IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 November 2024, accepted 19 December 2024, date of publication 30 December 2024,
date of current version 15 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3524091

RESEARCH ARTICLE

Geographical Peer Matching for P2P Energy Sharing

ROMARIC DUVIGNAU[1], (Member, IEEE), VINCENZO GULISANO[1], MARINA PAPATRIANTAFILOU[1], AND RALF KLASING[2]

[1]Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, 405 30 Gothenburg, Sweden
[2]CNRS, LaBRI, Université de Bordeaux, 33405 Talence, France

Corresponding author: Romaric Duvignau (duvignau@chalmers.se)

The work of Romaric Duvignau, Vincenzo Gulisano, and Marina Papatriantafilou was supported in part by the Chalmers Area of Advance Energy Projects Dynamic and Efficient Energy-sharing P2P networks (DEEP) and Information and Data-processing in Focus for Energy Efficiency (INDEED); and in part by InTelligent Energy DAta MaNagement and Online DEcision Making (TANDEM) Project within the framework of the Swedish Electricity Storage and Balancing Centre (SESBC), funded by the Swedish Energy Agency together with five Academic and 26 Non-Academic Partners. The work of Ralf Klasing was supported in part by the Agence Nationale de la Recherche (ANR) Project Temporal Graph Algorithms: models, questions, resolution (TEMPOGRAL) under Grant ANR-22-CE48-0001.

ABSTRACT Significant cost reductions attract ever more households to invest in small-scale renewable electricity generation and storage. Such distributed resources are not used in the most effective way when only used individually, as sharing them provides even greater cost savings. Energy Peer-to-Peer (P2P) systems have thus been shown to be beneficial for prosumers and consumers through reductions in energy cost while also being attractive to grid or service providers. However, many practical challenges have to be overcome before all players could gain in having efficient and automated local energy communities; such challenges include the inherent complexity of matching together geographically distributed peers and the significant computations required to calculate the local efficient matching options. We define and analyze in this work a precise mathematical modeling of the geographical peer matching problem, and demonstrate the inherent intractability of the problem, highlighting its high computational cost and underscoring the critical need for scalable approaches that effectively balance performance trade-offs as system size grows. Furthermore, we propose and study analytically and empirically a spectrum of approaches to address it and perform a cost-efficient matching of peers in a computationally efficient fashion. Our experimental study, based on real-world energy data, demonstrates that our proposed solutions are efficient both in terms of cost savings achieved by the peers and in terms of communication and computing requirements. Our scalable algorithms thus provide one core building block for practical and data-efficient peer-to-peer energy sharing communities within large-scale optimization systems.

INDEX TERMS Geographical peer matching, hypergraph matching, P2P energy sharing, prosumers.

I. INTRODUCTION

Renewable electricity generation is becoming more affordable to end-users as the initial investment cost has been drastically cut, thus transforming the traditional residential households from consumers into *prosumers* [14] (capable of producing locally their own electricity). Sharing resources, for example solar photovoltaic (PV) panels and battery systems, in *Peer-to-Peer* (P2P) [15] setups can be leveraged as a way to optimize the cost-benefits from those distributed renewable resources. Driven largely by the goal of reducing costs [16] and enhancing renewable energy integration, interest in P2P energy communities has been steadily increasing. This high-potential concept has attracted considerable attention from the research community in recent years (see [17], [18] and references therein) and continues to generate a significant amount of research [19], [20], [21], [22], [23].

The principle of P2P energy sharing is for different end-users to share their resources locally in groups, in order

The associate editor coordinating the review of this manuscript and approving it for publication was Junho Hong.

TABLE 1. Comparison in terms of scale between our work and related work on P2P energy sharing community formation within *long-term communities*, as well as cost-optimization within *short-term coalitions* (†) based on real-world consumption data.

Year	Authors (reference)	Target objective(s)	Size of communities / household pool	No. of *formed* trading groups	No. of *possible* communities
This work (1 week of data;§ VI-C)		Total cost-saving	2-5 / up to 115492	up to 23140	$4.65 \cdot 10^{18}$
This work (1 year of data;§ VI-B)		Total cost-saving & global cost	2-5 / 2221	444 to 1110	$4.17 \cdot 10^{11}$
2023	Chau *et al.* [2], [3]	Total & individual cost-savings	2-3 / 31	10 to 15	4495
2021	Azim *et al.* [4] †	Individual cost-savings	2-36 / 36	1-18	1[a]
2021	Duvignau *et al.* [5]	Total cost-saving	2-100 / 100	1-50	4950
2021	Tushar *et al.* [6] †	Peak demand & energy cost	2-12 / 12	1-6	1[a]
2020	Duvignau *et al.* [7]	Total cost-saving	2-100 / 100	1-50	4950
2019	Tushar *et al.* [8] †	Individual cost-savings	2-10 / 10	1-5	1[a]
2019	Heinisch *et al.* [9]	Total cost-saving	2-101 / 101	1	1
2019	Khorasany *et al.* [10] †	Individual cost-savings	7 / 7	1	1
2019	Chau *et al.* [11]	Total & individual cost-savings	2 / 31	15	465
2018	Lüth *et al.* [12] †	Total cost-saving	4 / 4	1	1
2018	Long *et al.* [13]	Total cost-saving	100 / 100	1	1

[a] Coalitions apply to a single trading slot, hence over a billing period all peers may perform exchanges with each other and thus belong to the same community.

to reduce their energy bill while increasing energy efficiency and self-reliance within the communities. Locally in each community, energy is either ''traded'' at regular intervals (e.g. every hour) or ''exchanged for free'' with a later gratification scheme. P2P energy sharing bypasses the centralized grid by letting the peers cooperate in a distributed fashion in order to share in the best way their energy resources. Hence, to lower their cost, households are encouraged to use as much as possible the electricity that is generated locally instead of that from the grid, with local benefits due to reduced tax fees, as well as an increase in local self-consumption. There are many challenges that are still required to be resolved before P2P energy sharing becomes the norm [24], [25], [26], [27] and in particular robust communication protocols [17], data protection and privacy [28], legal and economic challenges [29], how to handle payments [30] as well as alleviating security concerns [31].

This work focuses on one challenging task that remains to be dealt with before P2P energy sharing over large systems can be a reality: how to efficiently organize a large number of end-users into independent sharing communities.

