Technical University of Crete

Thesis

Employing Hypergraphs for Efficient Coalition Formation with an Application to Electric Vehicle Cooperatives

Author:
Filippos Christianos

Committee:
Associate Professor Georgios Chalkiadakis (Supervisor)
Associate Professor Michail Lagoudakis
Associate Professor Eftichios Koutroulis

A thesis submitted in partial fulfillment of the requirements for the diploma of Electrical and Computer Engineer

in the

School of Electrical and Computer Engineering
Technical University of Crete

February 24, 2017

TECHNICAL UNIVERSITY OF CRETE

School of Electrical and Computer Engineering

Employing Hypergraphs for Efficient Coalition Formation with an Application to Electric Vehicle Cooperatives

by Filippos Christianos

Abstract

This thesis proposes, for the first time in the literature, the use of *hypergraphs* for the efficient formation of effective agent coalitions. We put forward several formation methods that build on existing hypergraph pruning, transversal, clustering and hybrid algorithms, and exploit the hypergraph structure to identify agents with desirable characteristics. Our approach allows the near-instantaneous formation of high quality coalitions, adhering to multiple stated quality requirements. Moreover, our methods are shown to scale to *dozens of thousands* of agents within fractions of a second; with one of them scaling to even *millions* of agents within seconds. We apply our approach to the problem of forming coalitions to provide *(electric) vehicle-to-grid (V2G)* services. Ours is the first approach able to deal with *large-scale*, *real-time* coalition formation for the V2G problem, while taking *multiple criteria* into account for creating the electric vehicle coalitions. A sketch of these ideas appeared originally in a short paper in the 22nd European Conference on Artificial Intelligence (ECAI-2016). Afterwards, a full paper describing our work was published in the 14th European Conference on Multi-Agent Systems (EUMAS-2016).

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Χρήση Υπεργράφων για την Αποτελεσματική Δημιουργία Συνασπισμών με Εφαρμογή σε Συνεταιρισμούς Ηλεκτρικών Οχημάτων

Φίλιππος Χριστιανός

Περίληψη

Αυτή η διπλωματική εισάγει, για πρώτη φορά στη βιβλιογραφία, την χρήση υπεργράφων για την ταχεία δημιουργία αποτελεσματικών συνασπισμών αυτόνομων πρακτόρων. Προτείνουμε ορισμένες μεθόδους σχηματισμού, που βασίζονται σε υπάρχοντες αλγορίθμους υπεργράφων, όπως οι pruning, transversal, clustering και hybrid, και εκμεταλλευόμαστε την δομή του υπεργράφου για να εντοπίσουμε πράκτορες με επιθυμητά χαρακτηριστικά. Η προσέγγισή μας επιτρέπει τον σχεδόν στιγμιαίο σχηματισμό συνασπισμών υψηλής ποιότητας, ικανοποιώντας πολλαπλές ποιοτικές απαιτήσεις. Επιπλέον, οι μέθοδοί μας κλιμακώνονται ώστε να δέχονται δεκάδες χιλιάδες πράκτορες ως είσοδο και να εμφανίζουν τα αποτελέσματα μέσα σε κλάσματα του δευτερολέπτου, με μια από αυτές να λειτουργεί με εκατομμύρια πράκτορες μέσα σε δευτερόλεπτα. Εφαρμόζουμε την προσέγγισή μας στο πρόβλημα της δημιουργίας συνασπισμών για την παροχή ρεύματος από ηλεκτρικά οχήματα προς το ηλεκτρικό δίκτυο (το λεγόμενο πρόβλημα *Vehicle-to-Grid*, ή *V2G*). Η προσέγγισή μας είναι η πρώτη που είναι σε θέση να ασχοληθεί με μεγάλης κλίμακας, και σε πραγματικό χρόνο σχηματισμό συνασπισμών για το πρόβλημα V2G, λαμβάνοντας υπ΄όψιν πολλαπλά κριτήρια για τη δημιουργία των συνασπισμών ηλεκτρικών οχημάτων. Ένα προσχέδιο των ιδεών αυτών εμφανίστηκε αρχικά σε μια σύντομη δημοσίευση στο 22ο *European Conference on Artificial Intelligence (ECAI-2016)* και έπειτα σε μια πλήρη στο 14ο *European Conference on Multi-Agent Systems (EUMAS-2016)*.

Acknowledgements

I would like to thank all the people who contributed in some way to the work described in this thesis. First and foremost, I would like to express my gratitude to my supervisor Georgios Chalkiadakis for his patience, motivation and knowledge. His guidance helped me immensely during the research and writing of this thesis.

Besides my advisor, I would like to thank Stefanos, Dimitris, Tonia and Panagiotis for being great friends and offering me invaluable support throughout these years. I would like to especially thank Dafne for all her love, support and helpful advice.

To my family, thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams. I am especially grateful to my parents, Rania and Vassilis, who supported me emotionally and financially and always wanted the best for me. This thesis is dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Coalition formation (CF) is a paradigm widely studied in multiagent systems and economics, as means of forming teams of autonomous, rational agents working towards a common goal [7]. *Game theory*, the study of strategies involved in interaction between intelligent rational agents with the goal of maximizing their rewards is also connected to *coalition formation*. Specifically, cooperative game theory allows players to form *coalitions* and achieve rewards through cooperation[6]. Coalition formation can be used in many real-life problems, such as improving the Smart Grid, and thus, is an active area of study in multiagent systems (MAS).

One domain where the formation of coalitions comes naturally into play is the so-called *vehicle-to-grid (V2G)* problem. In V2G, battery-equipped *electric vehicles (EVs)* communicate and strike deals with the electricity grid in order to either lower their power demands or return power to the network when there is a peak in the request for power. This helps the grid to maintain a balanced power load [31]. G2V is V2G's "sister" problem, where EVs connect and draw power from the Grid without overloading it [34]. In both cases, the coordination of EVs efforts, is essential.

To elaborate, several recent approaches have called for the formation of EV coalitions in order to tackle the V2G problem [29, 21, 20]. The existing approaches, however, typically exhibit the following characteristics: *(a)* they attempt to form *optimal* coalitions or coalition structures; and *(b)* they either attempt to form coalitions with respect to a single criterion, or employ lengthy negotiation protocols in order to capture various coalitional requirements while respecting the constraints of individual agents.

The inherent hardness of the optimal coalition structure generation problem [30], however, along with the fact that negotiation protocols can be lengthy and thus highly time-consuming, can severely restrict the practicality and scalability of such algorithms: existing algorithms can handle at most a few hundred EVs. In reality though, there exist hundreds of thousands of EVs that connect to the grid and that could potentially offer their services. Any formed coalition would be required to possess *a multitude of desirable characteristics* high collective storage capacity and high collective discharge rate, and so on; and, if the aim is to balance the electricity demand in real-time, any such service should be offered by the appropriate coalition almost instantaneously.

In this thesis, we overcome the aforementioned difficulties by employing, for the first time in the literature, *hypergraphs* to achieve the timely formation of coalitions that satisfy *multiple criteria*. In our approach, EV agents that share specific characteristics are organised into *hyperedges*. Then, building on the existing hypergraphs literature [13, 39], we propose algorithms for *(i)* hypergraph *pruning*, to focus on interesting parts of the search space; *(ii)* hypergraph *transversal* to identify sets of vertices (agents) that combine several desirable characteristics; and *(iii)* hypegraph *clustering*, that allows the identification of clusters of high quality agents. Moreover, we put forward *(iv)* a heuristic formation algorithm that benefits from pruning and generates high quality coalitions near-instantaneously, while scaling linearly with the number of agents.

In contrast to existing approaches, we do not attempt to generate an optimal coalition structure, nor do we attempt to compute a single optimal coalition. Instead, we exploit the hypergraph representation of our problem in order to select agents and form highly effective coalitions, while being able to scale to *dozens of thousands* of agents within fractions of a second; and, in the case of our heuristic method, even to *millions* of EV agents, within a few seconds.

Though here we apply it to the V2G problem, our approach is generic and can be used in *any* coalition formation setting. It is perhaps surprising that a powerful model like hypergraphs has not been so far exploited for devising efficient coalition formation methods, despite its intuitive connections to the concept of coalitions. Regardless, we are not aware of any work to date that has exploited hypergraphs and related algorithms in order to perform *real-time*, *large-scale*, *multi-criteria* coalition formation, as we do in this thesis.

A sketch of these ideas appeared originally in a short paper in the ECAI-2016 [11]. Afterwards, a full paper describing our work was published in the EUMAS-2016 [10].

1.1 Thesis Outline.

The rest of the thesis is structured as follows. Chapters 2 and 3 introduce concepts as the Smart Grid, electric vehicles and coalition formation. Furthermore, we provide background information and present works related to our research. Chapter 4 presents our approach and the transversal, clustering, heuristic and hybrid methods used (Sec. 4.3, 4.4, 4.5, 4.6) to solve the aforementioned problem. Finally, Chapter 5 presents our experimental results while Chapter 6 concludes and outlines future works.

Chapter 2

Background

This chapter presents some background for the research presented in this thesis. Specifically, section 2.1 starts with a definition and overview of the Smart Grid and connects it to electric vehicles. In section 2.2 we discuss coalition formation and finally, in section 2.3 we present background information on hypergraphs.

2.1 Electric Vehicles in the Smart Grid

The current electricity Grid, is the network that delivers power to consumers. It uses large central power stations that distribute the energy through high capacity power lines to both industrial and domestic areas. Historically the Grid handled peak hours poorly, with blackouts and power cuts being common [16]. Only more recently and after establishing patterns in electricity demands, could the daily peaks be met, using part-time generators (usually the expensive gas turbines). The current structure of the Grid is a product of an evolutionary process that lasted decades, connected to growing needs of consumers. Thus, the current infrastructure was not planned as a whole, but rather extended several times, creating weak links and containing outdated designs. One of the most important issues, is the centralized nature of the Grid. It is built around large power plants producing all the energy required by the consumers. With the growth of smaller, usually renewable, energy producers, the Grid by necessity has to move to a less centralized and more interactive structure.

The Smart Grid, therefore, is a modernized electricity Grid that collects and uses information to improve efficiency, reliability, economics and sustainability of the electrical Grid. It is also planned to be more decentralized making efficient use of small scale producers and prosumers. [1] Thus, the Smart Grid can potentially become much more reliable than its "classic" counterpart by eliminating single points of failure. Another important characteristic of the Smart Grid is that it is much more interactive. Everyone connected, communicates and coordinates with the Grid, rendering the consumption and production more balanced. Connected consumers also implement smart technologies that drive their own consumption down. With techniques like the ones mentioned above, the network's energy load is balanced and thus the distribution is more efficient - eliminating, if possible, high-cost energy producers like gas turbines. To understand the efficiency of the Smart Grid, experiments have, for example, shown that just a small scale coordination of a few battery-equipped houses can lower everyone's electric bill [37].

Electric vehicles (EVs) are a promising new concept for the automotive industry. EVs use energy stored in a battery and electric motors to generate propulsion. Electricity offers many advantages against petrol-powered vehicles. Specifically, EVs are cost effective, require less maintenance, and have no direct emissions since they run in electricity powered engines. In their current state, the batteries of electric vehicles (which rapidly become even more efficient and cost effective [28]) are capable of at least 300km [15][38] of range. [2] To achieve this range, the batteries have a large capacity usually in the 60kWh-100kWh range. To put this into perspective, batteries as low as 4kWh can have a significant impact on the energy footprint of a typical household [37].

Another important factor for the battery is the discharge rate. A vehicle requires a large amount of energy during acceleration. For example, accelerating a typical vehicle to 100km/h in 10 seconds can require up to 65kW of power [38]. Since EV batteries are actually designed for discharging at these rates for typical driving, we can safely assume that we can use this discharge rate for other uses.

[1] Prosumer is a small scale electricity consumer that might also produce energy[25].

[2] The range of an EV is defined as the driving range using power only from its battery pack during a single charge.

Charging the batteries is another characteristic that must be accounted for. Currently, charging the battery takes a few hours depending on the battery's state of charge (Soc). Nevertheless for an everyday use scenario, a battery can be expected to charge (using fast charging) to a reasonable amount in half an hour [38]. In this thesis we will not examine how EVs charge, or how we can regulate its charging.

Due to the previously mentioned growth of EVs and their energy capacity we can safely assume that they will play a significant role in the future of the electricity Grid [31]. As a result, two categories of problems arise. First, the issue of how we can successfully provide the energy those vehicles need, and charge them without overloading the Grid. The second is how energy stored in EVs can be used to balance out peaks in consumption or even serve as backup power. Those categories are called Grid to Vehicle (G2V) and Vehicle to Grid (V2G) [27] [23] respectively.

G2V is better explained by noticing that, due to the common working hours, large numbers of EV owners might return home and charge their vehicles, at about the same time. Since EVs can draw a huge amount of power the Grid will overload due to huge spikes on consumption. Nevertheless the charging could have been coordinated and EVs charged during the night, without causing spikes. Finding an optimal way to charge EVs this way though is quite complicated, since possibly millions of batteries have to eventually be charged. Several attempts have been made to tackle the problem [14][34] but are not usually scalable to large numbers of EVs.

V2G, a problem related to our work here, in contrast to G2V, is the question of how EVs can supply power (usually stored in the vehicle's battery) to the Smart Grid during power peaks. This can lower or even eliminate the need of expensive back up generators. Since the batteries can charge when power is cheap (e.g. at night) and return the power when its more expensive, this raises an opportunity for profit for EV owners. Nevertheless this does not come without several issues to be addresses. Specifically, a single EV must know if its owner will need the energy that will be sent to the Grid and not participate in an exchange if there's a chance the owner needs to use the vehicle. In addition, EVs can be used by owners without any previous notice and might be unplugged from the electricity Grid at any moment.

