Stable Coalitions of Buyers in Real World Agriculture Domain

Chattrakul Sombattheera(✉)

Multiagent, Intelligent and Simulation Laboratory (MISL),
Faculty of Informatics, Mahasarakham University,
Khamreang, Kantarawichai, Mahasarakham, Thailand
chattrakul.s@msu.ac.th

Abstract. The Department of Agricultural Extension, Ministry of Agriculture, Thailand, has developed and deployed an AI-based system, namely, Personalized Data (PD), to help millions of Thai farmers to make better decisions with regards to growing and selling crops. One of the AI module equipped with the system applies cooperative game theoretic principles, namely Kernel, a stability concept, as important part of the system. While most applications of game theory in real world domains concentrate on one game setting, this system may have to scan for much larger search space to ensure stability. This paper examines how much time it take and how large the search space can be in order to examine stability in practice. Although there are several algorithms involved, we present only ones for generating coalitions and examining whether the given payoff configuration is in Kernel. The former repeatedly generates all coalitions of a given set of agents. The latter compares between each pair of agents in every coalition of the given configuration payoff whether one of the agent's payoff outweighs the other's. For the search space of 15–30 agents, we find that it can range from 11 to 20 °C of magnitude. For execution time, we find that it takes minutes for up to 18 agents and it takes hours for 19 agents. For larger number of agents, it can take days or months and will require a much more powerful computer to reduce the execution time.

Keywords: Stable · Kernel · Coalition formation · Agriculture

1 Introduction

As part of its strategic plan for the next 20 years, the government of Thailand has launched the national policy on deploying modern technologies to help drive the nation to prosperity. One of the technologies being mentioned in the plan is artificial Intelligence. As it has been seen world wide that the adoption of AI has been widely successfully in many areas. Agriculture has been a successful area where AI has been increasingly deployed world wide. Based on this fact,

Supported by the Department of Agriculture Extension of Thailand.

O. Surinta and K. Kam Fung Yuen (Eds.): MIWAI 2022, LNAI 13651, pp. 178–190, 2022.
https://doi.org/10.1007/978-3-031-20992-5_16

the Thai government has been driving and hoping for the development on this area with the country.

The Department of Agricultural Extension (DOAE), Ministry of Agriculture, of Thailand has been quick to react to this call. DOAE has been maintaining large databases, collecting data from thousands of its personnel and millions of farmers all over the country for more than 40 years. It is keen to adopt AI technologies to utilize this massive amount of data to be personalized to help farmers make better decisions based on their interests. There are two main AI modules: machine learning-based for predicting prices, yields and costs of crops, and game theoretic modules, both cooperative and non-cooperative, for analyzing the derived data and help make better decision for both DOAE personnel and farmers. For cooperative game theory, there are three important solution concepts, namely, optimal coalition structure for global and individual optimality, Shapley value for fairness among coalition members, and Kernel for stability among coalition members.

This paper focuses on the stability issue of coalition formation among farmers. The Kernel is, one of, if not, the most widely adopted stability concepts. In general, the payoff configuration, a vector specifying how farmers form coalitions and their respective payoffs, will be provided. Then it will be examine whether the payoff configuration is in Kernel. This is not very easy to examine because the algorithm has to go to almost each coalition, out of 2^n coalitions for n farmers, for each agent. This makes it almost 2^{2n}, which can be very large for small n. Although optimal coalition structure is likely to guarantee stability in most cases, particularly for superadditive environment, it is still important that we have to thoroughly examine to ensure stability. We thoroughly analyze the size of the search space and empirically investigate to ensure how large can n be such that the result can be achieved in reasonable, taking into account the reality that farmers and DOAE personnel are awaiting for the results. This paper's contribution is to practically show how large search space can be and how long it takes to ensure stability among agents.

The paper is structured as follow. We review for progress in coalition formation in agricultural domain. We then briefly discuss the circumstance of this project. We discuss in details for fundamental concept of coalition formation and Kernel. We then discuss about experiments and results, then conclude, lastly.

2 Related Works

Game theory [5], both non-cooperative [4] and cooperative, has been widely adopted in AI research. Cooperative game, also known as coalition formation [3,17], provides fundamental concepts for cooperation among decision making. Important solution concepts include efficiency (core [2] and optimal coalition structure [6]), stability [1], and fairness [7]. Here we shall review related work being deployed in agriculture and related domains.

Bistaffa et al. [8] consider coalition formation among energy consumers in the smart grid applying stability concept. The peaks of demands are flatten.