Short-term *coalitions* (lasting, for example, 10 minutes) are typically formed using game-theoretic approaches [8] and take into account only the current state of the system (e.g., the amount of electricity being produced and consumed by various end-users, as well as market prices).

Forming *long-term communities* (used over weeks, months or even years [3], [11]) is an ever more rewarding and challenging task. Often also associated with geographical closeness, the long-term communities present many advantages from an infrastructure point of view, being able to better regulate local load-balancing of distributed energy resources, providing higher local self-sufficiency and/or reducing the local peak demand. Chau et al. [11] have investigated *stable* partitioning, i.e., a given partitioning can be rejected if any group of peers would gain more by forming a different one, similar to the *stable marriage* [32] problem for pairs. In [2] and [3], the partitioning is expanded to communities of size $k > 2$ and partition-forming algorithms are evaluated for groups of size 2 or 3 over a set of 30 households. Duvignau et al. [5], [7] show that small-scale communities made of a few peers only (2 to 5) are both efficient in terms of data and cost. As the authors essentially focused on energy cost-optimization (increasing benefits for end-users) and data-efficiency (decreasing amount of shared data), no mathematical analysis is performed concerning the cost of computing the different matchings. In all the aforementioned works, the computed long-term partitions in the experimental study do not involve a high number of nodes (100 at maximum) or larger neighborhoods than those of size 3, nor do they use information about the geographical positions of the nodes in calculating the partitioning.

A. MOTIVATIONS

We summarize previous works dealing with formation of P2P energy sharing communities and coalitions in Table 1. Current state-of-the-art [3], [5], [9], [12] relies on exhaustive search to compute the optimal solution from datasets containing only a small number of nodes, whereas our aim is to provide efficient algorithmic solutions for a large pool of households. To appreciate the difference in scale between the empirical evaluation of our work (and exemplify its scalability properties that is also analyzed in this article) and previous studies (cf. Table 1), the problem, for example with about 2000 households, implies 100 billion possible communities to choose from (when limited to communities of up to 5 members). Using a week-long dataset covering 115,492 users, the number of possible communities reaches an astronomical scale of several quintillion – that is, billions of billions of possible communities.

To scale up towards such large systems, we introduce the *Geographical Peer Matching* (GPM) problem that consists in forming the energy communities based on both geographical information about the peers as well as their local matching preferences. This provides a natural way of reducing the search space but as we show in this work, dedicated

algorithms are still required to cope with the computational complexity of the GPM problem. Indeed, as aforementioned, even when enforcing geographical constraints, there may exist billions of possible communities when the set search radius is not restricted to a small value.

B. CHALLENGES AND RESEARCH QUESTIONS

The main challenges in establishing efficiently P2P energy sharing communities are threefold: (i) peers continuously produce data and have limited knowledge of how their future local data will look like, (ii) computing peers' optimization options for preferences requires both access to the relevant data and the execution of a Linear Programming (LP) solver on a large input (cf. [5], [7] and Section III-B) and (iii) peers should favor getting matched with geographically neighboring peers to reduce transmission losses and the impact on the underlying infrastructure. In particular, weights in the matching (i.e., the local matching preferences) are not known at the system's start but must be computed on the fly, and this requires additional communication between the participants. An additional difficulty stems from allowing the formation of groups of size 4 and above, as the matching problem becomes intractable in this case and thus in practical systems, it is not feasible to do exhaustive searches any longer. This raises the following research questions: (i) Can one translate the challenges to a formal model that can capture the benefits and complexity for the prosumers? (ii) How do the maximum size and geographical diameter for communities influence the cost-efficiency of the peer matching? (iii) Is it possible to design matching algorithms that are efficient in terms of cost (with good quality for the solution) and scalable (with low computational burden) and where is the best trade-off?

C. CONTRIBUTIONS

We present partitioning mechanisms between consumers and prosumers to form P2P energy sharing communities. One core contribution is the presentation of a mathematical modelling of the GPM problem expressed as finding a maximum weight bipartite partitioning in hypergraphs. We establish bounds on the inherent complexity of the problem, demonstrating its inapproximability within a given factor and highlighting the challenges in achieving efficient solutions. With Proposition 3, we provide an important theoretical result for the smart grid research community, formally lower- and upper-bounding the benefits that can be extracted from a prosumer by a coordinated group rather than independent individuals. We further propose and analyze different algorithms that solve the GPM problem. We study the trade-off between cost- and computational-efficiency of different algorithms and parameters based on an experimental study involving consumption data from 2,221 real households and realistic solar profiles (over an entire year), and a realistic distribution of renewable energy resources among peers. Our analysis and proposed solutions address the challenges of the GPM problem and can guide P2P energy-sharing communities toward both cost efficiency (providing significant savings to all peers) and scalability (capable of accommodating pools of hundreds of thousands of users).

Implementing these solutions into a real-world setting is an exciting prospective follow-up of our work, however, it is noteworthy to highlight that our study focuses on demonstrating that our solutions are both computationally feasible and efficient.

D. PLAN

The next section discusses relevant related work. Section III presents in more detail P2P energy sharing and cost-optimization of distributed resources in this context. Section IV presents our mathematical modelling of the GPM problem and some of its variants. In Section V we present our algorithms and analyze their computational overhead. In Section VI, we present an extensive experimental study of the performance of the algorithms based on electricity data from real-world households. Finally, Section VII presents our conclusions.

II. RELATED WORK

A. P2P ENERGY SHARING

P2P energy sharing has been in the focus of numerous research works in recent years (cf. the comprehensive surveys [17], [18], [33] and references therein). Among those works, one can distinguish two types of P2P sharing communities. Short-term coalitions (lasting for e.g. 10min) are formed to cover a single timeslot where game-theoretic approaches are used to optimize the individual gains among other things [8]. On the contrary, long-term coalitions [3], [5], [9], [13] seek to form communities that will take coordinated decisions within the same group of peers over months to years.

Research on optimizing the gain outcome from the peering process using constrained optimization has focused on different aspects: how communications are handled [13], reaching stable partitions [3], [11], finding optimal resources based on community sizes [9], privacy aspects and amount of data transmitted over the network [5], [7], [34], etc. Small neighborhoods have been shown to provide a high share of possible gain while being favorable in terms of data exchanges [5] and different matching algorithms appear in [3] and [11], based upon stable partitions (i.e., nodes with self-interest) and on a cost-sharing mechanism known beforehand. However, the algorithmic problem behind the formation of localized and long-term P2P energy sharing communities with a global objective has not been studied before to the best of our knowledge, neither analytically nor from a practical point of view.