This raises the issue of reliability: how certain we are that a vehicle that promises to deliver power during a timeslot, can actually keep its promise. Finally, while the batteries can store and provide a respectable amount of energy, the needs of the Grid are proportionally much greater. Thus EVs must be able to cooperate to provide sufficient and reliable services. Several aspects of V2G have been researched. We will mention several such attempts in Chapter 3.

2.2 Coalition Formation

Coalition Formation deals with how agents can form one or more groups, called coalitions, that can tackle a common problem. CF theory analyzes several of its aspects, that range from creating such coalitions to fairly dividing rewards to the members of a coalition. Individual agents usually have different degrees of efficiency. Thus, we must form groups of agents with characteristics that compliment each other and exploit their individual strengths[33]. As discussed in [32], coalition formation has three activities. *Coalition structure generation* (CSG), is the first of these activities, namely the partitioning of the set of agents into mutually disjoint coalitions (or groups), in a way that the resulting coalitions maximize the sum of the rewards of all agents (known as social welfare) [30]. Next is *the optimizations problem* of each coalition, that tries to maximize the rewards from outside the coalition and optimize the allocation of resources and tasks between agents of the respective coalition. The last activity of CF is the *division of the rewards* among agents. This must be done in such a way that the rewards are fair, and no agent can be motivated to leave his coalition.

Finding the optimal coalition structure is generally computationally expensive, especially in a large set of agents, and the computational requirements grow exponentially. As such, finding in a reasonable time a CS that is within a bound of the optimal one, is also hard problem. There are several attempts to solve this problem [32] [30].

Nevertheless, in this thesis we will not attempt to solve the CSG problem. Instead we will focus on finding a simple coalition that is able to perform a specific task. Such a coalition will be created by selecting agents from an extensive set, in a way that the result can efficiently handle the appointed task. In addition each agent will have several attributes that contribute in different ways to the completion of the goal. We will not be using a utility function (that ultimately combines the attributes, thus losing accuracy within its particular dimensions).

By contrast, in essence we will be tackling what we call *multi-criteria* coalition formation, that is, the problem of forming coalitions that can accomplish a task that requires meeting a range of task-related goals: for instance, offering a minimal value of charging capacity, a minimal value of charging rate and so on. Due to the nature of this problem, the results cannot be easily evaluated. There is possibly a great number of possible coalitions with similar capacity to handle the task. In addition, since the agent set is magnitudes larger that what usual optimal CF algorithms can handle we cannot find how close to optimality our solution is. Thus we will focus on creating coalitions that can complete the task efficiently and can be generated in a minimal amount of time.

This problem can be quite natural in todays world. There are several real world examples where efficiency is sacrificed for performance. In this thesis, we will present a way to form an EV coalition in just a few seconds, able to fulfill the energy requirements of the Smart Grid.

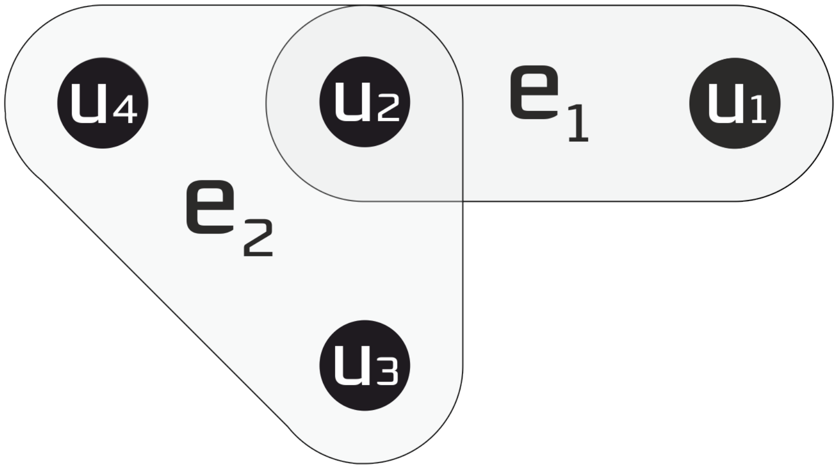

FIGURE 2.1: A Simple Hypergraph

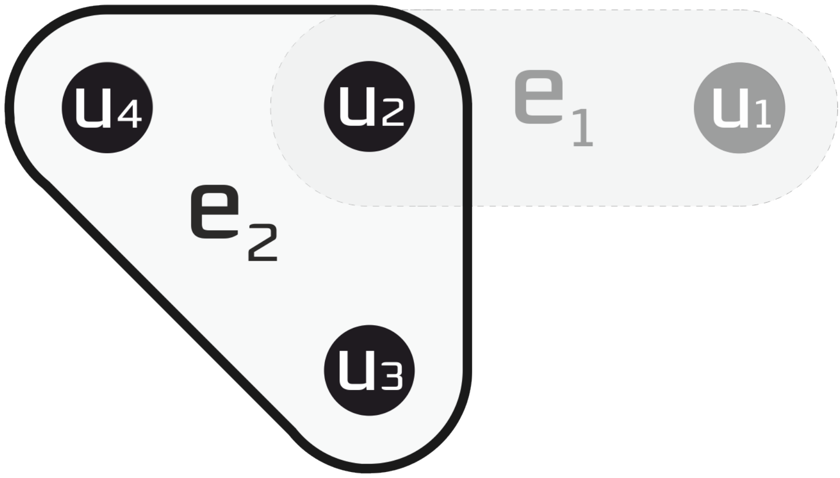

FIGURE 2.2: A Single Hyperedge

2.3 Hypergraphs

The Hypergraph, is a generalization of a graph. In contrast to a simple graph, hypergraph's edges can connect multiple vertices. We formally define a hypergraph as $H = (V, E)$ where V is a set of unique *vertices* or *nodes* and E is the set of *edges* or *hyperedges*.

A simple hypergraph can be seen in figure 2.1. In this example, the vertices of the hypergraph are $V = \{u_1, u_2, u_3, u_4\}$ and the edges $E = \{e_1, e_2\}$. Edge e_1 contains vertices u_1, u_2, while e_2 contains u_2, u_3, u_4.

The difference of graphs and hypergraphs is outlined in the figure of the hyperedge e_2 2.2. As shown, a hyperedge can connect multiple vertices, in contrast to simple graphs where an edge always connects two nodes.

Hypergraphs are well studied, and thus there are many definitions in literature which can help us use them as a data structure. First, a *transversal*, or *hitting set*, of a hypergraph is a set of nodes $T \subset V$ such that T intersects with any edge E of the hypergraph. A *hitting set* that does not contain any other hitting set is called *minimal*. This is better illustrated in Fig. 2.3. As shown, the hypergraph has three *minimal transversals*: $T_1 = \{u_1, u_4\}, T_2 = \{u_1, u_3\}$ and $T_3 = \{u_2\}$. Any other transversal such as $T_4 = \{u_2, u_4\}$ is not minimal since $T_3 \subset T_4$ (it contains at least one other transversal). Finally, the *set of minimal transversals* is called the *dual* of a hypergraph or *minimal set-cover* and denoted as H^d.

We expect this problem to be of high complexity since it is essentially the set-cover problem extended to hypergraphs. The search version of the set-cover problem in graphs is NP-hard. However, computing the dual hypergrah is a problem widely studied, and hence, many algorithms offer efficient solutions. To begin, Berge's algorithm [2], while slower than the rest, is foundation of many algorithms which are merely an improvement on it. Specifically, the algorithms described here, are divided in two types: improvements over Berge and hill-climbing algorithms. The Berge algorithm updates the set of minimal transversals by iteratively adding hyperedges.

Dong and Li [12], for instance, is a Berge-based algorithm that improves upon it. This method decreases the search space by avoiding to generate several non-minimal transversals. Bailey *et al.* [1] algorithm, starts with a limited vertex set and update both hyperedges and hitting sets by constantly adding new vertices. Kavvadias *et al.* [22] propose a memory-bound depth-first approach which generates a constant stream of minimal hitting sets. Nevertheless this method does not come with a time complexity bound. Khachiyan *et al.* [4] provide a *quasi-polynomial* algorithm for enumerating all minimal transversals. In a sense of time complexity this is the fastest of the algorithms presented here, but in practice it falls behind in several test-cases. Extending on that, another publication [24] finds a multi-threaded solution with excellent time complexity assuming multiple cores. Finally [18], a solution not based on Berge, offers a hill-climbing algorithm that adds vertices in increasing order and checks if they satisfy a minimal transversal condition.

Another well-studied hypergraph-related problem is hypergraphs clustering. This is done by regarding the edges as node attributes and using

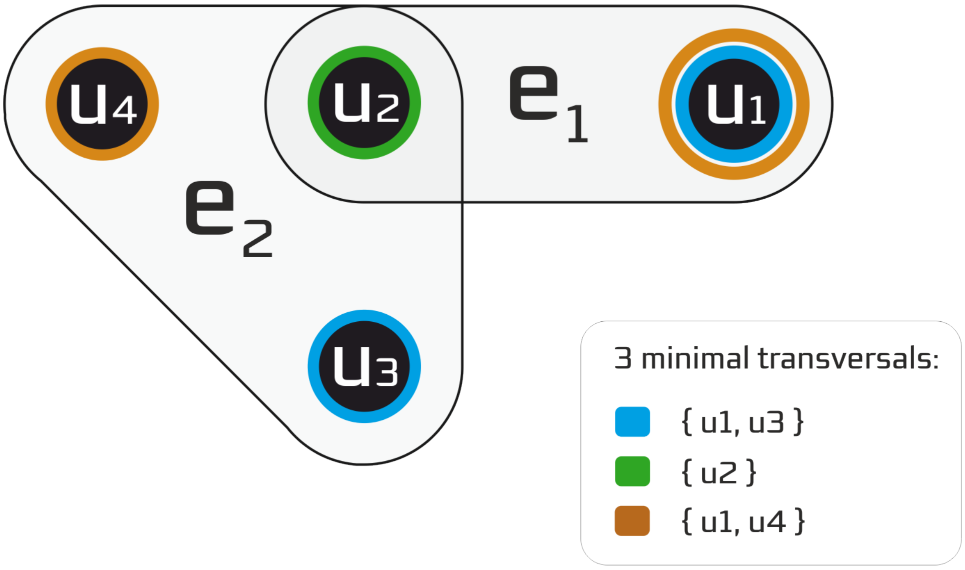

FIGURE 2.3: Color-coded Transversals

several techniques to cluster nodes with similar attributes together. For instance [39], proposes powerful methods of spectral clustering on hypergraphs and algorithms for classification and embedding. The methods proposed had a significant advantage when used in hypergraphs over simple graphs, since they managed to store complex relationships among objects on the hyperedges. A game theoretic approach to hypergraph clustering is found in [5]. Specifically, there the cluster is treated as the game-theoretic concept of equilibrium, and the problem of partitioning the hypergraph to clusters as non-cooperative multiplayer game. This has several advantages over classical approaches, e.g. the final number of clusters is not needed beforehand. Finally, Leordeanu and Sminchisescu [26] propose an efficient clustering method that updates which vertices correspond to which clusters in parallel through an iterative procedure. This manages to reduce computing requirements significantly.

In addition, there are also many ways for matrices to represent a hypergraph or specific attributes of it. For instance the incidence matrix, weight matrix and adjacency matrix are all easily defined, as we show in Section 4.4.

Hypergraphs thus, are a powerful and well-defined way to store information. For example we can easily store (better explained in Chapter 4) EVs in hyperedges that represent a specific quality. Edges then store both attributes of EVs and, possibly, complex relations between them. This comes with advantages like the instant selection of interesting parts of the graph or fast set operations like *intersection* or *union*.

Later, in Chapter 4 we explain how these advantages and the existing literature is exploited for efficient coalition formation.

Chapter 3

Related Work

Here we review related work mainly on the V2G and the SCG problems, and highlight its differences to our approach in this thesis. To begin, in their pioneer work, Valogianni *et al.* [34] propose an *adaptive smart charging algorithm* that adjusts *the power drawn* from the electrical Grid for charging EVs, based on each EV owner's utility from charging. backbone of the approach is based on a *reinforcement learning* for capturing agent needs and behavior. It also utilizes an optimization module that schedules the charging of each EV in order to maximize its utility, subject to network constraints. Though effective, this work fails to focus on the problem of feeding the network with power drawn from EVs in a coordinated fashion. As such EV coalitions and their potential are not being considered in this work.

Contrary to the aforementioned line of research, the work presented in [21] considers an attempt to exploit EV coalition formation in energy exchange. In particular in [21] EV coalitions are utilized in selling power in the regulation market. In more detail, EV coalitions provide the following service to the Grid every few seconds: they, either, (i) *scale down* their power draw (or discharge); or they (ii) *scale it up*, and request more power from the Smart electrical Grid. Despite the effectiveness of this approach, there are considerable limitations with respect to its practical application that renders its usability in real setting scenario, questionable. In more detail there is a need for a complicated and resource-consuming EV selection process by an aggregator agent. Moreover, in this context, and to limit the respective complexity, the simulations involved in this work considered a limited pool of three hundred vehicles only.

That said, the potential of coalition formation is not only exploited in the narrow context of EVs. In more detail, coalition formation has long been investigated to provide regulation services to the Smart Grid and, in recent years several works in this direction emerged. For instance the work in [35] adapts a game-theoretic perspective on the formation of coalitions in the Smart Grid. In this context, it considers the optimal coalition structure generation problem (CSG). To this end, it utilizes an approach of forming *virtual energy consumer* coalitions. In the context of these coalitions, it manages to flatten the energy demand. This, in turn, enhances the negotiational ability with the Grid, enabling better prices in what could be a G2V arrangement. In more detail, the solution of the CSG, provides the best VEC for every consumer on the market; and guarantees a core-stable payoff distribution outcome. Nevertheless, the computational complexity of this approach renders it impractical in real settings. In particular, this work has been shown to perform adequately on social graphs of limited size (with only a handful of agents). Notably, against this background, our approach manages to produce high quality solutions in milliseconds, and scales to the number of millions (as further discussed in Chapter 5).