Blankernburg et al. [9] investigate safety and privacy preservation in forming stable coalitions among informative agents. Yamamoto et al. [10] propose a coalition formation scheme in e-commerce to buy as a group on a category of items. This system allows for buyers to post their needs and the system combine these needs a bunch. Sellers then bids for the request. Guthula et al. [11] model a specific troubled agricultural sub-system in India as a Multi-Agent System and use it as a tool to analyze the impact of policies and recommend some policies based on simulation result. Zaryouli et al. [12] help establish a predictive analysis on the impact of climatic change on red fruits by proposing solutions to optimize crop growth decisions by increasing yields and profitability of production for the farmer. Perez-ponz et al. [13] help make decisions in the purchase of sustainable agricultural products by proposing a multiagent system choosing a supplier for agricultural future market price forecast. Chevalier et al. [14] help achieve precision agriculture, combining quadrotor and tracked robots, by using coalition formation concept. Gonzalez et al. [15] help reduce water usage and increase efficiency and effectiveness in automotive irrigation processes in rural areas by using cooperative game concept.

As we can see, there is almost none of stability concept has been deployed in agricultural domain in recent years. The concept has been used in theoretical work or e-commerce domain. Therefore, this Personalized Data is among the first to apply this concept to help farmers make better decisions.

3 Real World Domain

One of the reasons DOAE wants to use AI to help farmers is to leverage their hidden collective power on both producing and negotiating for higher prices when selling and lower prices when buying. This system allows for farmer to specify their needs for fertilizers, seeds, etc. and collectively gain discounts, which to be distributed among participating agents. Any farmer can submit her/his need for buying as a group to other farmers to participate. Interested farmers can choose to participate. Not interested agents can insist to be the group leader. Participating agents specify their needs and constraints, for example, 5 tons of fertilizers of a reserved price maximum. There could be many farmers submitting requests at the same time. The system will scan for farmers looking for the same thing and are located in close proximity. Figure 1 shows a set of screen snapshots of farmers.

3.1 Computing Values for a Group of Farmers

Farmers looking for values accruing from joining a group. The values can be discounts achieved from bulk buy, Typically bulk buy help save a fair amount of expenses for sellers. From sellers perspective, the saved amount can be shared to buyer in order to attract them. On the other hand, farmers can also sell as a group and ask for higher prices because they can save buyers, who look for a large amount of crops, time and expenses in order to collect the crops up to their

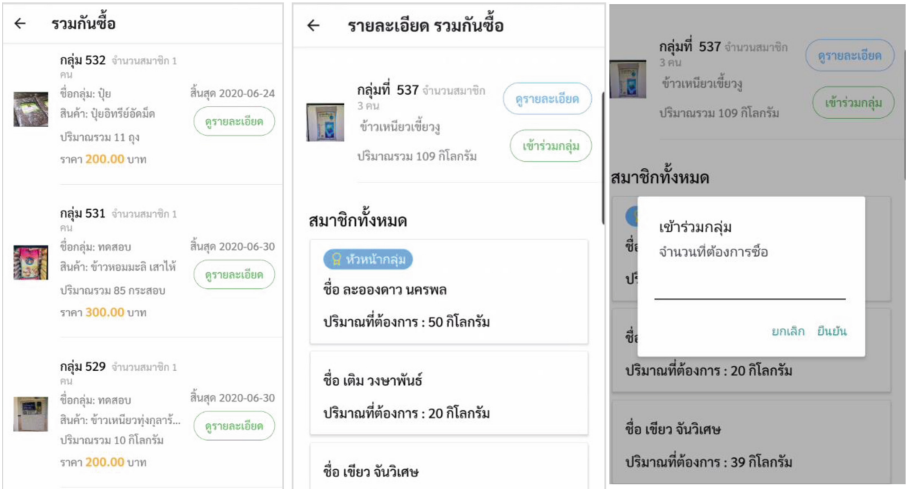

Fig. 1. Screen snapshots of farmers looking for available groups (left), existing members (center), and confirming joining a group (right.)

need. From buyers perspective, the time and expenses they can save are to be shared to farmers in order to attract them. From the system point of view, with respect to cooperative game theory, the saved amount to be shared to farmers is the value of forming a group, or coalition, of farmers. This value is yet to be distributed to farmers. There are two steps involved in computing values: *i*) is to find coalition center, and *ii*) to find the net value for the group. For selling crops, there must be a center of the coalition where the crops will be collected from members. For buying goods, the center of the coalition is where the goods will be delivered to and distributed to members. The center is the location of farmer where the collective distances to other agents is minimal. This ensures that the cost for delivering goods/crops is collectively the cheapest. The net value is the discounted or increased total price subtracted by the collective cost.