B. HYPERGRAPH MATCHING

The solutions explored in this paper relate to the *k-bounded Maximum-Weight Hypergraph Matching* (k-MWHM) problem, where k refers hereafter to the maximum size of an

hyperedge in the input hypergraph, especially to the works proposing mechanisms to compute practical solutions. The k-MWHM problem does not permit a polynomial time $o(k/\log k)$-approximation unless P = NP [35]. All the best heuristic algorithms that have been proposed to solve the problem are based on the notion of "local search" [36]. Local search consists in incrementally improving a starting solution by performing a series of small changes (typically switching membership of a node from one partition to another), that is known to be competitive to the optimal algorithm. A local search algorithm is used for example in [37] to solve greedily the weighted k-set problem while showing that the introduced algorithm is $2(k+1)/3$-competitive. One of the most competitive algorithms for k-MWHM is given by Berman [38], providing a $(k+1)/2$-approximation algorithm. The algorithm is based on finding independent set in d-claw free graphs, and it requires a series of prepossessing steps in order to solve the k-MWHM problem, e.g. [39] makes these steps explicit to obtain a hypergraph matching algorithm from Berman's algorithm. Other works [40], [41] have used a similar pipeline to solve greedily the k-MWHM problem based on local search. Recent improvements on Berman's algorithm are given in [42], [43], and [44]. k-MWHM has also been studied in a distributed setting [45], as well as under "b-matching" generalizations [46] where nodes can appear in b different hyperedges instead of one only.

All local search-based algorithms, however, assume access to the full list of weights since their starting point is the output of what we referred here as the "Classic Greedy" algorithm. Also, all those algorithms have a time complexity that is exponential in k [37]. In [47], computing all the weights was already identified as a computational bottleneck when the number of possible hyperedges is large, proportional to $\mathcal{O}(n^k)$ when there are n nodes in the input hypergraph. The authors hence propose a heuristic algorithm specific to their problem that only necessitates to compute $\mathcal{O}(kn^2)$ weights using an algorithm of time-complexity $\mathcal{O}(kn^3)$. For the number of computed weights, this matches the same bound shown here when using to what we refer as "memoryful weights" as heuristic input, and we further reduce it to only $\mathcal{O}(n^2)$ weights while introducing the notion of *memoryless weights* and exploiting pairwise weights as heuristic basis for the local preferences. In contrast, all the algorithms studied in this work have also noticeable smaller time complexities, namely $\mathcal{O}(kn^2)$, $\mathcal{O}(n^2)$ and $\mathcal{O}(n^2 \log n)$. Note the problem studied here adds three additional real-world motivated and computationally-demanding constraints to classical hypergraph matching: requirement for a bipartite solution, spacial constraints and a challenging environment where the weight of a hyperedge is computationally expensive to obtain.

C. THE INFLUENCE OF PROCESSING ORDERS

Recently, Duvignau and Klasing [48] and Duvignau et al. [49] demonstrated several analytical results for achieving bounded approximations for the assignment problem (i.e., bipartite graph matching) using greedy algorithms that focus on both producing a good matching while inspecting only $\mathcal{O}(n)$ weights. Their analysis assumes that the input graph is processed in an heuristic order where edges associated with higher weights are usually processed earlier by the matching algorithm. This completely matches our setting where prosumers with larger consumptions and renewable resources are more likely to produce more cost-saving benefit, and thus their results entail that formal performance guarantees can also be derived for the heuristic algorithms that we have introduced in § V. In addition, their results combined with our formal proofs given in § IV-C2 and § IV-D, imply that one of their introduced algorithms provides a $(k-1)\cdot\varepsilon$ approximation of the optimal (where ε can become arbitrary close to one for well chosen processing orders on favorable instances) while only calculating $\mathcal{O}(kn)$ weights.

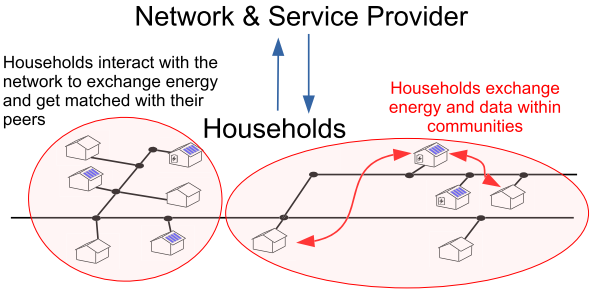

FIGURE 1. Overview of P2P energy sharing showing grouping and interactions between prosumers (equipped with PV panels on roof top and optionally a battery system) and traditional consumers (without any energy resources).

III. SYSTEM MODEL

We define here the notion of P2P energy sharing communities, its underlying assumptions and consequences. We then define how to optimize peers cost in such a context and present how this translates into a matching problem when several communities are considered.

A. P2P ENERGY SHARING

Let us recall the basic requirements of the traditional energy infrastructure, considering only electricity as energy source for the purpose of simplicity.

Consumers, which can be individual households or virtual entities aggregating consumptions over a building or neighbourhood of energy-management systems, must match their consumption by importing from the grid their electricity demand, whereas prosumers use in priority their local generation (e.g. PV panels); in case of surplus, the energy is sold to the grid, while in the opposite situation, prosumers must import electricity from the grid.

The situation complicates for prosumers equipped with both electricity production and storage, as they need to take online decisions whether to store the surplus or sell it to the grid, and whether to use stored electricity or rather buy it from the grid.

TABLE 2. Nomenclature used in the paper (P2P Energy Sharing variables, constants and functions in the left column, graph symbols/functions in the right column).

