Lecture 8

Representation and complexity

We have studied different solution concepts, mainly looking at their different properties. One natural question from the point of view of a (theoretical) computer scientist is how hard it is to compute a solution, or to check whether the set of solutions is empty or not. This is the question we are going to consider in this lecture

8.1 Naive representation

Let us assume we want to write a computer program for computing a solution concept.

The first question that comes to mind is how to represent the input of a TU game. A straighforward representation is by enumeration: we can use a array, each entry represents a coalition and contains the value of that coalition (and one can use the binary representation of a number to encode which agents are members of a coalition, e.g. $21 = 10101$ corresponds to coalition $\{1, 3, 5\}$). This requires storing 2^n numbers, which may be problematic for large values of n. Typically, computer scientists are made happier with an input of polynomial length.

The complexity of an algorithm is measured in terms of the input size. If we use enumeration, many algorithms may appear good as they manipulate an exponential input. To properly speak about complexity issues, one need to find a polynomial representation of the game, which we will call a compact or succinct representation.

In general, there is a tradeoff between how succinct the representation is and how easy or hard the computation is. The idea is that to represent the game compactly, one needs to encode a lot of information in a smart way, which may make it difficult to manipulate the representation to compute something interesting. Hence we look for a balance between succinctness and tractability.

8.2 Representations that are good for computing the Shapley Value

The nature of the Shapley value is combinatorial, as all possible orderings to form a coalition need to be considered. This computational complexity can sometimes be an advantage as agents cannot benefit from manipulation. For example, it is $\mathcal{NP}$-complete to determine whether an agent can benefit from false names [15]. Nevertheless, some representations allow to compute the Shapley value efficiently, and we are surveying few representations.

8.2.1 Bilateral Shapley Value

In order to reduce the combinatorial complexity of the computation of the Shapley value, Ketchpel introduces the Bilateral Shapley Value (BSV) [13]. The idea is to consider the formation of a coalition as a succession of merging between two coalitions. Two disjoint coalitions $\mathcal{C}_1$ and $\mathcal{C}_2$ with $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$, may merge when $v(\mathcal{C}_1 \cup \mathcal{C}_2) \geq v(\mathcal{C}_1) + v(\mathcal{C}_2)$. When they merge, the two coalitions, called founders of the new coalition $\mathcal{C}_1 \cup \mathcal{C}_2$, share the marginal utility as follows: $BSV(\mathcal{C}_1) = \frac{1}{2}v(\mathcal{C}_1) + \frac{1}{2}\left(v(\mathcal{C}_1 \cup \mathcal{C}_2) - v(\mathcal{C}_2)\right)$ and $BSV(\mathcal{C}_2) = \frac{1}{2}v(\mathcal{C}_2) + \frac{1}{2}\left(v(\mathcal{C}_1 \cup \mathcal{C}_2) - v(\mathcal{C}_1)\right)$. This is the expression of the Shapley value in the case of an environment with two agents. In $\mathcal{C}_1 \cup \mathcal{C}_2$, each of the founders gets half of its 'local' contribution, and half of the marginal utility of the other founder. Given this distribution of the marginal utility, it is rational for $\mathcal{C}_1$ and $\mathcal{C}_2$ to merge if $\forall i \in \{1, 2\}$, $v(\mathcal{C}_i) \leq BSV(\mathcal{C}_i)$. Note that symmetric founders get equal payoff, i.e., for $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}$ such that $\mathcal{C}_1 \cap \mathcal{C}_2 = \mathcal{C}_1 \cap \mathcal{C} = \mathcal{C}_2 \cap \mathcal{C} = \emptyset$, $v(\mathcal{C} \cup \mathcal{C}_1) = v(\mathcal{C} \cup \mathcal{C}_2) \Rightarrow BSV(\mathcal{C} \cup \mathcal{C}_1) = BSV(\mathcal{C} \cup \mathcal{C}_2)$. Given a sequence of successive merges from the states where each agent is in a singleton coalition, we can use a backward induction to compute a stable payoff distribution [14]. Though the computation of the Shapley value requires looking at all of the permutations, the value obtained by using backtracking and the BSV only focuses on a particular set of permutations, but the computation is significantly cheaper.