3.2 Computing Payoffs for Farmers

There another very important step involved, computing payoffs for farmers. There are multiple solution concepts in cooperative game that can be adopted, for example, efficiency, fairness, stability, etc. This Personalized System is equipped with these features. From farmer perspective, efficiency is the discounted or increased price they will receive. However, they simply keen to know whether their shared value is fair. With this regard, the system adopts Shapley value. We have discuss in details about this in another work. The last one is stability, which can be ensured by Kernel solution concept. We shall discuss in details in section below.

4 Stability in Coalition of Farmers

This system adopts the concept of cooperative game, also known as *coalition formation*. Below, we shall discuss the fundamental concept of coalition formation and Kernel stable concept.

4.1 Coalition Formation

Foundation of Coalition. A set of n agent is denoted by $A = \{a_1, a_2, \ldots, a_n\}$. A *coalition*, denoted by S of A, $S \subseteq A, S \neq \emptyset$, is a non-empty subset of of A. Therefore, the size of coalition, or the number of agents in S, also known as *cardinality*, $1 \leq |S| \leq n$, ranges between 1 and n. In practice, a coalition is formed once agents agree to cooperate and are abiding to their respective coalitions. The smallest coalition, $|S| = 1$, is the *singleton* coalition. Obviously, there are n singleton coalitions. The set A itself is the largest coalition, $|S| = n$, namely the *grand* coalition, The power set of A is the set of all coalitions, denoted by $\mathbf{S}$, where $\mathbf{S}$ is $2^n - 1$.. Given a set of 3 agents, $A = \{a_1, a_2, a_3\}$, there are $2^3 - 1 = 7$ coalitions. All the coalition $\{a_1\}, \{a_2\}, \{a_3\}, \{a_1, a_2\}, \{a_1, a_3\}, \{a_2, a_3\}$ and $\{a_1, a_2, a_3\}$. Once all agents in a give game agree to form coalitions, they are regarded as being mutually exclusive and exhaustive partitioned to m coalitions. Such a partitioned set of agents, A, is known as a *coalition structure*, denoted by $CS = \{S_1, S_2, \ldots, S_m\}$. The following mathematical definition defines mutual and exhaustive conditions: 1) $S_j \neq \emptyset, j = 1, 2, \ldots, m,$, 2) $S_i \cap S_j = \emptyset$ for all $i \neq j$, and 3) $\bigcup S_j = N$. For example, given $A = \{a_1, a_2, a_3\}$, all CS in $\mathcal{CS}$ are $\{\{a_1\}, \{a_2\}, \{a_3\}\}, \{\{a_1, a_2\}, \{a_3\}\}, \{\{a_1, a_3\}, \{a_2\}\}, \{\{a_2, a_3\}, \{a_1\}\}$, $\{\{a_1\}, \{a_2\}, \{a_3\}\}$. We denote the set of all CSs of A by $\mathcal{CS}$. The benefit jointly accruing from a coalition is known as *coalition value*. In general, coalition value can be anything valuable to agents. In game theory, coalition values are mostly money value, which can be dollars, pounds, etc. Whereas coalitions in real world need to do real business to achieve their values, we assume the $characteristic function, v : S \rightarrow \mathbb{R}^+$, associates to each coalition a non-negative value as its coalition value. The coalition value of a coalition S is denoted by $v(S)$. A coalitional game defined by a characteristic function is known as a *characteristic function* game. Here is an example of such a game of three agents defined by a characteristic function: $v(\{a_1\}) = v(\{a_2\}) = v(\{a_3\}) = 0$; $v(\{a_1, a_2\}) = 90; v(\{a_1, a_3\}) = 80; v(\{a_2, a_3\}) = 70$; $v(\{a_1, a_2, a_3\}) = 105$.

Payoff and Payoff Configuration. The ultimate reason each agent decides to join a coalition is the share of the joint benefit it receives from that coalition. This share is known as the *payoff* of the agent. Mathematically, agent a_i in its respective coalition is given a *payoff*, U_i. The *payoff vector*, $U = (U_1, U_2, \ldots, U_n)$, specifies the payoff for each agent. Generally, the payoff of each agent and the coalition structure are always associated to decide for the outcome. This association is known as the *payoff configuration* [16], $(U; CS)$ specifying payoffs of

agents and how agents form coalitions. Considering the game mention above, the payoff configuration, $(45, 0, 35; \{a_1, a_3\}, \{a_2\})$ specifies that agents decide to form two coalition $\{a_1, a_3\}$ and a_1, a_1 receives payoff 45, a_2 receives payoff 0, and a_3 receives payoff 35, respectively.