Symbol	Usage	Symbol	Usage
$\mathsf{cost}(h,t)$	electricity cost for user h at hour t (€)	V	pool (set of all households), $V = P \cup C$
$\mathsf{bill}(h,[t_0,t_r])$	electricity cost for h over period $[t_0,t_r]$ (€)	P	set of prosumer households
$\mathsf{el}_{in}(h,t)$	amount of electricity bought from the grid by h at hour t (kWh)	n	number of prosumers, i.e., $\lvert P\rvert$
$\mathsf{el}_{out}(h,t)$	amount of electricity sold to the grid by h at hour t (kWh)	C	set of consumer households
$\mathsf{el}_{cons}(h,t)$	consumption (or electricity demand) of user h for hour t (kWh)	m	number of edges, i.e., $\lvert P\rvert \cdot \bar{\delta_\Delta}$
$\mathsf{el}_{gen}(h,t)$	electricity generated by h during hour t (kWh)	Δ	geographical search radius
$\mathsf{bat}(h,t)$	battery level at time t for prosumer h (kWh)	k	maximum size for communities
$\mathsf{price}(t)$	price of electricity at hour t (€/kWh)	$\bar{\delta_\Delta}$	average neighborhood size
$\mathsf{sun}(t)$	the sun's intensity at hour t (kWh/kWp)	$w(e)$	weight of the hyperedge e
tax	relative tax level on top of market-price (e.g. 25%)	$\mathsf{dist}(v,v')$	geographical distance between v and v'
el_{tax}	fixed electricity tax added on top of market price (€/kWh)	E_Δ	pairs of $E \subseteq P \times C$ within distance Δ
el_{net}	small payment for selling electricity to the grid (€/kWh)	WA_t	cost-based memoryless weights at time t
PV_h	PV capacity for prosumer h (kWp)	WB_t	cost-saving memoryless weights at time t
B_h	battery capacity for prosumer h (kWh)	WC_t	cost-based memoryful weights at time t
$\mathsf{gain}(G,[t_0,t_r])$	cost saving of community G over period $[t_0,t_r]$ (€)	WD_t	cost-saving memoryful weights at time t

Set in the context of increasing decentralization of the energy infrastructure, *P2P energy sharing* (see Fig. 1) consists in forming local *energy communities* of cooperative end-users. The goal is to make the most of the distributed resources and hence achieve even greater cost reductions. However, this means that participants need now to consider also the state and decisions of the other actors in order to coordinate and optimize the benefits of their local resources. In such energy communities, any energy consumption can be offset by importing the equivalent amount of electricity from a peer; such an exchange may get instantly gratified leading to a local trading market [50], [51]. At regular time intervals (e.g. 1 hour), the community needs to coordinate the usage of energy resources (e.g. which battery system(s) to charge or discharge and by how much, and by consequence how much energy needs to be traded with the central grid).

In the literatures [3], [5], [9], and [13], independent *long-term communities* of end-users that can last for month(s) or year(s) have been introduced to alleviate the inherent infrastructure cost to allow and maintain energy exchanges among different end-users.

At the end of predetermined billing periods, each user pays an electricity bill taking into account their own consumption and all exchanges that occurred within the said period.

B. SINGLE-USER COST-OPTIMIZATION (LP-SOLVER)

The cost-optimization problem consists in minimizing the yearly electricity bill for a particular end-user, based on locally available electricity data: amount of consumption, generation and price. In order to minimize the cost, we adopt the following LP-formulation cost-optimization following similar models used in the recent literature (cf. [5], [9], [11]). The electricity cost $\mathsf{cost}(h,t)$ of the end-user h at hour t is assumed to be as follows (cf. Table 2 for definitions of all variables used hereafter):

$$\mathsf{cost}(h,t) = \mathsf{el}_{in}(h,t) \cdot \big(\mathsf{price}(t) \cdot (1+\mathsf{tax}) + \mathsf{el}_{tax}\big) - \mathsf{el}_{out}(h,t) \cdot \big(\mathsf{price}(t) + \mathsf{el}_{net}\big), \quad (1)$$

We assume $\mathsf{el}_{net} < \mathsf{el}_{tax}$ and $\mathsf{tax} \geq 0$.

Pure consumers do not have any generation or storage resources, hence the yearly cost is obtained directly from their consumption, that is $\mathsf{el}_{in}(h,t) = \mathsf{el}_{\mathsf{cons}}(h,t)$ for all hours t. Prosumers with only electricity generation but no storage have always interest to use in priority their local production to avoid to pay tax on electricity coming from the grid; hence, they optimize their cost by setting:

$$\mathsf{el}_{in}(h,t) = \begin{cases} \mathsf{el}_{gen}(h,t) - \mathsf{el}_{cons}(h,t), & \text{if } x_{h,t} > 0, \\ 0, & \text{otherwise;} \end{cases}$$

$$\mathsf{el}_{out}(h,t) = \begin{cases} \mathsf{el}_{cons}(h,t) - \mathsf{el}_{gen}(h,t), & \text{if } x_{h,t} < 0, \\ 0, & \text{otherwise;} \end{cases}$$

with the electricity balance $x_{h,t} = \mathsf{el}_{gen}(h,t) - \mathsf{el}_{cons}(h,t)$ where $\mathsf{el}_{gen}(h,t) = \mathsf{PV}_h \cdot \mathsf{sun}(t)$. Prosumers having both electricity generation and storage can optimize their cost over a period of time from t_0 to t_r through running an LP solver of the following formulation:

- **Objective function:**
  minimize $\mathsf{bill}(h,[t_0,t_r]) = \sum_{t=t_0}^{t_r} \mathsf{cost}(h,t)$.
- **Constraints (for all $t_0 \leq t \leq t_r$):**
  $0 \leq \mathsf{bat}(h,t) \leq \mathsf{B}_h$, and
  $\mathsf{bat}(h,t) = \mathsf{bat}(h,t-1) + \mathsf{el}_{gen}(h,t) - \mathsf{el}_{cons}(h,t) + \mathsf{el}_{in}(h,t) - \mathsf{el}_{out}(h,t)$.
- **Optimization variables:**
  $\{\mathsf{bat}(h,t), \mathsf{el}_{in}(h,t), \mathsf{el}_{out}(h,t) \mid t_0 \leq t \leq t_r\}$.

with $\mathsf{bat}(h,t_0-1)$ indicating the initial battery level.

This simplified model introduced in [5] as "aggregate model" does not account for transmission and battery losses. Note that in order to take continuous online decisions concerning usage of one's battery system, one would need to use forecast data as input to the optimization problem as in [5], [7], and [13].