Now, two recent papers which study *cooperative games* defined *over graphs* that impose constraints on the formation of the coalitions, are [8] and [9]. Specifically, they assume that the environment possesses some structure that forbids the creation of individual coalitions, due to limited resources and existing physical or even legal barriers. This is captured by an undirected graph providing a path connecting any two agents that can belong to the same coalition. Both of these papers, however, do not employ hypergraphs in any way. Hypergraphs have in fact been used for modelling agent interactions in cooperative game settings, where agents can simultaneously belong to multiple coalitions [19] [40]. Now, several papers [8] [9] [19] [40] focus on studying the theoretical problem of achieving *coalitional stability* via appropriately distributing the payoff among the agents. This is done rather than providing algorithms for large-scale coalition formation in real-world settings, as we do in this work.

By contrast, two papers that study the generation of optimal coalition structures while focusing on stability are [3] [36]. They focus on the use of synergy graphs. Those graphs connect agents with edges that represent a vital synergistic link, such as communication, trust or physical constrains. They propose efficient ways to generate all possible coalitions and find the optimal coalition structure. Although these approaches scale to thousands of agents they are limited in terms of scalability compared to our approach which scales to millions of users. Furthermore, their approach fails to tackle multiple formation criteria.

A paper that is more related to our work here, in the sense that it exploits constraints among vehicles for coalition formation, is the work of Ramos *et al.* [29]. In this context, they propose the dynamic formation of coalitions among EVs so that they can function as *virtual power plants* that sell power to the Grid as an aggregate. The method relies heavily on a inter-agent negotiations protocol. However, that work also attempts to tackle the optimal CSG problem and hence suffers from high complexity and scalability issues. As such, although it is empirically shown to produce solutions that are close to optimal (98%), this is only when tested in scenarios with a few dozens of agents. In addition, the work in [29] considers only a single criterion for the formation of a coalition—namely, the *physical distance* among the EVs. The physical distance, however, is not a very natural criterion; and, in any case, it is imperative that a multitude of criteria is taken into account—such as capacity, discharge power, and perceived reliability (see, e.g., [20]). Our approach, in contrast, is able to take into account any number of natural criteria to form EV coalitions.

Chapter 4

Our Approach

In order to develop multi-criteria coalition formation algorithms that generate coalitions efficiently, we employ the concept of *a hypergraph*. A hypergraph $H = (V, E)$ is a generalization of a graph, where each *hyperedge* $e \in E$ can contain any number of *vertices (or nodes)* in the set V.

Vertices in H correspond to agents; while we view a hyperedge as corresponding to some particular *attribute* or *characteristic* possessed by the agents in the hyperedge. In the V2G setting, the agents correspond to EVs (i.e., an EV is represented by a node in our hypergraph); while the hyperedges correspond to vehicle characteristics. More specifically, a hyperedge corresponds to a "quality level" of some EV attribute, as we explain below.

In order to represent the different *quality* of the various hyperedges, and utilize it in our algorithms, we mark each hyperedge with a weight.[1] These weights define the *degree* of a node: *The degree* $deg(u)$ *of a node* u *is the sum of the weights of its edges*. Intuitively, *a high degree node is a high quality one*. This fact is exploited in our algorithms below. A hyperedge (of a given quality) will be also called a *category*. The (quality of the) categories to which an EV belongs will be influencing the decisions of our *hypergraph pruning* algorithm, which we describe in Section 4.2 below. A node that belongs to a hyperedge characterizing the quality of a given agent attribute, cannot belong to some other hyperedge characterizing the quality of the same attribute.

To illustrate the use of hypergraphs in our setting, consider for example

[1] In our implementation, the weight of the edges, according to the quality of each attribute(capacity, reliability and discharge), are as follows: {*extremely-high: 8, very-high: 7, high: 6, medium-high: 5, medium-low: 4, low: 3,very-low: 2, extremely-low:1*}. Thus we have 24 edges + 1 containing commitment of EVs.

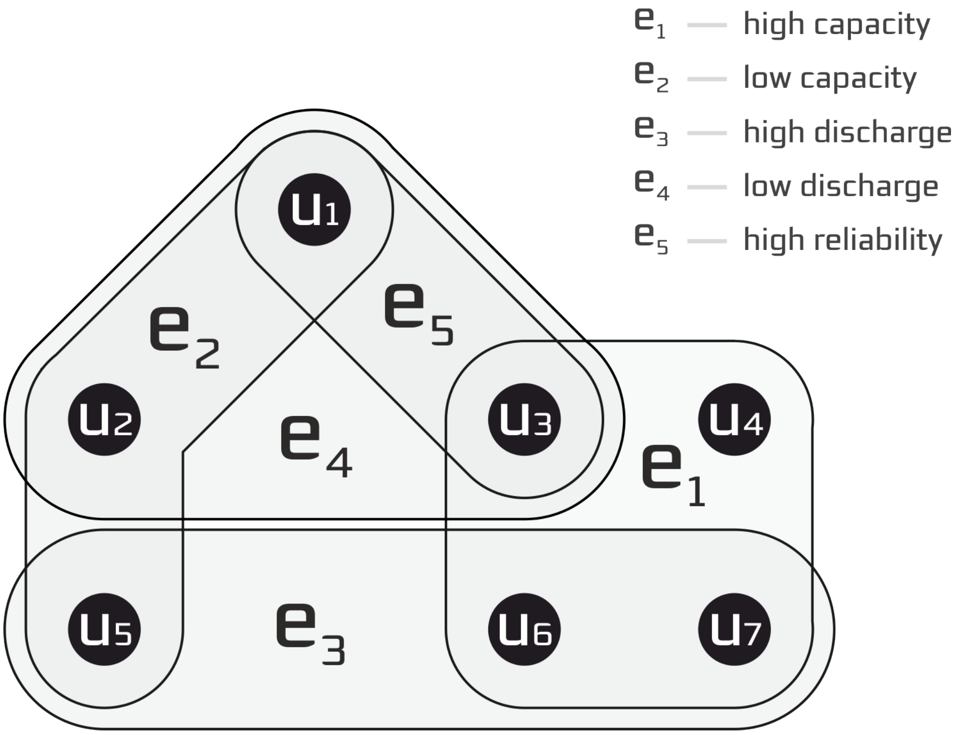

FIGURE 4.1: Storing EVs in a hypergraph

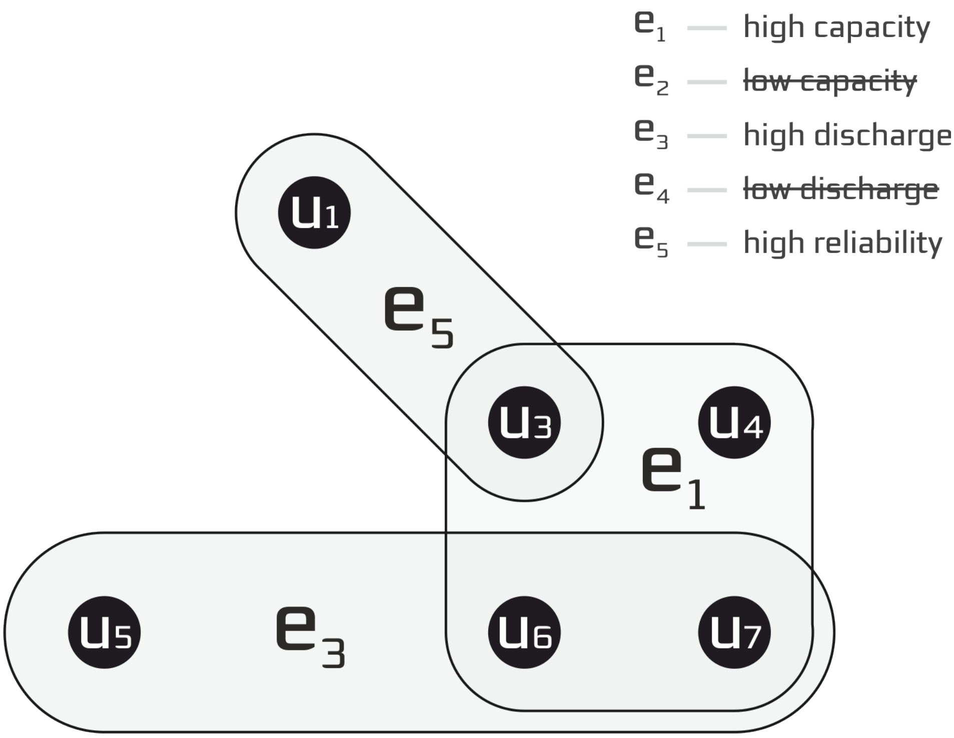

FIGURE 4.2: Pruning the hypergraph

the hypergraph of Fig. 4.1, which contains the hyperedges $e_{1 \ldots 6}$ and vertices $u_{1 \ldots 7}$. It is clear in this example that vertices may belong to multiple hyperedges: the hyperedge e_1 contains the vertices $u_{3,4,6,7}$, while the vertex u_1 belongs in the hyperedges e_2, e_5, e_4. Vertices in Fig. 4.1 correspond to EVs; while the hyperedges correspond to the "quality" of the following EV attributes: *capacity*, *discharge rate* and *observed reliability*. The meaning of these attributes is intuitively straightforward, but will be nevertheless explained in Section 4.1 below. Each attribute is related to at least one hyperedge in the hypergraph. For instance, in Fig. 4.1, the *capacity* attribute is represented by three hyperedges in the hypergraph: *low-capacity*, *medium-capacity*, and *high-capacity*. As noted above, no node can belong in more than one capacity-related hyperedges. In our figure,

- the hyperedge $e1$ represents the nodes which have high capacity;
- the hyperedge $e2$ contains nodes that have low capacity;
- $e3$ and $e4$ include the vehicles with high and low discharge rate, respectively;
- finally, $e5$ contains nodes that are expected to the *highly reliable*.

For example, node $u1$ is a *low-capacity*, *low-discharge* but *highly reliable* vehicle, while node $u3$ is a *high-capacity*, *low-discharge* and *highly reliable* one.

Organizing the information relating to specific agent attributes using hyperedges, enables us to both access this information efficiently, and keep it organized. Moreover, in many settings, agent characteristics captured by hyperedges, naturally correspond to criteria according to which we can form coalitions. For example, it is conceivable that we want to use agents with *high capacity* from the respective *high-capacity* edge, if our goal is to form coalitions with *high capacity*. Our approach of using hypergraphs is even more generic than what implied so far, since we can easily define hyperedges that contain agents which are or are not *permitted* to connect with each other, for various reasons; and since we can exploit the hypergraph to allow the formation of coalitions according to a multitude of criteria.

4.1 Criteria for Forming Coalitions

The algorithms presented in this work can be employed by any entity or enterprice (such as the Grid, utility companies or Smart Grid cooperatives) that wants to form EV coalitions for the V2G problem, using any set of criteria of its choosing. Here we identify three such natural criteria, namely *reliability*, *capacity* and *discharge rate*. These formation criteria are consistently mentioned in the related literature, though perhaps not with these exact names, and not explicitly identified as such [21, 20, 34].

First of all, a coalition has to be consistently *reliable*, i.e. it should be able to provide the power that has been requested without any disruptions. For a coalition to be reliable, its members must be reliable too, and gaps in reliability must be met with backup agents. We define *agent reliability* as *the estimated probability that an agent will fulfill its promises*. The *promise* of an agent is its *commitment* on being connected to the Grid during a specific time slot in order to contribute via providing energy to the Grid, if so requested. Such slots naturally correspond to electricity trading intervals.

Since the coalitions are formed to offer power services in future time slots, agents can be asked to state their availability. This availability is stored in commitment hyperedge .

In addition, a coalition must fulfill a *capacity* requirement. The *capacity* of a coalition is the amount of electricity (measured in kWh) the coalition will be offering to the Grid; while the capacity of en EV is, similarly, the amount of electricity (in kWh) the EV will be offering to the Grid. In fact, gathering enough EV capacity to cover the Grid needs during high demand periods, is the main objective of any V2G solution. Naturally, creating a coalition to meet a high power peak requires a considerable amount of capacity offered. On the other hand minor peaks can be stabilised by building EV coalitions with a much lower capacity.

Another factor in the V2G problem is the *discharge rate* of a coalition (or, of a single EV)—the rate by which the coalition (resp., the EV) is able to provide (electrical) energy to the Grid over a specified time period. Discharge rate is measured in kW. A high coalitional discharge rate could be required in cases where capacity should be offered within a small amount of time, for example when the Grid is under a heavy demand load. Naturally, a coalition has a high discharge rate if its members discharge rates are high; for our purposes, we assume that the discharge rate is additive, i.e., the discharge rate of a coalition is the sum of its EVs discharge rates. In Chapter 5, we will be forming coalitions in order to meet specific capacity and discharge rate targets; and observing how reliable the coalitions meeting these targets are.

Now, the hypergraph used in our current implementation was designed so that it could easily satisfy requests pertaining to these particular criteria. As such, there was a total of 25 hyperedges in the hypegraph—{*extremely-high, very-high, high, medium-high, medium-low, low, very-low, extremely-low*} $\times$ {*capacity, discharge rate, reliability*}; and a *committed* one, containing EVs that have stated they will be connecting to the Grid during the particular slot.[2]

In our model, we assume that, at any time step that this is required—due to a consumption peak, an unplanned event, or the need to regulate frequency and voltage—the Grid (or some other entity) advertises its demand for a V2G coalition with several desirable characteristics. As noted in [20], individual EVs are well-suited for providing services at short notice. What we show in this thesis, is that we can select agents from a huge pool of EVs to form *coalitions* that are able to provide large amounts of power at short notice, and with high reliability.