8.2.2 Weigthed graph games

[8] introduce a class of games called weighted graph games: they define a TU game using an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{W})$ where V is the set of vertices and $W : V \to V$ is the set of edges weights. For $(i, j) \in V^2$, w_{ij} is the weight of the edge between the vertices i and j. The coalitional game (N, v) is defined as follows:

- $N = V$, i.e., each agent corresponds to one vertex of the graph.
- the value of a coalition $\mathcal{C} \subseteq N$ is the sum of the weights between any pairs of members of $\mathcal{C}$, i.e. $v(\mathcal{C}) = \sum_{(i,j) \in \mathcal{C}^2} w_{ij}$.

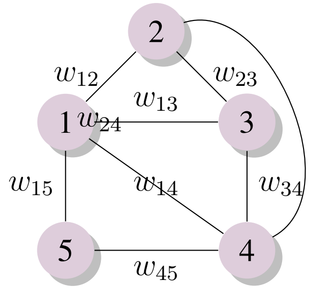

Figure 8.1: Example of a graph with 5 agents

This representation is succinct as we only need to provide n^2 values to represent the entire game. However, it is not a complete representation as some TU games cannot be represented this way (e.g., it is not possible to represent a majority voting game). If we add some restrictions on the weights, we can further guarantee some properties. For example, when all the weights are nonnegative, then the game is convex, and then the game is guaranteed to have a non-empty core.

8.2.1. PROPOSITION. *Let (V, W) be a weighted graph game. If all the weights are nonnegative then the game is convex.*

Proof. Let $S, T \subseteq N$, we want to prove that $v(S) + v(T) \leq v(S \cup T) + v(S \cap T)$.

$$\begin{aligned} v(S) + v(T) &= \sum_{(i,j)\in S^2} w_{ij} + \sum_{(i,j)\in T^2} w_{ij} = \sum_{(i,j)\in S^2 \vee (i,j)\in T^2} w_{ij} + \sum_{(i,j)\in (S\cap T)^2} w_{ij} \\ &\leq \sum_{(i,j)\in (S\cup T)^2} w_{ij} + \sum_{(i,j)\in (S\cap T)^2} w_{ij} = v(S \cup T) + v(S \cap T) \end{aligned}$$

□

One other nice property of this representation is that the Shapley value can be computed in quadratic time, as shown in the following theorem.

8.2.2. THEOREM. *Let (V, W) be a weighted graph game. The Shapley value of an agent i is given by $\Phi_i(N, v) = \sum_{(i,j)\in N^2, i\neq j} w_{ij}$.*

One simple proof of this theorem uses the axioms that define the Shapley value.

Proof. Let (V, W) a weighted graph game. We can view this game as the sum of the $|W|$ games: each game G_{ij} is associated to an edge (i, j) in the graph as follows $G_{ij} = (V, \{w^{ij}\})$ such that $w^{ij}_{kl} = w_{ij}$ if $i = k$ and $j = l$ and 0 otherwise.

For a game G_{ij} corresponding to edge (i, j):

- the agents i and j are substitutes (this is clear).

- all other agents $k \neq i, j$ are dummy agents (this is also clear).

Using the symmetry axiom, we know that $Sh_i(G_{ij}) = Sh_j(G_{ij})$. Then, using the dummy axiom, we also know that $Sh_k(G_{ij}) = 0$. This tells us that $Sh_i(G_{ij}) = \frac{1}{2}w_{ij}$.

Since (V, W) is the sum of all games, by the additivity axiom, we obtain $Sh_k = \sum_{i,j} Sh_k(G_{ij}) = \sum_{k,i} w_{ij}$ □

8.2.3 Multi-issue representation

Conitzer and Sandholm [7] analyse the case where the agents are concerned with multiple independent issues that a coalition can address. For example, performing a task may require multiple abilities, and a coalition may gather agents that work on the same task but with limited or no interactions between them. A characteristic function v can be decomposed over T issues when it is of the form $v(\mathcal{C}) = \sum_{t=1}^{T} v_t(\mathcal{C})$, in which, for each t, (N, v_t) is a TU game.