Solution Concept. The state which leads to the final and stable state of each game is known as *equilibrium*. The principles which agents can use to decide for equilibrium of each game is known as *solution concept*. There are three important solution concepts in coalition formation: *i*) **fairness** which helps agents receive fair payoffs based on agents' contribution to the coalition values, *ii*) **efficiency** which helps agents receive highest possible payoffs both individually and globally, and *iii*) **stability** which helps agents receive satisfactory payoffs such that there is no need for agents to deviate from their current payoffs for higher payoffs. In the next solution, we shall discuss a stability solution concept, namely, the Kernel.

4.2 Kernel Solution Concept

Principle. To help define a stable condition, Davis et al. [1] propose that agents do not need to deviate from their respective coalitions for higher payoffs if they have already achieved the highest possible payoffs. This solution concept is named the *Kernel*. that stabilizes coalition structure by balancing each pair of agents payoffs in every coalition. Given a payoff configuration $(U; CS)$ for a game $(N; \upsilon)$, there may be a group of agents contemplate leaving their respective coalitions to form a new coalition R for more payoff. The *excess*: $e(R; U) = \upsilon_R - U_R$ is difference between the value of R and the sums of the collective payoffs, U_R, of the agents. Considering any pair of agents $a_i \in S$ and $a_j \in S$, and $S \in CS$. Agent a_i may join R alone, $R \notin CS$ and $a_j \notin R$. Coalition R, any similar coalitions, will have excess (U, CS). The largest excess is the *maximum surplus* of a_i over a_j with respect to $(U; CS)$, e.g., $\mathcal{S}_{i,j} = max_{R|i \in R, j \notin R} \quad e(R; U)$. Agent, a_i can potentially gain that higher payoff. Similarly, a_j can do the same, with maximum surplus $\mathcal{S}_{a_j, a_i}$. If both a_i's maximum surplus is greater than that of a_j and a_j's payoff is greater than its singleton coalition's value, $\mathcal{S}_{i,j} > \mathcal{S}_{j,i}$ and $U_j > \upsilon_j$, then a_i **outweighs** a_j, with regards to $(U; CS)$ [1]. Agent a_i ask for compensation from a_j because of its higher maximal surplus otherwise it could join R for higher payoff. However, a_i can ask for up to a certain value because a_j can only accept payoff at least υ_j It is said that an equilibrium between each pair of agents exists when one of the following holds: *i*) $\mathcal{S}_{i,j} = \mathcal{S}_{j,i}$; each agent cannot ask for compensation be their maximum surpluses are equal, *ii*) $\mathcal{S}_{i,j} > \mathcal{S}_{j,i}$ and $U_j = V_j$; agent a_i cannot ask for compensation from a_j because agent a_j can be in its $\{a_j\}$ *iii*) $\mathcal{S}_{j,i} < \mathcal{S}_{i,j}$ and $U_i = V_i$; agent a_j cannot ask for compensation from a_i because agent a_i can be in its $\{a_i\}$. The global equilibrium is defined as the set **K** [1] of all payoff configurations $(U; CS)$, namely "Kernel", such that every pair of agents $a_i, a_j \in S \in CS$ are in equilibrium. Given this global equilibrium, it is said that the coalitions are stable.

Example. Let us consider the game defined in Sect. 4.1. Suppose we are given payoff configuration $(U; CS) = (45, 0, 35; \{1, 3\}, \{2\})$ to examine if it is in Kernel. Firstly, we consider coalition $\{a_1, a_3\}$. For a_1, it may join, excluding a_3, two coalitions, e.g., $\{a_1\}$ and $\{a_1, a_2\}$. The excesses and maximum surplus are $e(a_1; U) = \upsilon_1 - U_1 = 0 - 45 = -45$, $e(\{a_1, a_2\}; U) = \upsilon_{1,2} - U_1 = 90 - 45 = 45$, and $mathcalS_{1,3} = max(-45, 45) = 45$. Similarly, $\mathcal{S}_{a_3,a_1} = 35$. Hence, a_1 outweighs a_3 because $\mathcal{S}_{a_1,a_3} = 45 > \mathcal{S}_{a_3,a_1} = 35$. Therefore, payoff configuration $(U; C) = (45, 0, 35 : \{a_1, a_3\}, a_2)$ is not in the Kernel. Consider another payoff configuration $(U; CS) = (50, 30, 25; \{a_1, a_2, a_3\})$. The excesses and maximum surplus between agent a_2 and a_3 are: $\mathcal{S}_{2,3} = max(\upsilon_2 - U_2, \upsilon_{1,2} - U_2) = max(0 - 30, 90 - 30) = 60$ and $\mathcal{S}_{3,2} = max(\upsilon_3 - U_3, \upsilon_{1,3} - U_3) = max(0 - 25, 80 - 25) = 55$. Hence, a_2 outweighs a_3, or $(50, 30, 25; \{a_1, a_2, a_3\})$ is not in Kernel. Given another payoff configuration $(U; C) = (45, 35, 25; \{a_1, a_2, a_3\})$, we consider balance between each pair of them. Agent a_1 and a_2 are in equilibrium because $\mathcal{S}_{1,2} = max(0 - 45, 80 - 70) = 10 = max(0 - 35, 70 - 60) = \mathcal{S}_{2,1}$. Agent a_1 and a_3 are in equilibrium because $\mathcal{S}_{1,3} = max(0 - 45, 90 - 80) = 10 = max(0 - 25, 70 - 60) = \mathcal{S}_{3,1}$. Agent a_2 and a_3 are in equilibrium because $\mathcal{S}_{2,3} = max(0 - 35, 90 - 80) = 10 = max(0 - 25, 80 - 70) = \mathcal{S}_{3,2}$. Therefore, the $(U; C) = (45, 35, 25; \{1, 2, 3\})$is in the Kernel.