4.2 Pruning the Hypergraph

An important aspect of using hypergraphs for dealing with large state-spaces, is the resulting ability to perform node and edge pruning. Since dozens or hundreds of thousands of our EVs populate the hypergraph, and each one is a member of several hyperedges, running the algorithms without pruning would require an enormous amount of computing power. However, due to the nature of the hypergraph, and the way we store our vehicles and their attributes, it is extremely easy and effective to narrow

[2] We could have stored the commitment of the EVs on a "per time slot" basis, by using several hyperedges (one per slot) without any additional cost. However, in our experiments, we focus on a single time slot only.

down the number of vehicles and edges used, by leaving out EVs that are less promising as coalition members. For example, if achieving a high capacity for the to-be-formed coalition is a key goal, then, intuitively, we can narrow down our search for coalition members by focusing only on nodes belonging to the set of hyperedges (or "categories") $highcapacity \cup veryhighcapacity \cup exhighcapacity$.

To illustrate pruning, Fig. 4.1 shows a hypergraph that contains all EVs. In order to reduce the size of the hypergraph and thus the computing requirements, we could keep only EVs belonging to at least one high quality edge, as shown in Fig. 4.2.

Algorithm 1 Pruning the Hypergraph

procedure PRUNING($H, CategoriesKept$)
    for Hyperedge $\in$ H **do**
        if $Hyperedge \in CategoriesKept \cap Committed$ **then**
            $NewHEdges \leftarrow NewHEdges \cup HyperEdge$
            $NewNodes \leftarrow NewNodes \cup HyperEdge.nodes$
        end if
    end for
    $NewHGraph \leftarrow Hypergraph(NewNodes, NewHEdges)$
end procedure

Algorithm 1 is our implementation of pruning. The algorithm iterates over all hyperedges in the given hypergraph H, and keeps only the nodes belonging to hyperedges that correspond to the specified "categories of interest" (*CategoriesKept* in Alg. 1).

In our implementation, the *CategoriesKept* are heuristically selected, and depend on the algorithms. For instance, the *minimal transversal* algorithm requires a more aggressive pruning, since its complexity is sensitive to the number of nodes used as input (cf. Section 4.3), and we therefore empirically feed it with as few hyperedges as possible. In section 5.1 we provide Table 5.2 showing the efficiency of our pruning algorithm.

Our experimentation indicates that the use of pruning can lead to a significantly smaller hypergraph, and to vast improvements in terms of execution time for our algorithms. In our simulations, the hypergraphs are pruned to about $1/20$ of the initial size of the EVs pool, without sacrificing the methods' performance (cf. Section 5.1). Moreover, pruning using Algorithm 1 is almost instantaneous.

4.3 A Minimal Transversal Algorithm

Using hypergraphs allows to use an intuitive approach for locating agents for coalitions: to generate the set of *minimal transversals* for the *high-value hyperedges* [13]. A *transversal* (or *hitting set*) of a hypergraph H, is a set $T \subseteq V$ with hyperedges X where $X = E$ (i.e., vertices in T belong to *all* hyperedges in E). A *minimal transversal* is a set that does not contain a subset that is a hitting set of H. As such[3], generating several minimal transversal sets for *high-quality* hyperedges is expected to identify agents which are high-quality and should be used in the formation of a coalition. Subsequently, we join those agents together until our criteria are met.

Our approach with the minimal transversal set is to prune all edges but those of extremely high quality that are also "committed", as seen in Algorithm 2. Then we generate progressively the minimal hitting sets, using an algorithm similar to [13]. That is, we first generate the minimal hitting sets containing one node, then those with two, and so on. Then we randomly pick agents belonging to those minimal transversals, until the coalitions requirements are met. If the requirements are met during the progressive minimal transversal generation process, no further minimal transversals are generated.

To illustrate this concept with the help of Fig. 4.1, we prune the hypergraph to keep only the high-quality edges e_1, e_3, e_5, leaving us with the nodes $u_1, u_3 ... u_7$ and edges e_1, e_3, e_4, as seen in Fig. 4.2. Then we generate all the minimal transversal sets. The minimal transversals generated first are the ones with two nodes (since there are no minimal transversals with one node) i.e. the following$\{u_3, u_5\}, \{u_1, u_7\}, \{u_6, u_1\}$.

This method creates a set of agents with uniformly distributed high-quality characteristics. Though this is desirable in theory, in practice the results vary depending on the generated minimal transversal set. There are characteristics which might be of higher importance than others and this cannot be taken into account by the transversal algorithm due to its nature. Regardless, this method could be of much use for creating a base of quality agents; for uniformly improving the quality of an already formed coalition

[3] Of course there can be more than one minimal transversals, and it is not necessary that they have the same cardinality.

by adding agents from the minimal transversal sets; and for creating versatile coalitions without focusing on specific attributes.

Algorithm 2 Coalition formation using Minimal Transversal

```
1: procedure MINIMALTRANSVERSAL(H)
2:     H ← Prune(H, exhigh)        ▷ exhigh signifies all hyperedges with exhigh qualities
3:     T = ∅, C = ∅                ▷ Start with an empty coalition
4:     for i=1 to |E| do ▷ where |E| is the number of edges in the (pruned) H
5:         Create the union U of minimal transversal sets with size i, generated from H.
6:         T = T ∪ U
7:         while C does not meet the criteria do
8:             Randomly select an unselected node ∈ T and add it to C
9:         end while
10:        if criteria have been met then
11:            return formed coalition C
12:        end if
13:    end for
14: end procedure
```

Line 6 of Algorithm 2 is our implementation of minimal transversal [13]. Though there is no known polynomial time algorithm for the general hypergraph transversal problem, the algorithm given was shown experimentally to behave well in practice, and its memory requirements are polynomially bounded by the size of the input hypergraph, though it comes without bounds to its running time.

4.4 A Clustering Algorithm

The second approach is to create clusters of agents. After creating said clusters, we efficiently calculate the best cluster and then sample EVs from that group until our coalition criteria are met.

In more detail, we first generate a hypergraph of EV agents with the characteristics described previously. Then, hypergraph clustering is performed. The hypergraph clustering itself is an implementation of that proposed in [39], and is conducted as follows.

We begin by implementing functions that calculate

- *the Incidence Matrix*: A matrix H with entries $h(u, e) = 1$ if $u \in e$ and 0 otherwise.

- *the Weight Matrix*: A diagonal matrix W containing the weights of the hyperedges.
- *D_u and D_e*: Matrices containing the node and hyperedge degrees respectively.
- *the Adjacency Matrix*: A matrix defined as $A = HWH^T - D_u$

The matrices above are used for the final calculations of the hypergraph *Laplacian matrix*. This a matrix representation of a graph, that has information on the degrees of the nodes, and their connections with the hyperedges (cf. [39], Section 5). After its calculation, the Laplacian contains the node degrees in its diagonal (which enables us to discard the D_u matrix, for memory efficiency).

As explained in [39], having the Laplacian, enables us to calculate the Φ eigenvectors $[\Phi_1 ... \Phi_k]$ corresponding to the k lowest eigenvalues. These can then define $X = [\Phi_1 ... \Phi_k]$, a matrix that can be employed for k-way partitioning to cluster our agents. This is achieved via running the *k-means* algorithm [17] on the row vectors of X[39]. As explained in [39], the rows of X are representations of the hypergraph vertices in the k-dimensional Euclidean space. Of course, choosing a value for k has to be decided empirically. In Section 5.4 we will be testing different values for k. After generating the clusters, we are given the task to locate the "best" cluster among them. To do this efficiently, we simply sort them by looking at *the average of the node degrees*. [4] This provides us with a cluster that is better than the rest. We then sample nodes from the best cluster until our criteria are met. Algorithm 3 summarizes the method.

4.5 A Heuristic Algorithm

While using a minimal transversal generates quality sets of agents, computing the *degree* of a node can identify single agents with many quality attributes. As an example, when we have a reliable coalition as a base but we require more capacity, we can use the sorted list we have generated, to pick agents with high capacity. Intuitively, this approach will result to picking

[4] Note that the Laplacian matrix can also be used to extract easily high-quality agents, by retrieving nodes that have high values (high node degrees) in its diagonal.

Algorithm 3 Coalition formation using Hypergraph Clustering

procedure CLUSTERING(H)
    $H \leftarrow Prune(H, (vhigh \cup exhigh))$ ▷ exhigh and vhigh signify the sets of extremely high and very high quality hyperedges respectively
    Generate k clusters using the algorithm described in 4.4 [39]
    $C = \emptyset$ ▷ Start with an empty coalition
    Find the best cluster, A, by comparing the sum of node degrees of each cluster.
    while C does not meet the criteria **do**
        Randomly select a node $\in A$ and add it in C
    end while
end procedure

high overall quality agents for our coalition. We can also create coalitions by using only the best available agents. Moreover, we can use the aforementioned sorted-by-degree list of nodes in order to "fill gaps" and improve on the quality of already formed coalitions.

Thus, our heuristic method operates as follows. *(1)* First, we prune the hypergraph to include only "promising" nodes and hyperedges. For instance, we exclude nodes not in *extremely high* or in *very high* hyperedges. *(2)* Then we sort the remaining nodes based on their node degree. *(3)* Finally, we pick the highest degree nodes from the list until the coalition criteria are met. By starting at the top of the list, we can guarantee that agents have many positive characteristics.

We can see at step *(1)* above, that this algorithm, like the rest of our methods, employs pruning. As such, it does exploit the hypergraph structure. However, in practice the algorithm can deliver excellent results without much pruning. In our experiments in Chapter 5 below, the heuristic approach is shown to outperform the rest while pruning only the non-committed nodes in the hypergraph. In fact, one strength of this approach is that it does not *rely* on pruning, since its complexity is low: essentially, that of the algorithm employed for sorting (i.e., $O(nlogn)$, since we use with Python's built-in *Timsort* algorithm). By not relying on pruning, the algorithm can focus on promising nodes with high node degree (and, therefore, quality), irrespective of the exact hyperedges to which they belong.

4.6 A Hybrid Algorithm

In an attempt to exploit the strengths of each method we devised a hybrid algorithm that selected agents using both the transversal and heuristic method. As mentioned above, the transversal algorithm has the ability to find coalitions that are good in all aspects. The heuristic algorithm though, identifies single agents that are good depending on the weight of the edges.

The hybrid method works as follows. *(1)*It prunes the hypergraph using the methods described to shrink the size of the pool. *(2)*Then the transversal algorithm runs and generates a coalition with k times the needed goals. The agents of the new coalition are $V_1 \subset V$. *(3)* Finally, instead of randomly selecting agents from the minimal transversals generated, the heuristic method runs on a hypergraph $H_1 = (V_1, E)$ and selects the final coalition. It will also fill in the rest of the gaps in the coalition if required. This is better shown in 4.

Thus, the hybrid method can guarantee that coalitions generated share the properties of coalitions formed through the simple transversal algorithm, but also manages a better quality by selecting the best agents using the heuristic method.

Algorithm 4 Coalition formation using Hybrid Approach

```
1: procedure HYBRID(H)
2:     H ← Prune(H, (vhigh ∪ exhigh))    ▷ exhigh and vhigh signify the
   sets of extremely high and very high quality hyperedges respectively
3:     Generate a coalition V1 ⊂ V using algorithm 2 but using k times the
   requirements (goals) needed. (k is empirically selected)
4:     Generate a hypergraph H = (V1, E) and run the heuristic algorithm
   for possibly the final coalition.
5:     If needed (k < 1), run heuristic again to fill in the gaps.
6: end procedure
```

This method does have a disadvantage. While the resulting coalitions are of high quality, the runtime is much higher than the simple methods since both algorithms run sequentially (with a lower goal, though). Nevertheless, the runtime is still low and can be further reduced using methods discussed in Chapter 6.

4.7 A Simple Sampling Method

For interest, and in order to have a benchmark for the rest of our algorithms, a simple sampling algorithm was also developed. The algorithm takes random samples until the specified goals are achieved.

Chapter 5

Experiments and Results

In this section we present the evaluation of our algorithms. First we explain how the EV population is generated, and the time this generation process takes. Then, the performance of the algorithm is evaluated in terms of the quality of the formed coalition and also in terms of execution time and scaling behavior. All figures and tables present average values over multiple runs. Specifically, we generated 20 hypergraphs with $20,000$ EVs each, and then ran each algorithm on every hypergraph 10 times, and took the averages (and the average of those averages). Our experiments were run on a Sandy Bridge i7-2600K at 4.2 GHz. All the tests were running on a single thread on Python, meaning that there is a lot of room for optimization.

5.1 Generating the EV Population

To generate the population for each type of experiment we create the vehicles one by one, by first generating its properties as follows. The capacity of each vehicle is generated from a Gaussian distribution with mean value 100 and $\sigma = 80$. The discharge rate of each vehicle is generated from a Gaussian distribution with mean value 10 and $\sigma = 5$. The reliability of each vehicle is picked from a Gaussian distribution with mean value 0 and $\sigma = 1$. Each EV's commitment of being connected to the Grid is a *true / false* variable, with a 0.9 probability of being *true*. If *true*, then the EV is inserted in the *committed* hyperedge. When a vehicle has its properties created, it is added in the pool of available EVs. The computational complexity of generating the hypergraph is, as expected, $O(n)$.