8.2.3. DEFINITION. [Decomposition]The vector of characteristic functions $\langle v_1, v_2, \ldots, v_T \rangle$, with each $v_i : 2^N \to \mathbb{R}$, is a *decomposition* over T issues of characteristic function $v : 2^N \to \mathbb{R}$ if for any $S \subseteq N$, $v(S) = \sum_{i=1}^{T} v_i(S)$.

Using this idea, we can represent any TU game (we can express a TU game using a single issue).

The Shapley value for agent i for the characteristic function v is the sum of the Shapley values over the t different issues: $\Phi_i(N, v) = \sum_{t=1}^{T} \Phi_i(N, v_t)$. When a small number of agents is concerned about an issue, computing the Shapley value for the particular issue can be cheap. For an issue t, the characteristic function v_t concerns only the agents in $\mathcal{I}_t$ when $\forall \mathcal{C}_1 \in \mathscr{C}, \mathcal{C}_2 \in \mathscr{C}$ such that $\mathcal{I}_t \cap \mathcal{C}_1 = \mathcal{I}_t \cap \mathcal{C}_2 \Rightarrow v_t(\mathcal{C}_1) = v_t(\mathcal{C}_2)$. When the characteristic function v is decomposed over T issues and when $|\mathcal{I}_t|$ agents are concerned about each issue $t \in [1...T]$, computing the Shapley value takes $O(\sum_{t=1}^{T} 2^{|\mathcal{I}_t|})$.

8.2.4 Marginal Contribution Networks (MC-nets)

Ieong and Shoham propose a representation in which the characteristic function is represented by a set of “rules” [11]. A rule is composed by a pattern and a value: the pattern tells which agent must be present or absent from a coalition so that the value of the coalition is increased by the value of the rule. This representation allows to represent any TU game.

More formally, each player is represented by a boolean variable and the characteristic vector of a coalition is treated as a truth assignment. Each “rule” associates a pattern ϕ and a weight $w \in \mathbb{R}$. The pattern ϕ is a formula of propositional logic containing variables in N. A positive literal represents the presence of an agent in a coalition, whereas a negative literal represents the absence of an agent in the coalition. The value of a coalition is the sum over the values of all the rules that apply to the coalition.

8.2.4. DEFINITION. [Rule] Let N be a collection of atomic variables. A *rule* has a syntactic form (ϕ, w) where ϕ is called the pattern and is a boolean formula containing variables in N and w is called the weight, and is a real number.

Let us consider that there are two variables a and b and here are two rules:

- $(a \wedge b, 5)$: each coalition containing both agents a and b increase its value by 5 units.
- $(b, 2)$: each coalition containing b increase its value by 2.

8.2.5. DEFINITION. [Marginal contribution nets (MC-net)] An MC-net consists of a set of rules $\{(p_1, w_1), \dots (pk, w_k)\}$ where the valuation function is given by

$$v(\mathcal{C}) = \sum_{i=1}^{k} p_i(e^{\mathcal{C}})w_i,$$

where $p_i(e^{\mathcal{C}})$ evaluates to 1 if the boolean formula p_i evaluates to true for the truth assignment $e^{\mathcal{C}}$ and 0 otherwise.

The valuation function of the MC net $\{(a \wedge b,\ 5),\ (b, 2)\}$ is the following one:

$$v(\emptyset) = 0 \qquad v(\{b\}) = 2$$
$$v(\{a\}) = 0 \quad v(\{a, b\}) = 5 + 2 = 7$$

We can use negations in rules, and negative weights. Let consider the following MC-net: $\{(a \wedge b,\ 5), (b, 2), (c, 4), (b \wedge \neg c, -2)\}$ In this case, the valuation function is

$$v(\emptyset) = 0 \qquad v(\{b\}) = 2 - 2 = 0 \qquad v(\{a, c\}) = 4$$
$$v(\{a\}) = 0 \quad v(\{a, b\}) = 5 + 2 - 2 = 5 \quad v(\{b, c\}) = 4 + 2 = 6$$

When negative literals are allowed or when the weights can be negative, MC-nets can represent any TU-game, hence this representation is *complete*. When the patterns are limited to conjunctive formula over positive literals and weights are nonnegative, MC-nets can represent all and only convex games (in which case, they are guaranteed to have a non-empty core).

