the payoff obtained from an alternative strategy $q_A^{'}$. If there exists an alternative strategy $q_A^{'}$ that yields a higher payoff for player A, then q_A is not a best response. A similar process is conducted for player B, where each strategy q_B is compared to its alternatives to determine whether it is a best response to q_A. If a strategy pair (q_A, q_B) satisfies the condition where neither player can improve their payoff by unilaterally changing their strategy, then the pair is classified as a Nash equilibrium. The algorithm continues iterating until all possible strategy combinations have been examined. The final output consists of a list of all strategy pairs that satisfy the Nash equilibrium condition, which is then presented as the result. The complete algorithm is provided in the Appendix.

The complexity of the algorithm is determined based on the number of strategies available to each player in the supermodular game. Let n represent the number of strategies available to player A, and let m denote the number of strategies available to player B. The algorithm evaluates all possible strategy pairs (q_A, q_B), resulting in a total of $n \times m$ strategy combinations that need to be examined. For each strategy pair, the algorithm checks whether the selected strategy is a best response by comparing its payoff against all alternative strategies within each player's strategy set. This process consists of two nested loops for iterating over all possible strategies of the players, as well as two additional loops to evaluate the best response, each with a complexity of $O(n)$ and $O(m)$, respectively. Consequently, the total complexity of the algorithm can be expressed as $O(n^2m + nm^2)$. In the special case where both players have the same number of strategies (n=m), the complexity simplifies to $O(n^3)$. This complexity indicates that the algorithm exhibits polynomial growth in execution time concerning the number of strategies, making it reasonably efficient for games with a small to moderate number of strategies. However, for games with a large number of strategies, the algorithm's complexity may become a limitation, requiring further optimization to improve computational efficiency .

Next, we will provide an illustration of a supermodular game defined within a lattice with a lexicographic order. The example can be solved analytically and numerically using the algorithm described earlier. Let's consider a supermodular game involving two players, A and B, each choosing from strategies 1, 2, or 3. The payoff functions for the players are defined as follows: Player A's payoff function is $u_A(q_A, q_B) = q_A^2 + q_A \cdot q_B - q_B^2$, and Player B's payoff function is $u_B(q_A, q_B) = 2q_B - q_A + q_A \cdot q_B$. To find the Nash equilibrium, we systematically evaluate each strategy combination to determine whether any player has an incentive to deviate given the strategy of the other player. First, we consider the strategy pair (1, 1). For Player A, the payoff is $u_A(1,1) = 1$, and for Player B, the payoff is $u_B(1,1) = 2$. If Player A were to switch to strategy 2, their payoff would increase to 5, indicating an incentive to switch. Thus, (1, 1) cannot be an equilibrium. Next, we evaluate the strategy pair (1, 2). Here, Player A's payoff is $u_A(1,2) = -1$ and Player B's payoff is $u_B(1,2) = 5$. Player B has no incentive to switch as their payoff would decrease. However, Player A would benefit by switching to strategy 2, raising their payoff to 4. Therefore, (1, 2) is not an equilibrium. For the strategy pair (2, 1), Player A receives a payoff of $u_A(2,1) = 5$, and Player B receives $u_B(2,1) = 2$. Player B can improve their payoff by switching to strategy 2, where they would get a payoff of 6. Hence, (2, 1) is not an equilibrium either. When examining the strategy pair (2, 2), Player A's payoff is $u_A(2,2) = 4$ and Player B's payoff is $u_B(2,2) = 6$. In this case, neither player can increase their payoff by switching strategies. Player A switching to strategy 1 would reduce their payoff to -1, and Player B switching to strategy 1 would reduce their payoff to 2. Thus, (2, 2) is a stable strategy pair and a potential Nash equilibrium.

Finally, we consider the strategy pair (3, 3). Here, Player A's payoff is $u_A(3,3) = 9$ and Player B's payoff is $u_B(3,3) = 12$. Switching strategies would decrease their payoffs (Player A would get -5 if switching to 1, and Player B would get 4 if switching to 1). Therefore, (3, 3) is also a stable Nash equilibrium. In conclusion, the Nash equilibrium for this supermodular game is the strategy pair (3, 3), where both players choose strategy 3. In this equilibrium, Player A achieves a payoff of 9 and Player B achieves a payoff of 12, with no incentive for either player to deviate from their chosen strategy. In this example above, the strategy set $\{1,2,3\}$ clearly forms a lattice because each pair of strategies has a supremum and infimum. Strategies are considered in lexicographic order, for instance, when comparing the strategies $(1,1)$ and $(1,2)$.

5. CONCLUSION

Based on the analysis conducted, a lattice with an order has better properties than a lattice with an incomplete order in terms of the process of comparison and ordering of each member of the lattice. In its application to complementary game theory, with a domain that is a complete lattice, the possibility of selecting strategies that do not change when another player increases their strategy value can still be compared within the strategy set. This opens the possibility that such strategies could potentially become Nash equilibria in supermodular games. Additionally, the use of a lattice with lexicographic order paves the way for new theorems that explain the existence of Nash equilibrium under specific conditions of payoff functions.