5 Algorithms

5.1 Overview

As presented above, it takes a payoff configuration to determine whether the payoffs for agents are in the Kernel. There are three algorithms involved in verifying stability of coalitions. *i*) **Generate Coalitions** Since a coalition is merely a subset, any algorithm for generating subset can be applied for this purpose. The additional requirement is to associate a coalition value to each coalition. *ii*) **Generate Coalition Structure** This is quite a complex algorithm. There are two strong constraints, i.e. exclusiveness and exhaustiveness. *iii*) **Verify Kernel** This is also a complex algorithm because it has to iterate through a lot of coalitions. Because of limited space, we choose to demonstrate only algorithms for generating coalitions and verifying Kernel. This is based on the theoretical principle described above that a certain coalition structure is given as the input, along with payoff vector, in the payoff configuration. On the other hand, all coalitions must be repeatedly generated for examining whether it is in the Kernel.

5.2 Algorithm to Generate Coalitions

The coalitions are to be created in lexicographical order as shown in previous section, i.e. the coalition member at the right most position is the highest and the most value of the next member to the left is less by 1. We generate coalitions in form of array of integer, starting from size 1 to n. There is a main for loop where

array *coal* of the respective size is created. Each cell of the array is initialized with its corresponding index. The algorithm enters the second loop where array *coal* is output. Another array, *temp*, is then created and initialized with the value of *coal*. For each position in *temp*, the algorithm checks within the current size $i \leq n$ if its value can be increased. If it has reached the maximum value allowed, it moves to the next left position until there is nothing left.

Algorithm 1. Coalition Generation Algorithm

```
for i = 1 to n do
  int array coal[i]
  for j = 1 to i do
    coal[j] = j
  end for
  while coal ≠ null do
    output coal
    int array temp[k]
    for t = 1 to i do
      temp[t] ← coal[t]
    end for
    k ← i
    while k ≥ 1 and coal[k] = n − i + k do
      k ← k − 1
    end while
    if k = 0 then
      return
    else
      for j = k to i do
        temp[j] ← coal[k] + 1 + j − k
      end for
    end if
    for j = 1 to k do
      coal[j] ← temp[j]
    end for
  end while
end for
```

Although there are many ways to generate subsets or coalitions, our algorithm is suitable for verifying if a payoff configuration is in Kernel. According to the principle, each pair of agents in every coalition in the coalition structure of the given payoff configuration must be examined against almost coalitions for stability. Therefore generating a list of coalitions of the given set A and storing it for later referencing is suitable for our purpose.

5.3 Algorithm to Verify Kernel

At presented above, the concept of Kernel requires that we have to explore a lot of possible coalitions, i.e. up to 2^n, for every pair of agents in each coalition. In

order to complete the search, at least, the algorithm needs a payoff configuration, a characteristic function (from game theory perspective, or a list of coalition values), of a given game. Firstly, the payoff configuration is assumed to be in Kernel, $inKernel$ is set to true. The algorithm enters the main loop which goes through each S in the CS of the given payoff configuration. The second loop is to goes through each a_i. The third loop is to go through each $a_j, i \neq j$, checking for each pair a_i, a_j in the coalition. The forth loop is to go through all possible coalitions R, then determine their excesses and maximum surpluses. After that, the algorithm verifies the three conditions. If one of them is violated, the algorithm returns $false$. It is required by the principle that the verification must be done exhaustively. While the algorithm proceeds, the algorithm returns false whenever it finds that the one the three conditions is violated. We illustrate the algorithm below.