Using this representation and assuming that the patterns are limited to a conjunction of variables, the Shapley value can be computed in time linear to the size of the input (i.e. the number of rules of the MC-net).

8.2.6. THEOREM. *Given a TU game represented by an MC-net limited to conjunctive patterns, the* Shapley value *can be computed in time* linear *in the size of the input.*

Proof. (sketch) we can treat each rule as a game, compute the Shapley value for the rule, and use ADD to sum all the values for the overall game. For a rule, we cannot distinguish the contribution of each agent, by SYM, they must have the same value. It is a bit more complicated when negation occurs (see Ieong and Shoham, 2005). □

8.3 Some references for simple games

The computational complexity of voting and weighted voting games have been studied in [8, 9]. For example, the problem of determining whether the core is empty is polynomial. The argument for this result is the following theorem: the core of a weighted voting game is non-empty iff there exists a veto player. When the core is non-empty, the problem of computing the nucleolus is also polynomial, otherwise, it is an $\mathcal{NP}$-hard problem.

8.4 Some interesting classes of games from the computational point of view

We want to briefly introduce some classes of games that have been studied in the AI literature. Some of these classes of games can be represented more compactly than by using 2^N values, one for each coalition, using an underlying graph structure. In some restricted cases, some solution concepts can be computed efficiently.

minimum cost spanning tree games. A game is $\langle V, s, w\rangle$ where $\langle V, w\rangle$ is as in a graph game and $s \in V$ is the source node. For a coalition $\mathcal{C}$, we denote by $\Gamma(\mathcal{C})$ the minimum cost spanning tree spanning over the set of edges $\mathcal{C} \cup \{s\}$. The value of a coalition $V \setminus \{s\}$ is given by $\sum_{(i,j)\in\Gamma(\mathcal{C})} w_{i,j}$.

This class of game can model the problem of connecting some agents to a central node played by the source node s. Computing the nucleolus or checking whether the core is non-empty can be done in polynomial time.

Network flow games. A flow network $\langle V, E, c, s, t\rangle$ is composed of a directed graph (V, E) with a capacity on the edge $c : V^2 \to \mathbb{R}^+$, a source vertex s and a sink vertex t. A network flow is a function $f : E \to \mathbb{R}^+$ that satisfies the capacity of an edge ($\forall (i, j) \in E$, $f(i, j) \leq c(i, j)$) and that is conserved (except for the source and sink), i.e., the total flow arriving at an edge is equal to the total flow leaving that edge ($\forall j \in V$, $\sum_{(i,j)\in E} f(i, j) = \sum_{(j,k)\in E} f(j, k)$). The value of the flow is the amount flowing out of the sink node.

In network flow game [12], $\langle V, E, c, s, t\rangle$, the value of a coalition $\mathcal{C} \subseteq N$ is the maximum value of the flow going through the flow network $\langle \mathcal{C}, E, c, s, t\rangle$.

This class of games can model a situation where some cities share a supply of water or some electricity network. [12] proved that a network flow game is balanced, hence it has a non-empty core. [3] study a threshold version of the game and the complexity of computing power indices.

Affinity games. The class of *affinity games* is a class of hedonic games introduced in [5, 6]. An affinity game is defined using a directed weigthed graph $\langle V, E, w\rangle$ where V is the set of agents, E is the set of directed edges and $w : E \to \mathbb{R}$ is the weight of the edges. $w(i, j)$ is the value of agent i when it is associated with agent j. The value of agent i for coalition $\mathcal{C}$ is $v_i(\mathcal{C}) = \sum_{j\in\mathcal{C}} w(i, j)$.

Some special classes of affinity games have a non-empty core (e.g. when the weights are all positive or all negative). In this games, there may be a trade-off between stability and efficiency (in the sense of maximizing social welfare) as the ratio between an optimal CS and a stable CS may be infinite.