Funding: This research was supported by RKAT PTNBH Universitas Sebelas Maret Tahun Anggaran 2024, through research grant: Skema Penelitian Fundamental (PF UNS), Grant Number: 194.2/UN27.22/PT.01.03/2024.

REFERENCES

[1] S. Penmatsa and A. T. Chronopoulos, "Game theoretic static load balancing for distributed systems," *Journal of Parallel and Distributed Computing*, vol. 71(4), pp. 537–555, 2011.

[2] D. Grosu and A. T. Chronopoulos, "Noncooperative static load balancing for distributed systems," *Journal of Parallel and Distributed Computing*, vol. 65(9), pp. 1022–1034, 2005.

[3] M. Y. Leung et al., "Load balancing in distributed systems: An approach using cooperative games," *IEEE 16th International Parallel and Distributed Processing Symposium (IPDPS 2002), Fort Lauderdale, Florida*, pp. 8–11, 15-19 April 2002.

[4] D. Grosu and A. T. Chronopoulos, "A game-theoritic model and algorithm for load balancing in distributed systems," *IEEE 16th International Parallel and Distributed Processing Symposium (IPDPS 2002), Fort Lauderdale, Florida*, pp. 146–153, 2002.

[5] J. Bilbao, *Cooperative games on combinatorial structures*. Springer Science+Business Media, 2012, vol. 26.

[6] M. Van der Merwe, *Non-cooperative games on networks*. Ph.D dissertation, Stellenbosch University, 2012.

[7] F. R. Csori, *Structural and computational aspects of simple and influence games*. Tesis, Universitat Politecnica de Catalunya, 2014.

[8] D. M. Topkis, “Minimizing a submodular function on a lattice,” *Operations Research*, vol. 26(2), pp. 305–321, 1978.

[9] P. Milgrom and C. Shannon, “Monotone comparative statics,” *Econometrica*, vol. 62(1), pp. 157–180, 1994.

[10] G. Cachon, “Stock wars: Inventory competition in a two-echelon supply chain with multiple retailers,” *Oper. Res.*, vol. 49(5), pp. 658–674, 2001. doi: 10.1287/opre.49.5.658.10611

[11] D. M. Topkis, *Supermodularity and Complementary*. Princeton University Press, 1998.

[12] G. Cachon and S. Netessine, *Game Theory in Supply Chain Analysis, Handbook of Quantitative Supply Chain Analysis: Modelling in the E-Business Era*. Springer Science+Business Media, 2024.

[13] R. Setiawan, Salmah., I. Endrayanto, and Indarsih., “Analysis of the n-person noncooperative supermodular multiobjective games,” *Oper. Res. Lett.*, vol. 51, pp. 278–284, 2023. doi: 10.1016/j.orl.2023.03.007

[14] X. Vives, “Nash equilibrium with strategic complementarities,” *J.Math.Econ.*, vol. 19, pp. 305–321, 1990.

[15] G. Rota-Graziosi, “The supermodularity of the tax competition game,” *J.Math.Econ.*, vol. 83, pp. 25–35, 2019.

[16] G. Koshevoy, T. Suzuki, and D. Talman, “Supermodular ntu-games,” *Oper. Res. Lett.*, vol. 44, pp. 446–450, 2016. doi: 10.1016/j.orl.2016.04.007

APPENDIX

Algorithm 1 Determine Nash Equilibrium for a Supermodular Game using Lexicographical Order

```
1:  Function GetPayoffFunction(player)
2:    Print "Enter the payoff function for player", player, "in LaTeX format"
3:    payoff_function_latex ← User input
4:    Return payoff_function_latex
5:  End Function
6:  Function PayoffFunctionWrapper(function_str, q1, q2)
7:    Evaluate function_str with q1 and q2 using sympy
8:    Return Result of evaluation as float
9:  End Function
10: Function FindNashEquilibrium(payoff_A_str, payoff_B_str)
11:   strategies ← [1, 2]
12:   nash_equilibria ← []
13:   for each qA in strategies do
14:     for each qB in strategies do
15:       A_best_response ← True
16:       B_best_response ← True
17:       for each qA_prime in strategies do
18:         if PayoffFunctionWrapper(payoff_A_str, qA_prime, qB) > PayoffFunction-
            Wrapper(payoff_A_str, qA, qB) then
19:           A_best_response ← False
20:         end if
21:       end for
22:       for each qB_prime in strategies do
23:         if PayoffFunctionWrapper(payoff_B_str, qB_prime, qA) > PayoffFunction-
            Wrapper(payoff_B_str, qB, qA) then
24:           B_best_response ← False
25:         end if
26:       end for
27:       if A_best_response and B_best_response then
28:         nash_equilibria.append((qA, qB))
29:       end if
30:     end for
31:   end for
32:   Return nash_equilibria
33: End Function
34: payoff_A_str ← GetPayoffFunction('A')
35: payoff_B_str ← GetPayoffFunction('B')
36: nash_equilibria ← FindNashEquilibrium(payoff_A_str, payoff_B_str)
37: Print "Nash Equilibria:", nash_equilibria
```