C. COMMUNITY COST-OPTIMIZATION

One may wonder how the end-users can in this context make the most of their distributed resources. Instead of fixing one of the many forms of a local trade market, we consider as in e.g. [5], [7], [9] that the energy exchanges occur "for

free" within the community to obtain and analyze the lowest achievable cost as a community, while postponing the billing of individual exchanges to the end of the billing period. Hence, our focus in this work is on finding communities that reach the best benefits overall. A subsidiary mechanism can occur when the billing period ends to distribute the gain achieved by the community among the peers. Under the above assumptions and neglecting battery and transmission losses and communication issues, each community becomes equivalent to a single prosumer with aggregated PV and battery capacities over the full community.

1) EXAMPLE

Let us consider as an example a particular grouping of 6 households $V = P \cup C$, including 1 battery-equipped prosumer p_1 among 3 prosumers $P = \{p_1, p_2, p_3\}$ and 3 consumers $C = \{c_1, c_2, c_3\}$, as the one on the left of Fig. 1. Neglecting losses and communication faults, the community is then equivalent to a single larger entity with aggregated consumption from the 6 households, aggregated production from the 3 prosumers and having as much battery storage as p_1.

Suppose now the annual cost (i.e. $T = [t_0, t_r]$ spans one year) of each household $h \in V$ was $\mathsf{bill}(h, T) = 1000€$ each, for a total of 6000€. As a single community taking coordinated decisions, they may only have to pay 4800€, or 800€ each if the 1200€ gain is spread equally among participants. We use hereafter *cost saving* for the reduction in cost obtained through cooperation, e.g. 1200€ in this example.

Definition 1: Let the cost saving *(or* gain*) for the community G be defined as*

$$\mathit{gain}(G, [t_0, t_r]) = \sum_{x \in G} \mathit{bill}(x, [t_0, t_r]) - \mathit{bill}(\bar{G}, [t_0, t_r]),$$

where $\bar{G}$ is an aggregated prosumer equivalent to G, defined by: $\mathit{PV}_{\bar{G}} = \sum_{x \in G} \mathit{PV}_x$, $\mathit{B}_{\bar{G}} = \sum_{x \in G} \mathit{B}_x$, and for each $t \in [t_0, t_r]$, we have $\mathit{el}_{gen}(\bar{G}, t) = \sum_{x \in G} \mathit{el}_{gen}(x, t)$, $\mathit{el}_{cons}(\bar{G}, t) = \sum_{x \in G} \mathit{el}_{cons}(x, t)$.

D. FROM COMMUNITIES TO PARTITIONS

Among a *pool* (set of end-users willing to participate in P2P energy sharing), several communities can be managed independently from each other. How to partition efficiently a group of users into independent communities is the main focus of the present work. For any given partitioning (e.g. the partition of the households of Fig. 1 into 2 communities, one of size 6 and one of size 5), we can associate a *global cost saving* corresponding to the sum of the cost savings of each community. Optimizing cost-efficiency of all resources in a given pool corresponds, from a centralized perspective, to maximizing the global cost saving. One can thus summarize the problem of forming P2P energy communities as follows:

- **Input:** (forecast or historical data for) (1) energy consumptions over timespan $\mathcal{T} = [t_0, t_r]$ for each household $h \in V$, where $V = P \cup C$ is made of a set P of prosumers and C of consumers; (2) local solar intensity (depending on geographical location) over $\mathcal{T}$; (3) electricity prices (possibly set at regional level) over $\mathcal{T}$.
- **Ouput:** A partition M of $P \cup C$ into independent groups (whose size may be limited by a certain constant k).
- **Metrics of interests:** (1) amount of global cost saving obtained by the partition M (that is, the sum of the communities' gains, i.e., $\sum_{G \in M} \mathsf{gain}(G, [t_0, t_r])$, cf. Definition 1); (2) computational overhead of calculating M.

E. FROM PARTITIONS TO GEOGRAPHICAL PEER MATCHINGS

The search space for possible partitions of the pool of households into independent communities is large and much beyond computational capabilities of any real system of reasonable size. We hence propose several basic restrictions on the possible partitions to reduce the combinatorial possibilities. First, communities made uniquely of consumers and single-node communities can be discarded as they provide absolutely no benefit in terms of cost-saving. Second, following results from previous works in the area [5], [9], we include communities made of a single prosumer, as allowing several prosumers provides very little to no advantage in terms of global cost saving in comparison to splitting such group into two or more smaller communities. Third, we propose to limit communities to be only made of nearby households by restricting the maximal distance between the prosumer and any consumer in a given community to be within a certain constant Δ (hence any household of a community is always within 2Δ of each other). Limiting the search radius to Δ allows the service provider managing the P2P energy system to use aggregators[1] in charge of smaller geographical areas. Remote control of the end-users' distributed resources could in turn be delegated to such aggregators relieving the users of data exchange and computational work during the P2P energy sharing process. In addition, having geographically closer communities allows to have better independence and load-balance in the system, further reducing the impact of introducing sharing communities on the underlying grid infrastructure. Hence, the problem of forming communities reduces to finding a *matching* (or assignment) of nearby consumers to each prosumer in the system, and we thus refer to it as the *Geographical Peer Matching* (GPM) problem, further formalized and defined in Section IV.

In the GPM problem, peers are matched together based on their current *preferences* $w_t(G)$ at time t, which indicate e.g. the potential saving of a certain community G at time t. Since data is not known ahead of time, there are two strategies to compute $w_t(G)$: either using only past data, i.e.,

[1] Intermediate infrastructure level between end-users and service provider where data can be retrieved almost in real time and with fine granularity.

w_t is computed based on data recorded within some timespan $[t-\tau, t]$ for some τ, or using past and projected data.

Since our main focus in this work is on communities lasting for a long period of time (months to years), the preference calculation based on projection is not the most appropriate solution as the accuracy of the forecast degrades fast as the horizon grows (e.g. poor prediction is expected past 48h). In this context, peers' affinity is more reasonably captured by setting τ to the same length as the billing period (e.g. one year), however, note that our algorithmic approach will work as well for shorter-term communities (lasting e.g. days to weeks) and using predictions for calculating the peer preferences.