Algorithm 2. Kernel Algorithm

```
for each S ∈ CS do
  S_{i,j}
  for each a_i ∈ S do
    for each a_j ≠ a_i and a_j ∈ S do
      for each R ⊂ A and R ≠ S do
        U ← 0
        for each a_k ∈ R do
          U ← υ_k
        end for
        e(R;U) ← υ_R − U_R
        if S_{i,,j} <   e(R;U) then
          S_{i,,j} ← e(R;U)
        end if
      end for
      if !(S_{i,j} = S_{j,i}) or !(S_{i,j} > S_{j,i} and U_j = V_j) or !(S_{i,j} > S_{j,i} and U_j = V_j)
      then
        InKernel ← false
        return false
      end if
    end for
  end for
end for
return true
```

As mentioned, this algorithm strictly follow the principle of examining Kernel. One may use the figures provided in the previous section to learn and further understand Kernel algorithm.

6 Experiments and Results

There are two parts we consider about using the solution concept Kernel in real world setting. The first part is to analyze how large the search space can be. Due to the fact that the algorithm has to exhaustively search for all possible cases whether any agent can deviate for better payoffs, it is important to know how large the search space can be. As we have already mentioned in the review, Kernel is hardly adopted. We therefore present statistical results of our experiments. We conducted the experiments on a Ryzen 9 CPU, 16 core 32 thread and 32GB ram computer. Since coalition values can be varied, we divide the search space into 10 ranges, i.e. 10%, 20%, ..., 100%, of the number of coalitions. We consider cases, where $10 \leq n \leq 30$, whose number of coalitions are defined by 2^n, e.g., $1,024, 2,048, \ldots, 1,073,741,824$. Then we calculate, how large is the search space by multiply the number of coalitions for each n by $10\%, 20\%, \ldots, \%100$, respectively.

		Percentage for Iteration									
n	2^n	10	20	30	40	50	60	70	80	90	100
10	1024	1E+07	2.1E+07	3.1E+07	4.2E+07	5.2E+07	6.3E+07	7.3E+07	8.4E+07	9.4E+07	1E+08
11	2048	4.2E+07	8.4E+07	1.3E+08	1.7E+08	2.1E+08	2.5E+08	2.9E+08	3.4E+08	3.8E+08	4.2E+08
12	4096	1.7E+08	3.4E+08	5E+08	6.7E+08	8.4E+08	1E+09	1.2E+09	1.3E+09	1.5E+09	1.7E+09
13	8192	6.7E+08	1.3E+09	2E+09	2.7E+09	3.4E+09	4E+09	4.7E+09	5.4E+09	6E+09	6.7E+09
14	16384	2.7E+09	5.4E+09	8.1E+09	1.1E+10	1.3E+10	1.6E+10	1.9E+10	2.1E+10	2.4E+10	2.7E+10
15	32768	1.1E+10	2.1E+10	3.2E+10	4.3E+10	5.4E+10	6.4E+10	7.5E+10	8.6E+10	9.7E+10	1.1E+11
16	65536	4.3E+10	8.6E+10	1.3E+11	1.7E+11	2.1E+11	2.6E+11	3E+11	3.4E+11	3.9E+11	4.3E+11
17	131072	1.7E+11	3.4E+11	5.2E+11	6.9E+11	8.6E+11	1E+12	1.2E+12	1.4E+12	1.5E+12	1.7E+12
18	262144	6.9E+11	1.4E+12	2.1E+12	2.7E+12	3.4E+12	4.1E+12	4.8E+12	5.5E+12	6.2E+12	6.9E+12
19	524288	2.7E+12	5.5E+12	8.2E+12	1.1E+13	1.4E+13	1.6E+13	1.9E+13	2.2E+13	2.5E+13	2.7E+13
20	1E+06	1.1E+13	2.2E+13	3.3E+13	4.4E+13	5.5E+13	6.6E+13	7.7E+13	8.8E+13	9.9E+13	1.1E+14
21	2E+06	4.4E+13	8.8E+13	1.3E+14	1.8E+14	2.2E+14	2.6E+14	3.1E+14	3.5E+14	4E+14	4.4E+14
22	4E+06	1.8E+14	3.5E+14	5.3E+14	7E+14	8.8E+14	1.1E+15	1.2E+15	1.4E+15	1.6E+15	1.8E+15
23	8E+06	7E+14	1.4E+15	2.1E+15	2.8E+15	3.5E+15	4.2E+15	4.9E+15	5.6E+15	6.3E+15	7E+15
24	2E+07	2.8E+15	5.6E+15	8.4E+15	1.1E+16	1.4E+16	1.7E+16	2E+16	2.3E+16	2.5E+16	2.8E+16
25	3E+07	1.1E+16	2.3E+16	3.4E+16	4.5E+16	5.6E+16	6.8E+16	7.9E+16	9E+16	1E+17	1.1E+17
26	7E+07	4.5E+16	9E+16	1.4E+17	1.8E+17	2.3E+17	2.7E+17	3.2E+17	3.6E+17	4.1E+17	4.5E+17
27	1E+08	1.8E+17	3.6E+17	5.4E+17	7.2E+17	9E+17	1.1E+18	1.3E+18	1.4E+18	1.6E+18	1.8E+18
28	3E+08	7.2E+17	1.4E+18	2.2E+18	2.9E+18	3.6E+18	4.3E+18	5E+18	5.8E+18	6.5E+18	7.2E+18
29	5E+08	2.9E+18	5.8E+18	8.6E+18	1.2E+19	1.4E+19	1.7E+19	2E+19	2.3E+19	2.6E+19	2.9E+19
30	1E+09	1.2E+19	2.3E+19	3.5E+19	4.6E+19	5.8E+19	6.9E+19	8.1E+19	9.2E+19	1E+20	1.2E+20