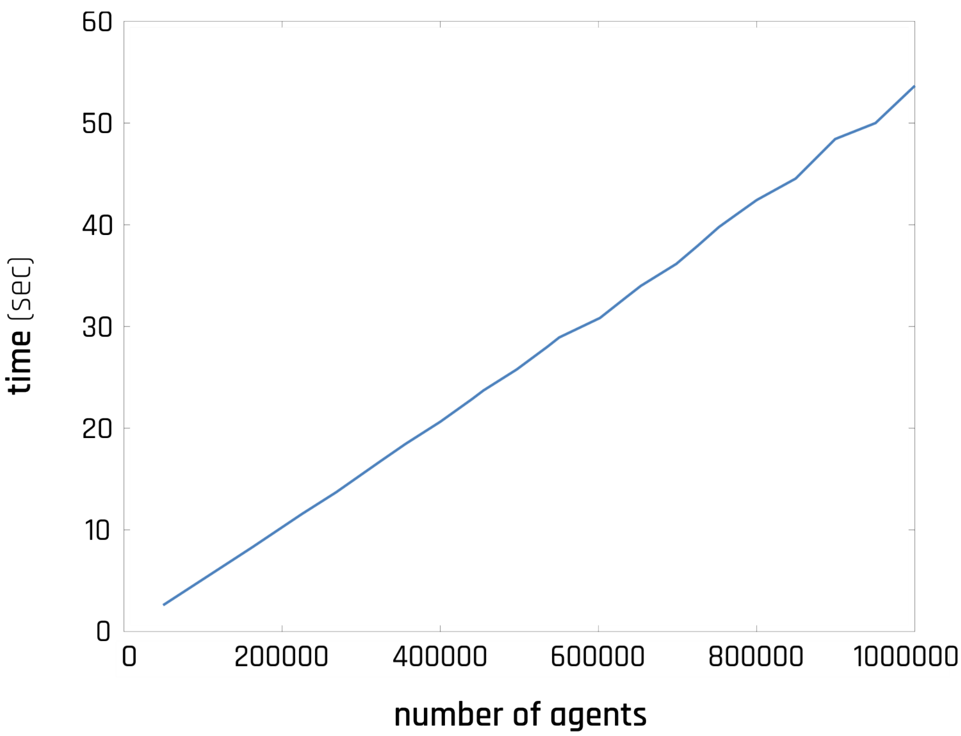

FIGURE 5.1: Hypergraph generation scaling

The coalition requirements are set to values which are commonly used in the regulation market [20], namely the following two. First, each coalition must have a total capacity of at least $10MWh$. The discharge rate must also be at least $1MW$ [20] These values are kept constant throughout all experiments—except when we test scaling against an increasing capacity goal, where capacity is treated as a variable. [1]

Creating the hypergraph is a problem that scales linearly with time. Specifically, generating the hypergraph, including the vehicles and distributing them to hyperedges, takes a very small amount of time and scales linearly up to a million within a minute (Table 5.1 and Fig 5.1). As mentioned above, the initial EV population was $20,000$ nodes. However, before feeding the nodes to the algorithms, we pruned the hypergraph to keep promising nodes. Table 5.2 shows the average hypergraph size finally fed to the algorithms.

[1] As stated in Chapter 4, our hypergraph used 25 hyperedges to store the attributes.

EVs	Generation Time (sec)
100,000	5.08
200,000	10.23
300,000	15.31
400,000	20.51
500,000	25.75
600,000	30.69
700,000	36.14
800,000	42.38
900,000	48.37
1,000,000	53.62

TABLE 5.1: Hypergraph generation scaling timings

Algorithm	Nodes after Pruning	Edges after Pruning
Transversal	1148.4	4
Clustering	1218.8	7
Heuristic	18012.6	25

TABLE 5.2: Pruning Results

5.2 Forming the Coalitions

We now proceed to evaluate the performance of our algorithms. Our evaluation will examine *(a)* how fast and *(b)* by selecting how many vehicles they can meet the set requirements. Naturally, the faster an algorithm forms a coalition that meets all the requirements, the better.

Moreover, coalitions with fewer vehicles are preferable, since intuitively, this allows for a more efficient allocation of resources, and also means that fewer EVs will share the payoff associated with forming the coalition (exactly how this payoff allocation will occur, is a problem we do not deal with in this thesis).

To begin, in all Figs. 5.2—5.5:

- In all subfigures, the horizontal axis depicts the progression of the coalition size.

- *Capacity subfig.* On the first graph of each figure, the capacity of the coalition is displayed. We can see how it is increased by selecting the appropriate agents until the goal (horizontal line) is reached.

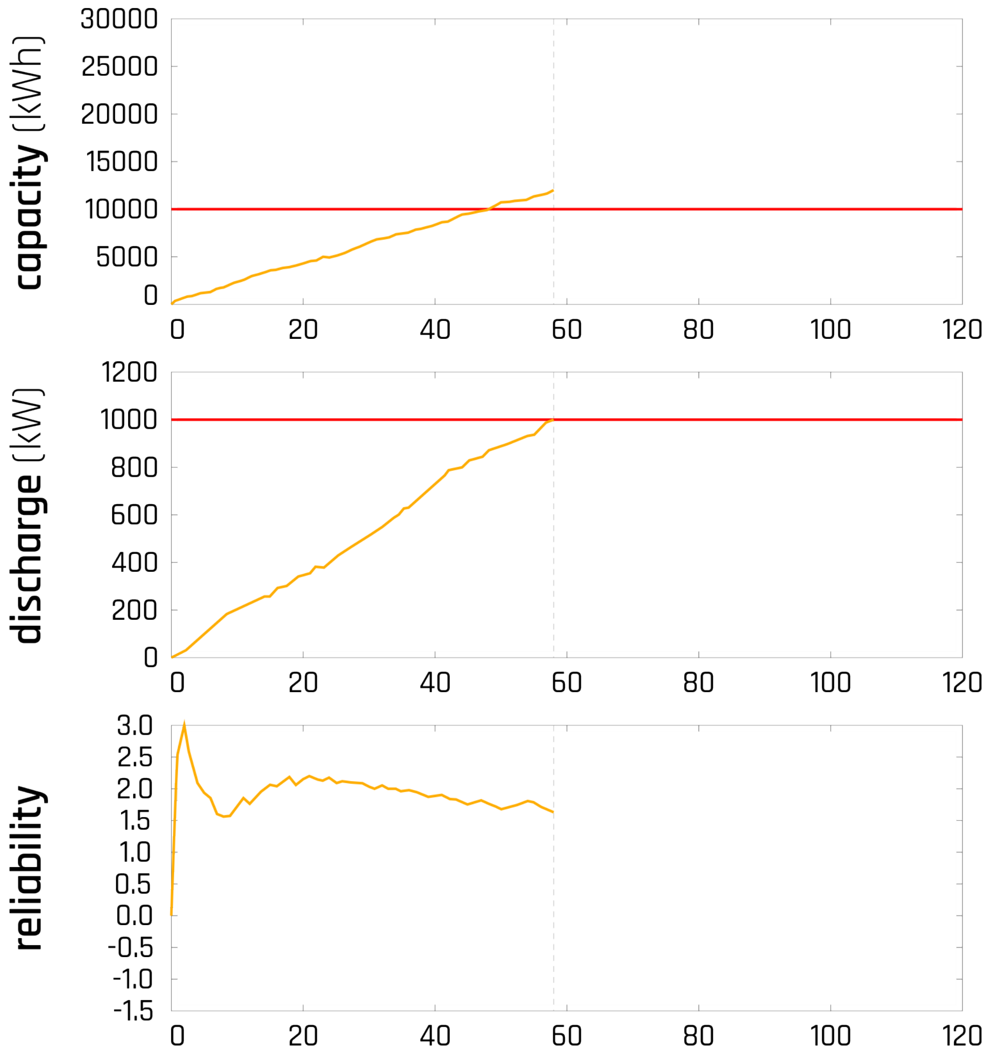

FIGURE 5.2: Coalition formation with the Heuristic Algorithm

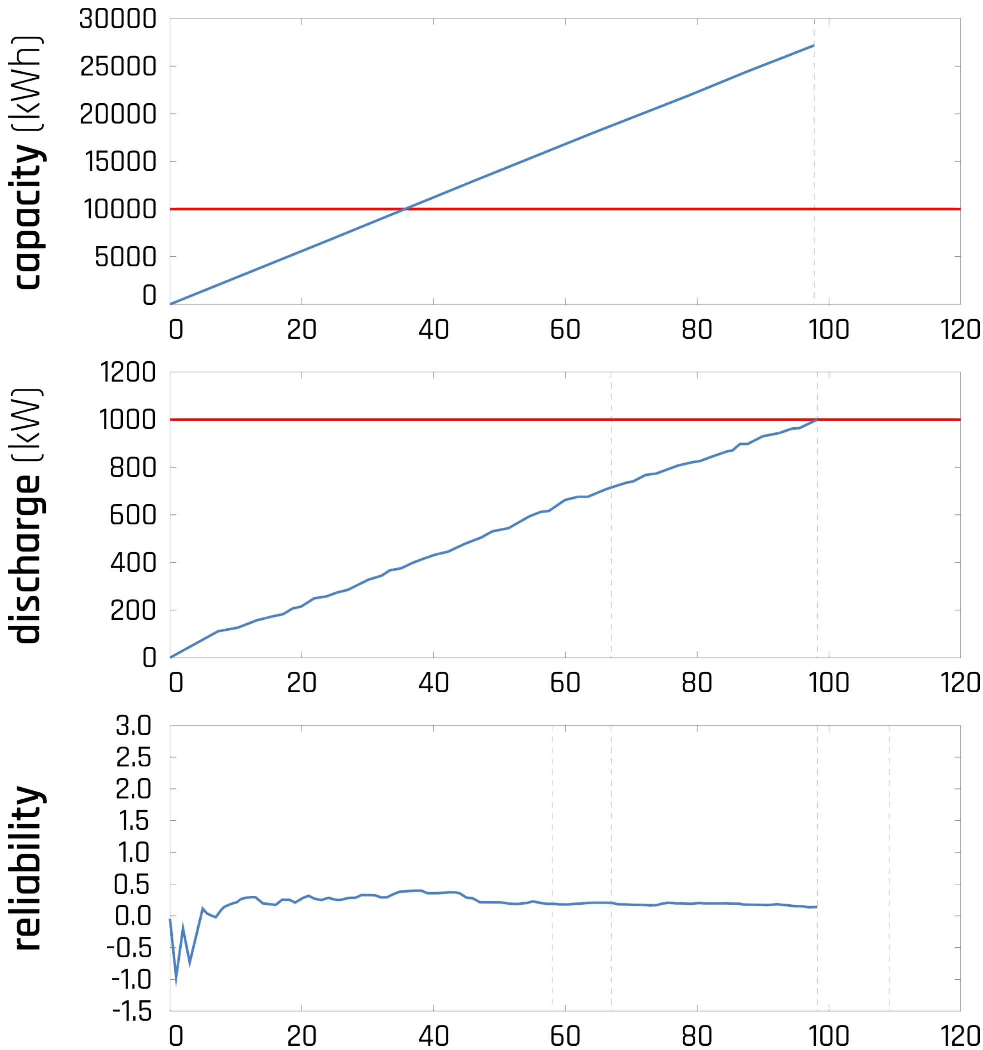

FIGURE 5.3: Coalition formation with the Clustering Algorithm

- *Reliability subfig.* The second graph displays the mean reliability of our coalition.
- *Discharge subfig.* The third and last graph displays the discharge rate of the coalition. The goal of $1,000$ kW is shown as a horizontal line.

Heuristic Algorithm As explained in Section 4.5, this algorithm attempts (in a rather "greedy" manner) to identify the best EVs from the hypergraph. As we can observe in Fig. 5.2, it takes on average only 58.5 vehicles to reach the goal requirements, which is the most efficient use of resources observed across all our methods. The reliability achieved is also high, reaching a value of more than 1.5. We remind the reader that the mean reliability of our pool of EVs is 0. This approach is also the most time and memory efficient of all. Specifically the algorithms average completion time is only $25ms$ for these experiments, and it also scales linearly into the millions as seen in Fig. 5.10 below.

Clustering Algorithm This method performs clustering, as explained in Section 4.4, and then takes random samples from the best cluster. Fig. 5.3 depicts its performance when using $k = 3$ clusters. Unfortunately, we cannot control how exactly the clusters are formed, so we do not have a guarantee that high quality vehicles will be clustered together. This leads to a mediocre result with an increased average coalition size, and a slightly-over-the-average reliability. The average size of coalitions meeting both requirements is 98. The average time required for the method's completion is $709ms$. In Section 5.4, we show how different k values affect our results.

Transversal Using the transversal algorithm and taking nodes from a list of minimal hitting set. Fig. 5.4 shows its performance. The transversal algorithm appears to work quite well since the average coalition size is only 64, slightly higher than that achieved by the heuristic approach. The reliability of the coalition is high, reaching values over 1.1. It can also scale quite well, reaching thousands of vehicles (cf. Fig. 5.8 and Table 5.5), but not as well as the heuristic approach. The average time to completion was $120ms$.