Also, we would like to highlight that a long time horizon for the duration of the energy communities does not imply that one can disregard the computational burden of calculating the peer matching. Indeed, the GPM problem is computationally hard to solve (cf.§ IV-B) and calculating an optimal solution for a large instance is considerably beyond the capacity of the computing resources available at the service provider for P2P energy-sharing. In addition, in order to compute $w_t(G)$, necessary data must be transmitted (enduring some communication overhead) and a run of an LP-solver (requiring computation overhead) is required to solve the cost-optimization problem (see § III-B and § III-C). Hence, the computation of a single preference $w_t(G)$ is therefore both costly in terms of data exchanged and local computation.

The goal of our approach is thus for the group matching to be computed while keeping the amount of preference-computation low.

F. GENERAL SYSTEM CONSIDERATIONS

We assume customers can see and use their own data, but do not have access to their peers' data. To minimize data exchanges and reduce stress on the architecture, we assume the matching happens at a higher level through a third party dedicated entity. This centralized point of view assumes the end-users have subscribed to such an external service provider (being the energy provider or a third agent) in order to participate in the sharing process, and paying for the service through a fixed share of the cost saving obtained by each community. Under this setting, the service provider is in charge of grouping prosumers and consumers and supervising the transmission of data needed for the matching; its goal is thus to achieve the best global cost saving to maximize its own benefit as well. This also means that, once a matching has been decided and propagated to the peers by the system, communities can then work in an independent fashion. In particular, they do not further need to rely on the service provider for managing their operate day exchanges, neither for optimizing their electricity cost. Since end-users change their consumption patterns and may revoke their will to participate through time, it may be beneficial to recompute the matching after a billing period has elapsed.

G. SECURITY AND PRIVACY IN OUR SYSTEM MODEL

Security [31], [52], [53] and privacy [28], [54], [55], [56], [57] aspects are important recurring concerns in relation to P2P energy sharing. We briefly detail hereafter the implications of our system model in those regards.

1) SECURITY ASPECTS

Concerning potential increase in cyber threats over smart grids, since we only focus in our study on the group formation step, there is no assumption of a direct control neither over the prosumers' energy resources nor on their interactions with the grid. Instead, we rely on historical data of the participant peers to compute the best possible groupings. Daily management and coordination within the communities is delegated to an auxiliary service that can subsequently be built with support of the rich literature covering P2P energy sharing systems [18]. In a decentralized group formation, malevolent end-users could also lie about their local matching preferences. Note, however, as we use in this work a global optimization criterion, mistrustful end-users can only have an impact and push the cost savings down in groups where they appear with a limited impact overall if their number is small relative to the size of the household pool.

2) PRIVACY ASPECTS

To participate in P2P energy sharing, participants are assumed to be willing to share information about their energy generation and consumption in exchange of the cost reduction offered by the system. In our model, we only rely on historical data to compute the partition of peers into the different communities, and since this can be operated by a service operator, sharing of the involved data with only the service provider is required avoiding sharing it with all the other participants in the network. If necessary, a possible decentralized privacy-preserving scheme as proposed in [5] can be set, removing the possibility for even the service provider to have access to individual end-user electricity consumption and generation data, and instead only the aggregation over small groups is known to the managing entity. Differential privacy [58], [59], [60] has also been proposed to mitigate peers' privacy concerns.

IV. ALGORITHMIC MODELING FOR GEOGRAPHICAL PEER MATCHINGS

We present in this section a formalism for the GPM problem in terms of finding a maximum-weight matching in bipartite hypergraphs. We then present how to measure the complexity of algorithms solving the problem and how certain considerations on the behaviors of the peers' preferences can simplify the GPM problem.

A. THE GPM PROBLEM ABSTRACTION

1) PRELIMINARIES

A hypergraph $\mathcal{G} = (V, E)$ is made of a set of vertices V and a set of hyperedges $E \subseteq 2^V \setminus \emptyset$, where a hyperedge $e \in$

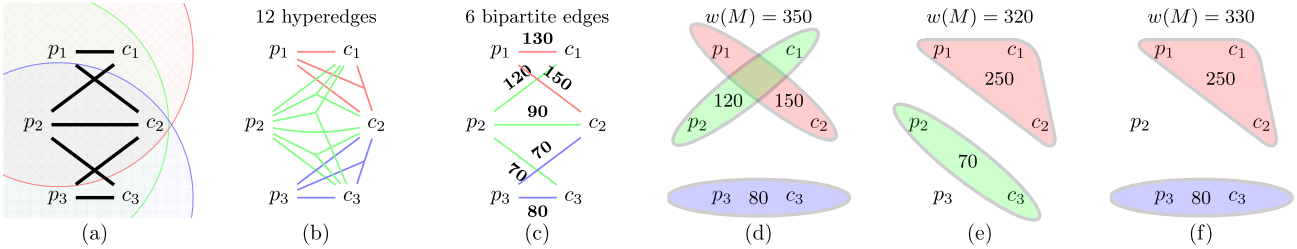

FIGURE 2. Illustration of the matching procedures with $P = \{p_1, p_2, p_3\}$, $C = \{c_1, c_2, c_3\}$ and $k = 3$: (a) authorized edge set E_Δ for a given search radius Δ, (b) all hyperedges based on E_Δ, (c) pairwise weights, and (d) Round Robin, (e) Single Pass, and (f) Classic Greedy matchings.

E is any non-empty subset of vertices. A hypergraph is said to be *weighted* if the hypergraph is associated with a weight function $w : E \to \mathbb{R}$. We say a hypergraph is k-bounded if all its hyperedges are of size at most k; a 3-bounded hypergraph is displayed in Figure 2(b).

A matching M in a hypergraph $\mathcal{G} = (V, E)$ is a set of *disjoint hyperedges* (e_1 and e_2 are considered disjoint when they do not share any vertices, i.e., $e_1 \cap e_2 = \emptyset$). The weight of such a matching is the sum of the weights of the selected hyperedges that it contains, i.e., $w(M) = \sum_{e \in M} w(e)$, slightly abusing the w-notation. Given a partition of the vertices into two disjoint sets P and C, a *bipartite hypergraph matching* (BHM) is a matching of a hypergraph $\mathcal{G} = (P \cup C, E)$ that contains in each selected hyperedge exactly one vertex within P, i.e., M is bipartite if $\forall e \in E, |e \cap P| = 1$. A k-bounded matching ensures that each output group is of size at most k; 3-bounded BHMs are displayed in Figure 2(d)-(f).