Fig. 2. Progresses of contribution values of 24 (left) and 25 (right) agents

As shown in Fig. 2, the search space can be very large. Since the search space is large, we herein will consider the degree of magnitude. The largest one is degree 20, existing in two cases, 90% and 100%, where $n = 30$. There are Given that most CPUs existing today runs at a few GHz. Let us assume, at the very best case, that it completes one case per a clock cycle. We may complete cases of magnitude degree 9 in a matter of seconds. In the worst case scenarios,

to complete cases of magnitude degree 12, it may take minutes or hours. To complete cases of magnitude degree 13, 14, and 15, it will take hours, days, weeks and months, respectively. If we use 10 CPUs to handle magnitude degree 13, it will approximately bring down the search time to hours. If we use 100 CPUs to handle magnitude degree 14 (or 15), it will approximately bring down the search time to hours. Magnitude degree 15 is found in when $n = 24$(10%–30%), $n = 23$(20%–100%), and $n = 22$(60%–100%). These are the worst cases where we can finish the search at the best possibile within hours, using hundreds of CPUs. Note that we are not searching the maximal or minimal points. Instead we must be able to exhaustively search the whole search space to ensure the correct results should the worst case scenario exist (Fig. 3).

		Search Space									
n	size	[10%]	[20%]	[30%]	[40%]	[50%]	[60%]	[70%]	[80%]	[90%]	[100%]
10	1,024	6	3	24	2	1	3	3	2	3	4
11	2,048	13	3	5	6	9	8	11	12	14	15
12	4,096	6	15	19	27	34	40	46	54	62	65
13	8,192	29	57	85	114	140	168	196	226	260	293
14	16,384	130	247	367	488	729	757	861	991	1,155	1,297
15	32,768	551	1,065	1,593	2,117	2,629	3,115	3,659	4,374	4,819	5,256
16	65,536	2,286	4,425	6,865	8,845	11,456	13,326	15,993	18,064	20,175	22,717
17	131,072	9,738	19,460	28,774	39,335	47,806	55,365	64,454	73,840	83,834	91,304
18	262,144	38,644	77,369	115,863	155,160	195,340	231,334	268,763	308,315	346,511	383,752
19	524,288	161,533	323,065	495,370	671,719	818,702	982,994	1,186,361	1,301,344	1,463,397	1,626,389

Fig. 3. Elapsed time of exhaustive search through 10–19 agents.

The second part is to carry out a set of experiments to investigate the execution time it might take to exhaustively examine stability, given the above analysis, in the best case scenarios. According to the definition of Kernel that for each agent in each coalition of a given payoff configuration, we have to exhaustively search by constructing nested loops to cover all the possibilities. Note that we assume that the actual elapsed time for comparison for each pair of coalitions is minimal. We have completed the experiment of 18 agents, all ranges, 10%–100% with magnitude degree 11–12. It took 383,752 milliseconds, over 6 min, to complete the -100% range. For the cases of 19 agents, range 100%, and magnitude degree 13, it took 1,626,389 milliseconds or almost 30 min to complete. At the time of writing this report, we can finish 60% range of 20 agents with magnitude degree 13 by 5,028,333 milliseconds, over 83 min, or 1 h and 23 min. This means the 100% range of 10 agents with magnitude degree 14 could easily takes days. If we take the problems of 15 °C of magnitude, 30% of 24 agents, 20%–100% of 23 agents, and 60%–100% of 22 agents, it may take up to months. In this case, we might need hundreds of CPUs to finish the search in hours.