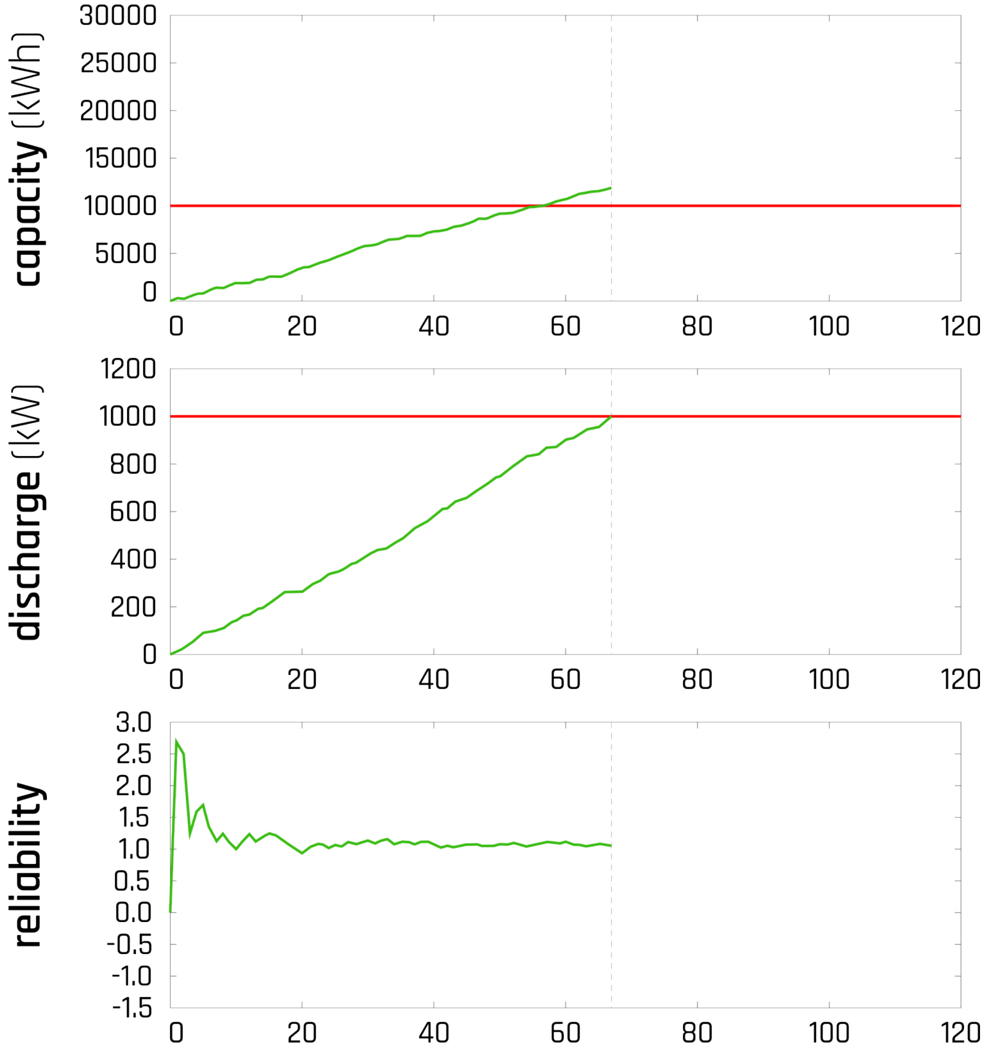

FIGURE 5.4: Coalition formation with the Minimal Transversal Algorithm

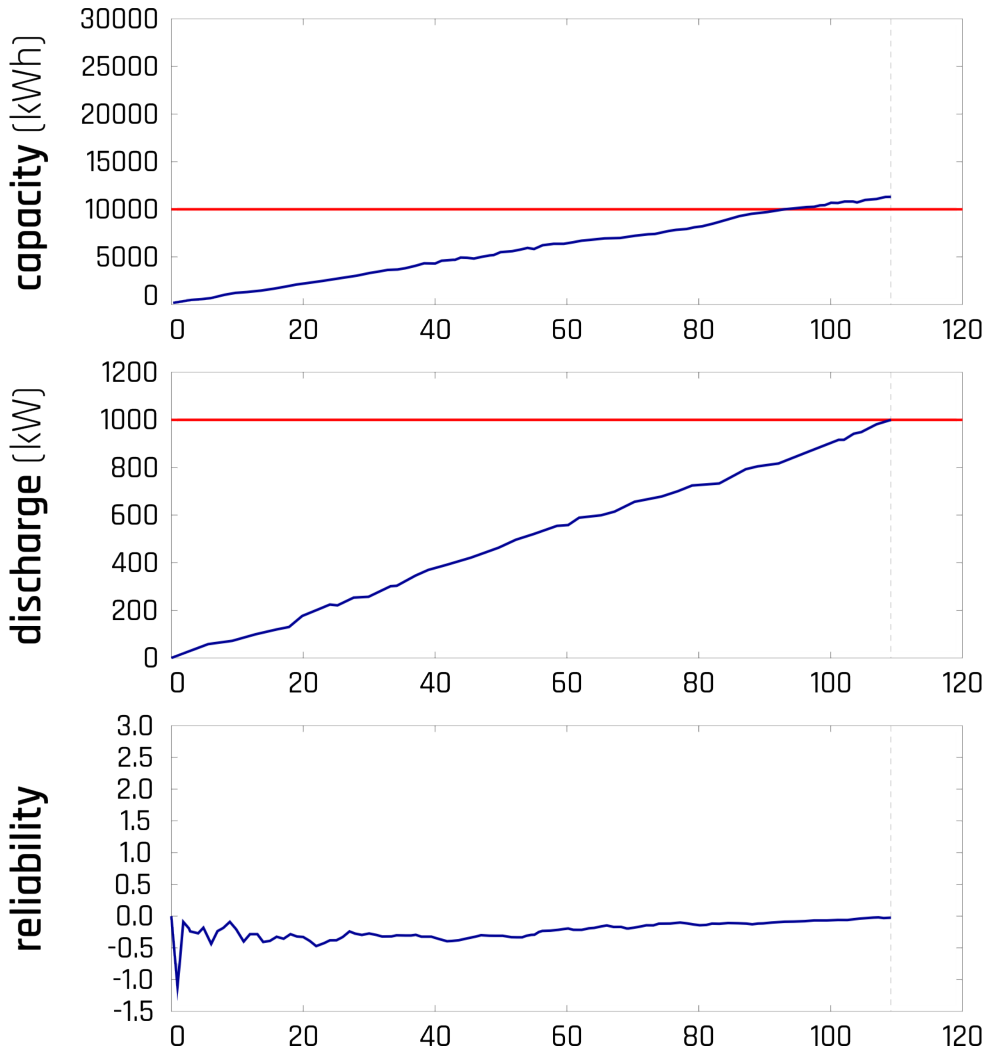

FIGURE 5.5: Coalition creation with the Simple Sampling Algorithm

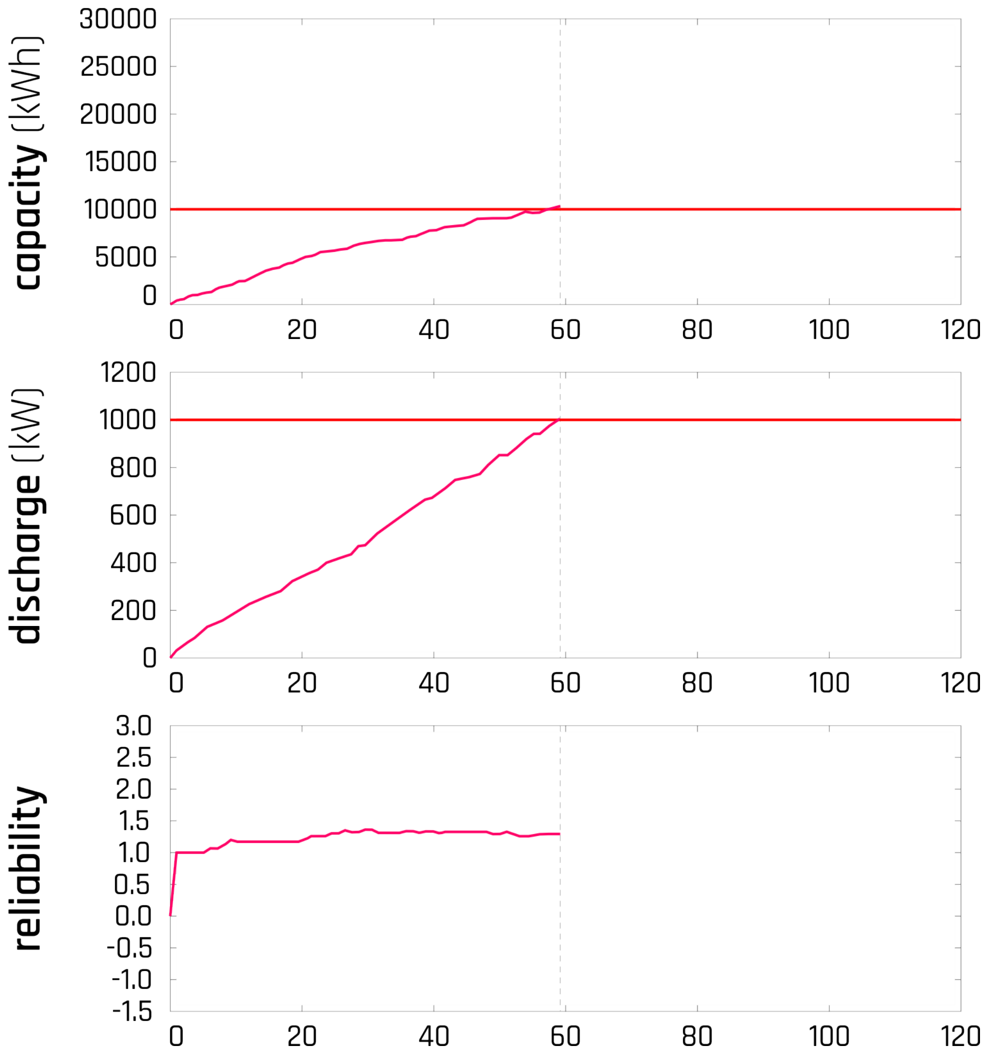

FIGURE 5.6: Coalition creation with the Hybrid Algorithm

Algorithm	Coal. Size(#EVs)	Run. Time(ms)	Gen.+Run. Time(ms)
Heuristic	58.5	25	1041
Clustering	98	709	1725
Transversal	64	120	1136
Hybrid	59.6	75	1091
Sampling	109.3	24	1040

TABLE 5.3: Summarizing the performance results

Hybrid The hybrid algorithm takes EVs from coalitions created with transversal and heuristic. It runs faster than the transversal algorithm with better results while still having the advantages discussed in Sec. 4.3. Fig. 5.6 depicts its performance. The average coalition size is just a bit higher than the heuristic at 59.6. Finally the mean running time is 75ms.

Simple Sampling Fig. 5.5 depicts our results for the Simple Sampling method. The average coalition size achieved with this algorithm is 109.3. The average completion time was 24ms. As expected, this algorithm achieves the weakest results among all our algorithms.

Finally, Fig. 5.7 and Table 5.3 summarize the results for convenience. In the figure it can be easily seen that the hybrid and heuristic approach manage smaller coalitions by balancing correctly between *discharge* and *capacity* and finding overall good EVs. It can also be seen that the clustering algorithm selects high capacity EVs early on, thus wasting a lot of potential afterwards, by having to select the rest of the EVs to reach the *discharge* goal. This can be mitigated by increasing the number of clusters (Sec. 5.4)

5.3 Scaling Behaviour

We now test the scaling behaviour of our algorithms. First, we show how our algorithms scale with time when the *capacity* goal is increased. Then, we show how they scale as the number of EVs under consideration increases.

In Fig. 5.9 and Table 5.4 we can see how the transversal, heuristic and clustering algorithm scale against an increasing capacity goal (assuming any other goal remains fixed). The starting size of the available agents was kept constant at $20,000$ EVs for this experiment. We observe that the scaling behaviour of the heuristic algorithm against an increasing capacity goal is

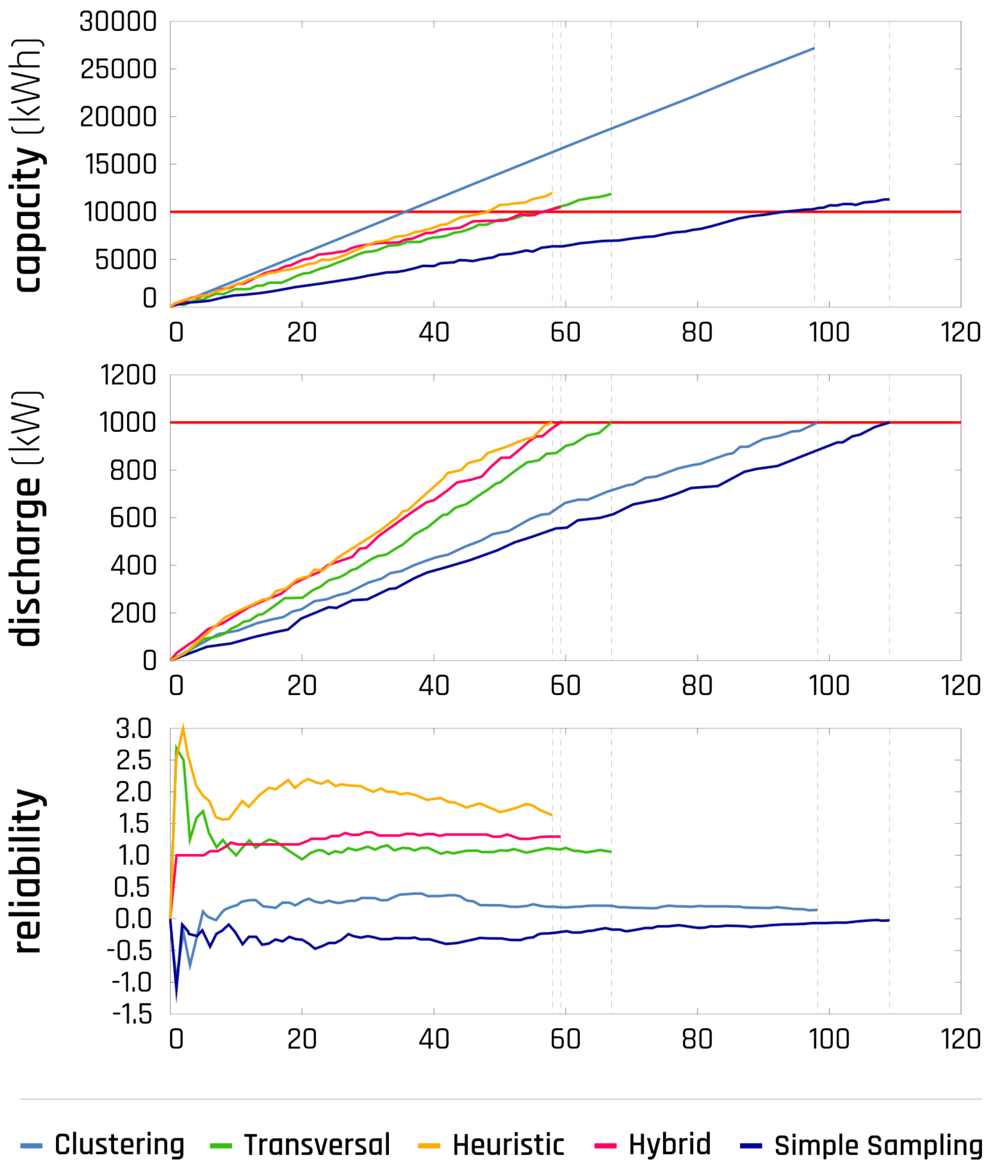

FIGURE 5.7: Coalition creation

Goal (kWh)	Heuristic (sec)	Clustering (sec)	Transversal (sec)
10,000	0.03	0.049	0.20
40,000	0.04	0.049	0.36
70,000	0.06	0.049	0.34
100,000	0.11	0.049	0.33
130,000	0.19	0.049	0.33
160,000	0.28	0.049	0.33
190,000	0.39	0.049	0.34
220,000	0.50	0.049	0.32
250,000	0.64	0.049	0.32
280,000	0.86	0.049	0.32