2) GEOGRAPHICAL PEER MATCHING (GPM)

We define the *Maximum-Weight Bipartite Hypergraph Matching* (MWBHM) problem that consists in finding a bipartite hypergraph matching with maximum weight in a given hypergraph over vertex set $P \cup C$. The problem becomes dynamic when the weighting function w is time-dependent w_t and the matching problem should be solved for each time step. The *GPM* problem of parameter (k, Δ), for $k \geq 2$ and $\Delta > 0$, written in short form as (k, Δ)-GPM, is a bounded version of the MWBHM problem where one adds the following three additional conditions:

1) **Neighborhoods**: M is a k-bounded bipartite hypergraph matching, i.e., $\forall e \in M, |e| \leq k$.
2) **Spatiality**: peers are geographically distributed and the matching M should adhere to each peer's locality: every consumer of a selected hyperedge of M must be within geographical distance Δ of its matched prosumer, i.e., $\forall e \in M, \{p\} = e \cap P, \forall c \in e \cap C, \ \mathsf{dist}(p, c) \leq \Delta$.
3) **Computationally-intensive weights**: weights are assumed unknown beforehand but must be dynamically calculated (cf. § III-E), and the function is considered expensive (due to inherent communication and computational costs) and the main bottleneck of the system. Complexity of an algorithm solving the GPM problem is thus mainly measured in the number of **weight** computations, as further explained in § IV-B.

3) P2P ENERGY SHARING AS A HYPERGRAPH MATCHING PROBLEM

We model the problem of forming P2P energy sharing communities in a continuous fashion as a (dynamic) GPM problem of parameters k and Δ. The underlying k-bounded hypergraph is $\mathcal{G} = (V, \mathcal{E}_\Delta)$ where the vertex set $V = P \cup C$ is made of P, the set of prosumers (users equipped with renewable energy resources) and C, the set of consumers (with no resources). The set of hyperedges $\mathcal{E}_\Delta$ is made of all possible hyperedges that can be part of a k-bounded BHM and consumer-prosumer pairs are pairwise within distance Δ (where k represents the maximum allowed size for the communities), i.e., $\mathcal{E}_\Delta \subseteq \{e = (p, X) \in P \times 2^C \mid 2 \leq |e| \leq k, \ \Delta \geq \max_{c \in X} \mathsf{dist}(p, c)\}$. The weighting function $w_t(G)$ is dynamic and can be used to e.g. capture the cost saving of a particular community G at time t (cf. § V-D). Then, adding the hyperedge G in the BHM is equivalent of forming the P2P energy sharing community G.

B. COMPLEXITY OF THE GPM PROBLEM

1) METRICS OF INTEREST FOR THE PROBLEM

The GPM problem belongs to the family of *Discovery Problems* [48] where the input is not entirely accessible at the algorithm's start but must be queried during its execution. In the case of the GPM problem, the discovery deals with the weight of hyperedges (i.e., possible communities in the P2P energy sharing context) and as the main bottleneck in computing a solution, the number of calls to the **weight** function is one of the main complexity measures to evaluate an algorithm, along with the quality of the solution. Indeed, one should not forget that GPM is essentially a sub-variant of MWBHM and hence its primary goal is still to find the best possible matching of the hyperedges. GPM can therefore be considered as a *bicriteria* optimization problem: maximize the weight of the matching and minimize the number of calculated weights.

Hence, considering the difficulty of finding the optimal solution (cf. Proposition 1) and the additional assumption on computational-intensive weights, an algorithm $\mathcal{A}$ that produces a matching M as solution to the GPM should be

evaluated on two main criteria: (1) the **number of weights** that were computed by $\mathcal{A}$ in order to find M, and (2) the **quality of the solution** indicated by $w(M)$.

One should observe that minimizing the number of weight calculations alone is trivial by just producing an arbitrary valid matching without calling the weight function. Let's note that finding the minimum number of weights that need to be computed to reach a fraction α of the weight of the optimal matching is out of the scope of the present work.

Definition 2: We call neighborhood *of node $v \in P$ the set*

$$N_\Delta(v) = \{v' \in C \mid \textit{dist}(v, v') \leq \Delta\},$$

and denote the average neighborhood size *by*

$$\bar{\delta_\Delta} = \frac{1}{|P|} \sum_{v \in P} |N_\Delta(v)|,$$

used for complexity computations in the remaining.

EXAMPLE

For instance, assuming that the nodes are spread uniformly and independently on a square zone of size $L \times L$ with toroidal properties to simplify,[2] it is easy to calculate the average neighborhood size as $\bar{\delta_\Delta} = m \cdot \pi(\Delta/L)^2 = \Omega(m)$ for $L = \Omega(\Delta)$ where $m = |C|$ (recall Table 2 contains definitions for all variables).

2) COMPUTATIONAL COMPLEXITY FOR THE PROBLEM

Proposition 1: For $k \geq 4$, the (k, Δ)-GPM problem is not approximable within a factor of $o(k/\log k)$ in polynomial time, unless P = NP.

Proof: In general, the hypergraph matching (HM) problem that consists in computing the maximum-weight matching of hyperedges does not permit a polynomial time $o(k/\log k)$-approximation unless P = NP [35]. There is a trivial reduction to the weighted k-set packing problem, known to be NP-complete from Garey and Johnson [61]. Indeed, a hyperedge is nothing more than a subset of the vertex set, and the k-set packing problem [37] is looking for a maximum weight sub-collection of disjoint sets, which is equivalent to finding a maximum-weight matching of non-overlapping hyperegdes.

Now, adding the bipartite constraint is not reducing the difficulty of the problem, as one can reduce the GPM problem to HM as follows. Let $\mathcal{G} = (V, E)$ be a k-bounded hypergraph and let $\mathcal{G}' = (V_1 \cup V_2, E')$ be a hypergraph that contains all vertices of $\mathcal{G}$ plus $|E|$ "extra vertices", i.e., $V_1 = \{v_e \mid \forall e \in E\}$ and $V_2 = V$. Then all hyperedges of $\mathcal{G}'$ are made of those of $\mathcal{G}$ with one extra vertex in each, that is $E' = \{e \cup \{v_e\} \mid e \in E\}$ with $w(e \cup \{v_e\}) = w(e)$ for every $e \in E$. Any matching M of E' is only made of edges containing exactly one vertex of V_1 each; therefore, it is a bipartite hypergraph matching under our definition. In turn, M is also a matching of $\mathcal{G}$ (just remove the extra vertex in each hyperedge). Lastly, we can easily affect the position of each vertex so that they are all within distance $\Delta > 0$ from each other, thus defeating the additional spatial constraint. □

Proposition 1 states that for cases where $k \geq 4$, no polynomial-time algorithm can guarantee an approximate solution for the (k, Δ)-GPM problem within a factor better than $o(k/\log k)$, unless P = NP. In practical terms, this means that as k grows, the difficulty of finding even a reasonably close solution increases significantly, underscoring the inherent computational intractability [62] of the problem and justifying the introduction of dedicated algorithms in § V.