Skill games. This class of games, introduced by [4] is represented by a triplet $\langle N, S, T, u \rangle$ where N is the set of agents, S is the set of skills, T is the set of tasks, and $u : 2^T \rightarrow \mathbb{R}$ provides a value to each set of tasks that is completed. Each agent i has a set of skills $S(i) \subseteq S$, each task t_i requires a set of skills $S(t_i) \subseteq S$. A coalition $\mathcal{C}$ can perform a task t when each skill needed for the task is the skill of at least a member of $\mathcal{C}$ (i.e. $\forall s \in S(t)$, $\exists i \in \mathcal{C}$ such that $S(i) = s$). The value of a coalition $\mathcal{C}$ is $u(T_\mathcal{C})$ where $T_\mathcal{C}$ is the set of tasks that can be performed by $\mathcal{C}$.

This representation is exponential in the number of agents, but variants of the representation lead to polynomial representation. For example when the value of a coalition is the number of tasks it can accomplish, or when each task has a weight and the value of a coalition is the sum of the weights of the accomplished tasks. In general, computing the solution concepts with these polynomial representation is hard. However, in some special cases, checking whether the core is empty or computing an element of the core can be performed in polynomial time. The problem of finding an optimal CS is studied in [2].

Some more papers are studying the computational complexity of some subclasses of games, e.g. in [1, 10] to name a few. We do not want to provide a full account of complexity problem in cooperative games as it could be the topic of half a course.

Bibliography

[1] Haris Aziz, Felix Brandt, and Paul Harrenstein. Monotone cooperative games and their threshold versions. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1 - Volume 1*, AAMAS '10, pages 1107–1114, Richland, SC, 2010. International Foundation for Autonomous Agents and Multiagent Systems.

[2] Yoram Bachrach, Reshef Meir, Kyomin Jung, and Pushmeet Kohli. Coalitional structure generation in skill games. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 703–708, July 2010.

[3] Yoram Bachrach and Jeffrey Rosenschein. Power in threshold network flow games. *Autonomous Agents and Multi-Agent Systems*, 18:106–132, 2009. 10.1007/s10458-008-9057-6.

[4] Yoram Bachrach and Jeffrey S. Rosenschein. Coalitional skill games. In *Proc. of the 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS-08)*, pages 1023–1030, May 2008.

[5] Simina Brânzei and Kate Larson. Coalitional affinity games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 1319–1320, May 2009.

[6] Simina Brânzei and Kate Larson. Coalitional affinity games. In *Proc of the 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, pages 79–84, July 2009.

[7] Vincent Conitzer and Tuomas Sandholm. Computing shapley values, manipulating value division schemes, and checking core membership in multi-issue domains. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 219–225, 2004.

[8] Xiaotie Deng and C H Papadimitriou. On the complexity of cooperative solution concetps. *Mathematical Operation Research*, 19(2):257–266, 1994.

[9] Edith Elkind, Leslie Ann Goldberg, Paul Goldberg, and Michael Wooldridge. Computational complexity of weighted threshold games. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-07)*, pages 718–723, 2007.

[10] Gianluigi Greco, Enrico Malizia, Luigi Palopoli, and Francesco Scarcello. On the complexity of compact coalitional games. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 147–152, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[11] Samuel Ieong and Yoav Shoham. Marginal contribution nets: a compact representation scheme for coalitional games. In *EC '05: Proceedings of the 6th ACM conference on Electronic commerce*, pages 193–202, New York, NY, USA, 2005. ACM Press.

[12] E. Kalai and E. Xemel. Totally balanced games and games of flow. *Mathematics of Operations Research*, 7:476–478, 1982.

[13] Steven P. Ketchpel. The formation of coalitions among self-interested agents. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 414–419, August 1994.

[14] Matthias Klusch and Onn Shehory. Coalition formation among rational information agents. In Rudy van Hoe, editor, *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[15] Makoto Yokoo, Vincent Conitzer, Tuomas Sandholm, Naoki Ohta, and Atsushi Iwasaki. Coalitional games in open anonymous environments. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pages 509–515. AAAI Press AAAI Press / The MIT Press, 2005.