TABLE 5.4: Scaling against an increasing "capacity" goal

exponential. Nevertheless, its total required execution time is low, since it takes the algorithm 0.9 seconds to reach the goal capacity of 300,000 kWh. The transversal algorithm scales with steps. The main reason for this is that the minimal transversal sets are generated before we select the agents of a coalition. If a minimal transversal set does not achieve the goal capacity, we generate a new one with more agents, till we reach the set capacity goal. This generates a step pattern, the first stages of which are shown in Fig. 5.9. In Fig. 5.9 we actually manage to see only one step because generating the minimal transversals with 3 EVs is enough to find good coalitions for all goals from $40,000$ kWh onwards (while it was enough to generate the minimal transversals with 2 EVs to cover the $10,000$ kWh capacity goal).

Now, the running time of the hypegraph clustering algorithm is largely independent of the size of the stated capacity goal. This is because the clustering itself, which is the part of the algorithm that requires the most processing power, takes place regardless of the final coalition requirements. Indeed, we observe in Fig. 5.9 that after an initial jump due to increased sampling requirements (cf. lines 6—8, Alg. 3) when moving from a goal of $10,000$ to $40,000$ kWh, the algorithm's running time remains largely unaltered. Fig. 5.8 displays scaling against the initial EV population. The coalition goals were kept constant, and the same for all algorithms. The heuristic algorithm shows a linear scaling in time as the agent size grows. Specifically, the heuristic algorithm can scale *up to a million agents* within an acceptable time.

Fig. 5.10 demonstrates this behaviour, starting from $50,000$ EVs. Of course, one expects that when the population reaches several millions, the complexity of the sorting algorithm will kick in, creating bottlenecks. Regardless, the fact that linear scalability is maintained up to $1,000,000$ agents is reassuring. By contrast, looking at Fig. 5.8, we observe that the transversal and clustering algorithms scale exponentially.