7 Conclusion

Based on the Personalized Data system, developed by the Department of Agricultural Extension, Ministry of Agriculture, Thailand. Among many AI libraries

and concepts used, the stability concept, Kernel, is needed when farmers cooperate and do not deviate for other alternative groups for better payoffs. We must be ensured that given any worst case scenario in real world environment, we can still deliver the answer in reasonable time This means the number of agents participating in forming coalitions must not be too large. We firstly analyze how large the search space can be. We found that with up to 30 agents, the largest space is 1.2×10^{20}. By approximation with typical GHz CPUs, we may finish the search with minutes for 18 agent, hours for 19 agents, and days for 20 agents. We may bring down these figures by adding more CPUs by 10 s or 100 s of them. We have also carried out the approximate search time, assuming that the exact comparison between each pair of coalitions can be done at minimal time, for up to 60% of 20 agents. This takes almost an hour and a half. This implies that by using single CPU for 20 agents we may need merely hours. For 21 and 22 agents it might take days or weeks for a single CPU. By adding 10 s or 100 s CPUs, it may take hours or days for 23 or 24 agents. Therefore, we can use this system in practice to verify if a payoff configuration is in Kernel. In the future, more advanced techniques can be developed to search for a payoff configuration that is in Kernel.

References

1. Davis, M., Maschler, M.: The kernel of a cooperative game. Naval Res. Logist. Q. **12**(3), 223–259 (1965)
2. Gillieszz, D.B.: Some Theorems In N-person Games. Princeton University, New Jersey (1953)
3. Kahan, J.P., Rapoport, A.: Theories of Coalition Formation. Psychology Press, New York (1984)
4. Nash, J.F.: Equilibrium Points in N-person Games. Proc. Nat. Acad. Sci. U.S.A. **36**(1), 48–49 (1950)
5. Von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior, 3rd edn. Princeton University Press, Princeton (1953)
6. Sandholm, T.W., Larson, K.S., Anderson, M.R., Shehory, O., Tohme, F.: Anytime Coalition Structure Generation with Worst Case Guarantees. In: Proceedings of the AAAI 1998, WI, pp. 46–53. AAAI (1998)
7. Shapley, L.S.: Notes on the n-Person Game - II: The Value of an n-Person Game. RAND Corporation, Santa Monica (1951)
8. Bistaffa, F., Farinelli, A., Vinyals, M., Rogers, A.: Decentralised stable coalition formation among energy consumers in the smart grid. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1461–1462 (2012)
9. Blankenburg, B., Klusch, M.: On safe kernel stable coalition forming among agents. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, vol. 2, pp. 580–587. IEEE Computer Society, New York (2004)
10. Yamamoto, J., Sycara, K.: A stable and efficient buyer coalition formation scheme for E-marketplaces. In: Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, Quebec, Canada, pp. 576–583. Association for Computing Machinery, New York (2001)

11. Guthula, S., Simon, S., Karnick, H.: Analysis of agricultural policy recommendations using multi-agent systems. Comput. Res. Repository (CoRR) **1**(1), 1–13 (2020)
12. Zaryouli, M., Fathi, M.T., Ezziyyani, M.: Data collection based on multi-agent modeling for intelligent and precision farming in Lokoss region Morocco. In: 2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), pp. 1–6 (2020)
13. Pérez-Pons, M.E., Alonso, R.S., García, O., Marreiros, G., Corchado, J.M.: Deep Q-learning and preference based multi-agent system for sustainable agricultural market. Sensor **21**(16), 1–16 (2021)
14. Chevalier, A., Copot, C., De Keyser, R., Hernandez, A., Ionescu, C.: A multi agent system for precision agriculture. In: Buşoniu, L., Tamás, L. (eds.) Handling Uncertainty and Networked Structure in Robot Control. SSDC, vol. 42, pp. 361–386. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26327-4_15
15. González-Briones, A., Mezquita, Y., Castellanos-Garzón, J.A., Prieto, J., Corchado, J.M.: Intelligent multi-agent system for water reduction in automotive irrigation processes. Proc. Comput. Sci. **151**, 971–976 (2019)
16. Kahan, J., Rapoport, A.: Theories of Coalition Formation. Lawrence Erlbaum Associates, Hillsdale (1984)
17. Shehory, O., Kraus, S.: Feasible formation of coalitions among autonomous agents in non-super-additive environments. Comput. Intell. **15**(3), 218–251 (1999)