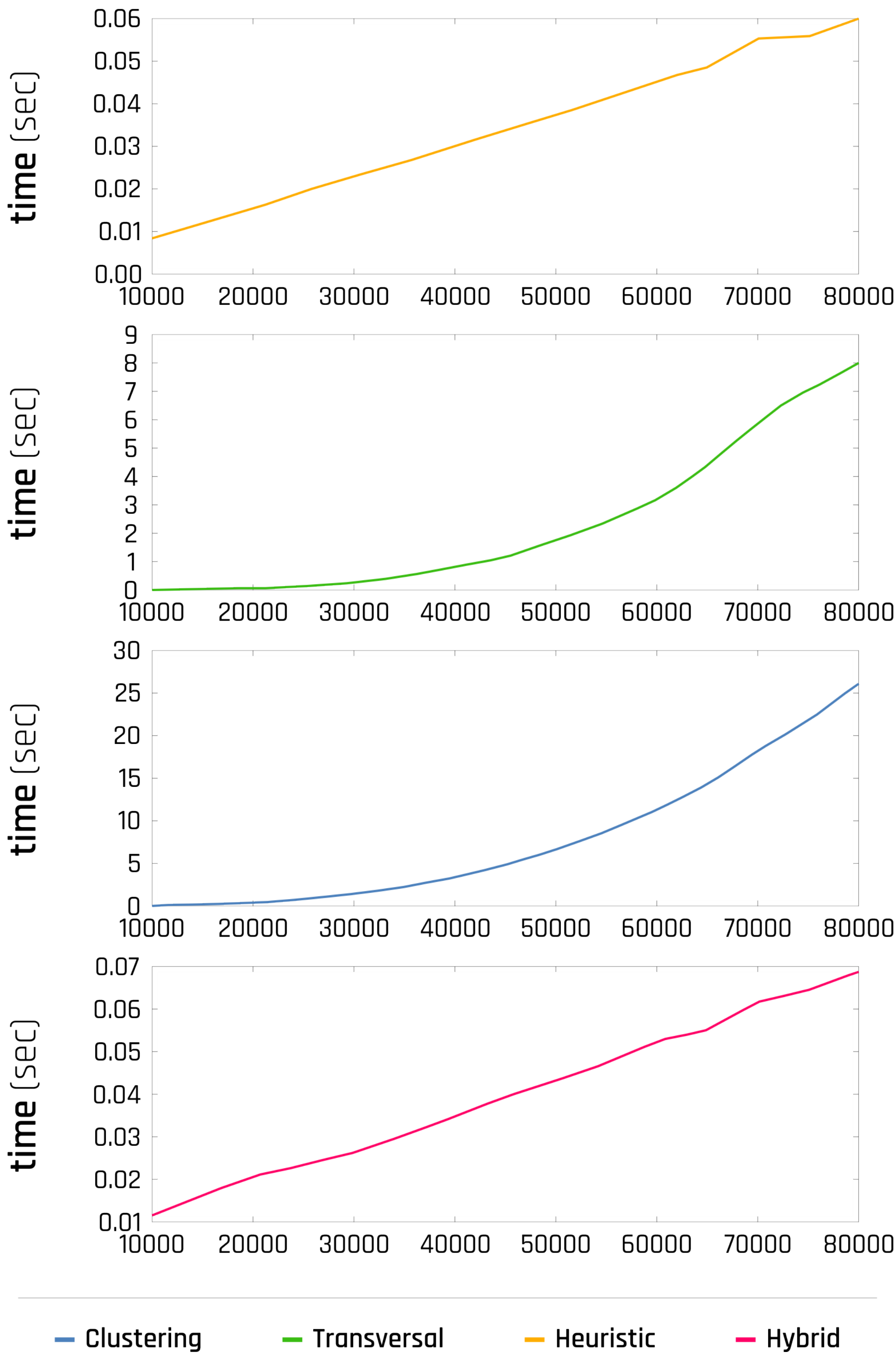

FIGURE 5.8: Scaling against an increasing EV population

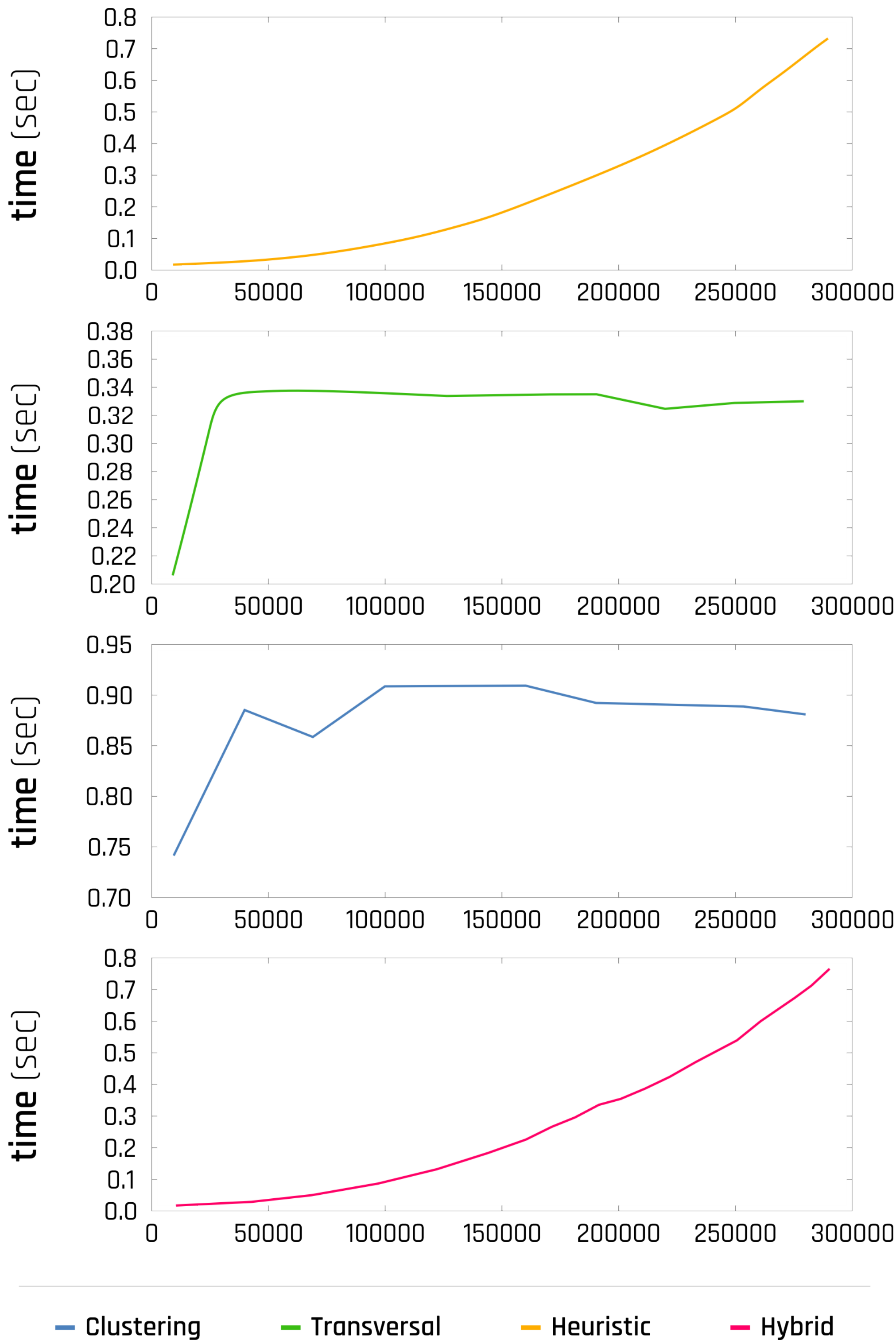

FIGURE 5.9: Scaling against an increasing "capacity" goal

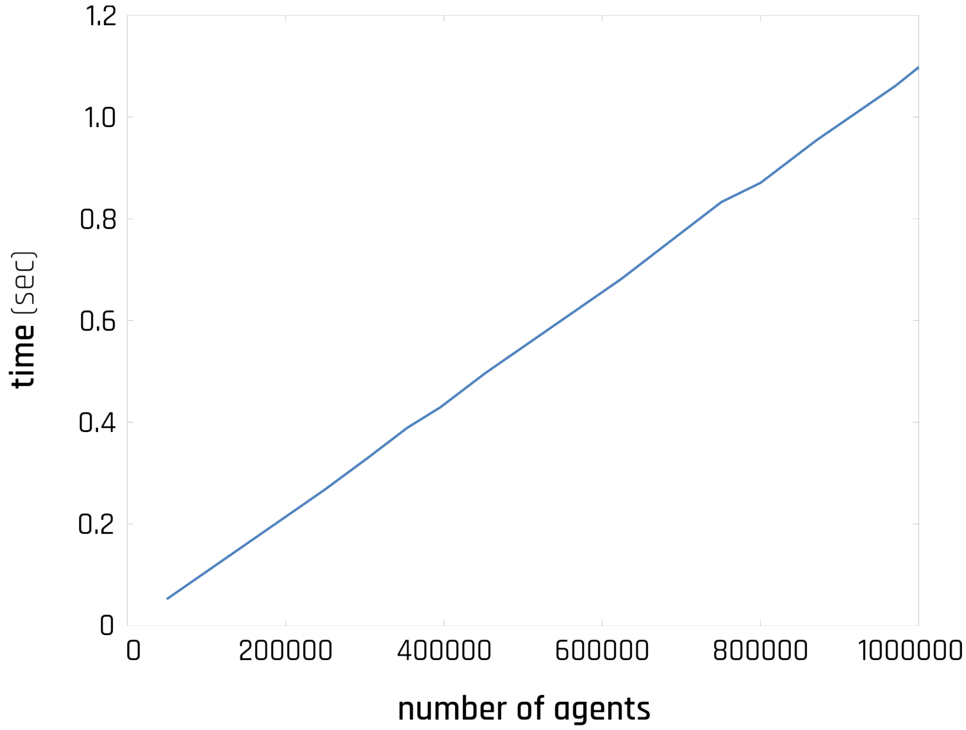

FIGURE 5.10: Scaling of the Heuristic Algorithm

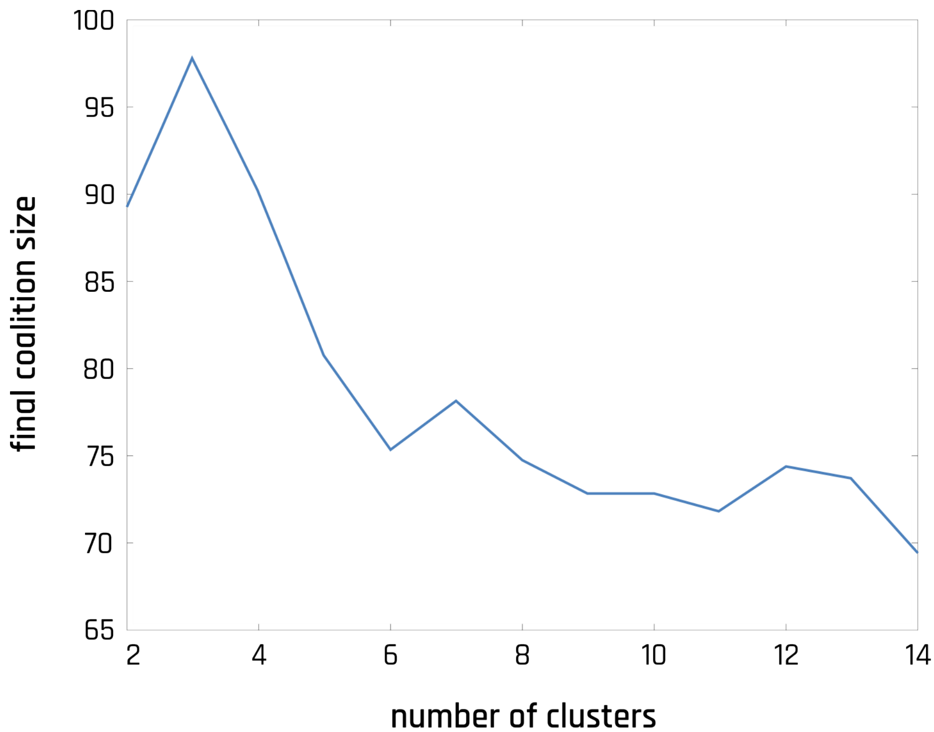

FIGURE 5.11: Evolution of the average size of coalitions produced with the Hypergraph Clustering method, when varying the number of clusters

EVs	Heuristic (sec)	Clustering (sec)	Transversal (sec)
10,000	0.012	0.14	0.03
11,000	0.013	0.17	0.04
12,000	0.015	0.22	0.05
13,000	0.016	0.25	0.06
14,000	0.017	0.31	0.07
15,000	0.018	0.11	0.07
16,000	0.019	0.36	0.08
17,000	0.020	0.43	0.08
18,000	0.021	0.50	0.09
19,000	0.023	0.57	0.10
20,000	0.024	0.69	0.12

TABLE 5.5: Scaling against an increasing EV population

5.4 Varying the number of hypergraph clusters

We test our clustering algorithm further by modifying the number of clusters, k, since this is a parameter that can be optimized empirically, as explained in Section 4.4.

Fig. 5.11 displays the relation between k and the average coalition size that results from the clustering method (and which achieves the set goals). Creating a larger number of clusters results in smaller, and thus better, coalitions. Regardless, even when $k = 15$, the clustering algorithm still produces coalitions with more EVs than the heuristic one.

Chapter 6

Conclusions and Future Work

In this thesis, we demonstrated how to employ hypergraphs for creating coalitions based on multiple criteria. The existence of several hypergraph transversal and clustering algorithms makes hypergraphs easy to work with. Moreover, the ability to select almost instantaneously parts of the hypergraph that are interesting, offers a significant advantage, enabling one to generate coalitions with desirable characteristics within seconds. This makes hypergraph use quite attractive for real-world, real-time scenarios.

We presented several coalition formation methods that employ hypergraphs for tackling the V2G problem, and evaluated their performance. Our proposed heuristic algorithm, in particular, was shown to be the most effective and efficient of our methods, as it is able to use a minimal number of EVs to provide the required capacity, discharge rate, and reliability to the Grid in a few milliseconds; while it exhibits exceptional scaling behaviour with respect to the number of EVs under consideration. Ours is the first approach that is able to deal with *large-scale* coalition formation for the V2G problem, while taking *multiple criteria* into account for creating the EV coalitions.

Future work includes implementing a more efficient *minimal transversal* algorithm as follows.

Finding all the minimal transversals of a hypergraph is a computationally difficult task. As the size of the graph increases the number of patterns increases - exponentially in the worst case scenario. Nevertheless, while the pool of EVs might be huge, a coalition meeting the requirements could be small enough and require only a handful of transversals to be generated.

For this reason, it is worth exploring ways to generate transversals in a depth-first manner. This would enable us to create as many as we required to fulfill the requirements of the coalition. Such an attempt was presented in [22]. This paper suggest a way to create transversals one by one as opposed to generating simultaneously all sets of each size, and while it doesn't offer a time bound for the worst case scenario, it does offer bound in terms of memory use.

By implementing this algorithm, we could stop the execution as soon as the coalition being built reached the requirements. This can greatly the execution time of the Minimal Transversal algorithm and allow it to scale to higher pool sizes. This method, since it is a depth-first approach, might even move the scaling bottleneck from the size of the EV pool to the requirements.

Another algorithm, proposed in [24], for finding k minimal transversals using parallel processors. The time is bound by polylog$(|V|, |H|, k)$ assuming poly$(|V|, |H|, k)$ numbers of processors. This algorithm could be the most efficient for finding a collection of minimal transversals but unfortunately an estimation for k must be made first.

By implementing this algorithm we could gain several advantages. For instance the algorithm would not scale with pool size anymore but only with requirements. It could eventually be faster than heuristic since it can run in multiple processors (sorting used in heuristic cannot run in multiple cores). Nevertheless, the results should generally stay the same (coalition size for example) since it is only an optimization.

Finally, additional future work includes improving the clustering approach with an alternative method for representing the vertices in the Euclidean space; and for identifying promising clusters. Finally, all algorithms can be equipped with multithreading capabilities, to substantially improve their performance.

Bibliography

[1] James Bailey, Thomas Manoukian, and Kotagiri Ramamohanarao. "A Fast Algorithm for Computing Hypergraph Transversals and its Application in Mining Emerging Patterns." In: *ICDM*. Vol. 3. Citeseer. 2003, p. 485.

[2] Claude Berge. *Hypergraphs, volume 45 of*. 1989.

[3] Filippo Bistaffa et al. "Anytime coalition structure generation on synergy graphs". In: *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2014, pp. 13–20.

[4] Endre Boros et al. "An efficient implementation of a quasi-polynomial algorithm for generating hypergraph transversals". In: *European Symposium on Algorithms*. Springer. 2003, pp. 556–567.

[5] Samuel R Bulò and Marcello Pelillo. "A game-theoretic approach to hypergraph clustering". In: *Advances in neural information processing systems*. 2009, pp. 1571–1579.

[6] Georgios Chalkiadakis. "A Bayesian Approach to Multiagent Reinforcement Learning and Coalition Formation under Uncertainty." PhD thesis. University of Toronto, 2007.

[7] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. "Computational aspects of cooperative game theory". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 5.6 (2011), pp. 1–168.

[8] Georgios Chalkiadakis, Gianluigi Greco, and Evangelos Markakis. "Characteristic function games with restricted agent interactions: Core-stability and coalition structures". In: *Artificial Intelligence* 232 (2016), pp. 76–113.

[9] Georgios Chalkiadakis, Evangelos Markakis, and Nicholas R Jennings. "Coalitional stability in structured environments". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems. 2012, pp. 779–786.

[10] Filippos Christianos and Georgios Chalkiadakis. "Efficient Multi-Criteria Coalition Formation using Hypergraphs (with Application to the V2G Problem)". In: *In Proc. of the 14th European Conference on Multi-Agent Systems (EUMAS-2016), December 2016, Valencia, Spain*. 2016.

[11] Filippos Christianos and Georgios Chalkiadakis. "Employing Hypergraphs for Efficient Coalition Formation with Application to the V2G Problem". In: *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands*. 2016, pp. 1604–1605.

[12] Guozhu Dong and Jinyan Li. "Mining border descriptions of emerging patterns from dataset pairs". In: *Knowledge and Information Systems* 8.2 (2005), pp. 178–202.

[13] Thomas Eiter and Georg Gottlob. "Identifying the minimal transversals of a hypergraph and related problems". In: *SIAM Journal on Computing* 24.6 (1995), pp. 1278–1304.

[14] Lingwen Gan, Ufuk Topcu, and Steven H Low. "Optimal decentralized protocol for electric vehicle charging". In: *IEEE Transactions on Power Systems* 28.2 (2013), pp. 940–951.

[15] "Global EV Outlook 2016: Beyond one million electric cars". In: ().

[16] Bernard Gorowitz. *The General Electric Story: A Heritage of Innovation, 1876-1999*. Schenectady Museum &, 2000.

[17] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28.1 (1979), pp. 100–108.

[18] Céline Hébert, Alain Bretto, and Bruno Crémilleux. "A data mining formalization to improve hypergraph minimal transversal computation". In: *Fundamenta Informaticae* 80.4 (2007), pp. 415–433.

[19] Tackseung Jun and Jeong-Yoo Kim. "Hypergraph formation game". In: *Hitotsubashi Journal of Economics* (2009), pp. 107–122.

[20] Sachin Kamboj, Willett Kempton, and Keith S Decker. "Deploying power grid-integrated electric vehicles as a multi-agent system". In: *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2011, pp. 13–20.

[21] Sachin Kamboj et al. "Exploring the formation of electric vehicle coalitions for vehicle-to-grid power regulation". In: *AAMAS workshop on agent technologies for energy systems (ATES 2010)*. 2010.

[22] Dimitris J Kavvadias and Elias C Stavropoulos. "An efficient algorithm for the transversal hypergraph generation." In: *J. Graph Algorithms Appl.* 9.2 (2005), pp. 239–264.

[23] Willett Kempton and Jasna Tomić. "Vehicle-to-grid power fundamentals: Calculating capacity and net revenue". In: *Journal of power sources* 144.1 (2005), pp. 268–279.

[24] Leonid Khachiyan et al. "A new algorithm for the hypergraph transversal problem". In: *International Computing and Combinatorics Conference*. Springer. 2005, pp. 767–776.

[25] Ioannis Lampropoulos, Greet MA Vanalme, and Wil L Kling. "A methodology for modeling the behavior of electricity prosumers within the smart grid". In: *2010 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe)*. IEEE. 2010, pp. 1–8.

[26] Marius Leordeanu and Cristian Sminchisescu. "Efficient Hypergraph Clustering." In: *AISTATS*. 2012, pp. 676–684.

[27] Rodica Loisel, Guzay Pasaoglu, and Christian Thiel. "Large-scale deployment of electric vehicles in Germany by 2030: An analysis of grid-to-vehicle and vehicle-to-grid concepts". In: *Energy Policy* 65 (2014), pp. 432–443.

[28] Björn Nykvist and Måns Nilsson. "Rapidly falling costs of battery packs for electric vehicles". In: *Nature Climate Change* 5.4 (2015), pp. 329–332.

[29] Gabriel de Oliveira Ramos, Juan C. Burguillo, and Ana LC Bazzan. "Dynamic constrained coalition formation among electric vehicles". In: *Journal of the Brazilian Computer Society* 20.1 (2014), pp. 1–15.

[30] Talal Rahwan et al. "An anytime algorithm for optimal coalition structure generation". In: *Journal of Artificial Intelligence Research (JAIR)* 34.1 (2009), pp. 521–567.

[31] Sarvapali D Ramchurn et al. "Putting the'smarts' into the smart grid: a grand challenge for artificial intelligence". In: *Communications of the ACM* 55.4 (2012), pp. 86–97.

[32] Tuomas Sandholm et al. "Coalition structure generation with worst case guarantees". In: *Artificial Intelligence* 111.1 (1999), pp. 209–238.

[33] Onn Shehory and Sarit Kraus. "Methods for task allocation via agent coalition formation". In: *Artificial Intelligence* 101.1 (1998), pp. 165–200.

[34] Konstantina Valogianni et al. "Effective management of electric vehicle storage using smart charging". In: *Proceedings of 28th AAAI Conference on Artificial Intelligence*. 2014, pp. 472–478.

[35] Meritxell Vinyals et al. "Stable Coalition Formation Among Energy Consumers in the Smart Grid". In: *Proceedings of the 2012 International Workshop on Agent Technologies for Energy Systems*. International Foundation for Autonomous Agents and Multi-Agent Systems. 2012.

[36] Thomas Voice, Sarvapali D Ramchurn, and Nicholas R Jennings. "On coalition formation with sparse synergies". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2012, pp. 223–230.

[37] Perukrishnen Vytelingum et al. "Agent-based micro-storage management for the smart grid". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2010, pp. 39–46.

[38] Kwo Young et al. "Electric vehicle battery technologies". In: *Electric Vehicle Integration into Modern Power Networks*. Springer, 2013, pp. 15–56.

[39] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. "Learning with hypergraphs: Clustering, classification, and embedding". In: *Advances in Neural Information Processing Systems*. 2006, pp. 1601–1608.

[40] Yair Zick, Georgios Chalkiadakis, and Edith Elkind. "Overlapping coalition formation games: Charting the tractability frontier". In: *Proceedings of the 11th International Conference on AAMAS-2012-Volume 2*. 2012, pp. 787–794.