When forgetting the spatial constraint, we note that for $k = 2$, the problem becomes polynomial and is equivalent to finding the maximum-weight matching in a weighted bipartite graph, usually then named "the assignment problem". This is a classic problem where the Hungarian algorithm [63] provides the optimal solution in time $\mathcal{O}(nm + n^2 \log n)$ for n vertices in the smaller vertex set and m edges, cf. [64]. Now adding spaciality and assuming $\bar{\delta_\Delta}$ as the average neighborhood size for search range Δ, this gives a running time of $\mathcal{O}(n^2\bar{\delta_\Delta} + n^2 \log n) = \mathcal{O}(n^3)$. We note that such a running time can be already prohibitive for large n and even for $k = 2$, the problem becomes more challenging when considering its distributed equivalent [65]. Efficient discovery algorithms for the assignment problem were recently explored in [48]. For $k = 3$, the complexity of the GPM problem remains an open issue.

3) SUMMARY

The (k, Δ)-GPM problem has two metrics of interest: *quality of the output matching M* (weight of M compared with the optimal matching M_{opt}) and *number of weight calculations* that were required to compute M. Regardless of the number of computed weights, we show in Proposition 1 that calculating the optimal matching M_{opt} is NP-hard for $k \geq 4$ whereas it is solvable in cubic time for $k = 2$.

C. REDUCTION AND APPROXIMATION

1) REDUCTION TO A ONE-TO-MANY ASSIGNMENT PROBLEM

Let's assume here that the weight of any group $G = p \cup \{c_1, \ldots, c_\ell\}$ with $\ell \leq k - 1$, $p \in P$ and $c_i \in C$, can be calculated as the sum of the individual *pairwise weights*, i.e., $w(G) = \sum_{1 \leq i \leq \ell} w(\{p, c_i\})$. In this situation, finding the best group (in terms of weight) of size k containing p is then equivalent of picking the $k - 1$ best partners for p. Hence, the maximum weight matching can be reduced to a one-to-many assignment problem, that matches members of the set P with at most $k - 1$ members of the set C such that the sum of the individual pairwise weights, i.e., the "edge weights" $w(\{p, c\})$, is maximum. This problem can be further reduced to the classical and well known one-to-one assignment problem in the following manner: for each $p \in P$, make $k - 1$ copies $p^1, \ldots, p^{k-1}$ of node p, while keeping

[2] A torus topology means that each node has on average the same number of neighbors regardless of its position on the map and is very similar to the usual map when $L \gg \Delta$.

the original weights, i.e., $\forall c \in C, w(\{p^j, c\}) = w(\{p, c\})$. Finally solve the one-to-one assignment problem (maximum matching in bipartite graphs) with the input $P' = \{p^j \mid j \in [1..k-1], p \in P\}$ and C. The hypergraph matching is then obtained by aggregating all selected edges sharing the same vertex copy. As aforementioned, the assignment problem can be solved exactly using the Hungarian algorithm in time $\mathcal{O}(|P'|^2 \cdot |C|) = \mathcal{O}(kn^2 \cdot \max\{\bar{\delta_\Delta}, \log n\}) = \mathcal{O}(kn^3)$, which is already prohibitive for large systems, see e.g. [66]. We refer hereafter to as *Optimal Pairwise* the hypergraph matching calculated based on the optimal matching using pairwise weights.

We note that setting the original weights (in the hypergraph) to the cost saving does not follow this paradigm and even though finding the maximum matching of the pairs do provide a hypergraph matching, it is not guaranteed any longer to be maximal in terms of the sum of the weights of the hyperedges. Such an example is given in Figure 2(f) where $M = \{\{p_1, c_1, c_2\}, \{p_3, c_3\}\}$ maximizes $\sum_{(p,c)\in P\times C \mid \exists e\in M, \{p,c\}\in e} w(\{p, c\}) = 360$ but $w(M) = 330$ and thus M is not maximum, cf. the matching of Figure 2(d) having a weight of 350.

2) APPROXIMATION FOR THE GPM PROBLEM

As shown in § IV-B, in general the GPM problem is intractable, however, we show thereafter that if the weights of the hyperedges can be approximated by using the sum of the pairwise weights, then it is possible to obtain a (polynomial-time) solution to the GPM within a bounded-approximation of the optimal.

Proposition 2: Assume that for every $p \in P$ and $X \subseteq C$ with $1 \leq |X| \leq k-1$ so that $\forall c \in X, \{p, c\} \in E$, we have

$$\alpha_1(k) \cdot \sum_{c\in X} w(\{p, c\}) \leq w(\{p\} \cup X) \leq \alpha_2(k) \cdot \sum_{c\in X} w(\{p, c\}),$$

and let A be an algorithm for the one-to-many assignment problem with approximation ratio r, then A provides an $r \cdot \alpha_2(k)/\alpha_1(k)$-approximation for the $(k, \Delta)-$GPM problem.

Proof: Let's suppose we use the construction explained in § IV-C1 to build a one-to-many assignment problem, i.e., a bipartite graph $G_\mathrm{p} = (P_k \cup C, E_\Delta)$ with $P_k = \{p^j \mid p \in P, 1 \leq j \leq k-1\}$ from a given $(k, \Delta)-$GPM instance over prosumer set P and consumer set C.

Let $\mathcal{M}_{opt}$ be an optimal solution for the (k, Δ)-GPM problem and M_{opt} be an optimal solution for the one-to-one assignment problem in G_p. Let $\mathcal{M}_A$ be the hypergraph matching obtained by merging together all pairs sharing a copy of a vertex p in the matching $M = A(G_\mathrm{p})$ obtained by executing A over the input G_p, the graph having been constructed in a way so that $\mathcal{M}_A$ is an answer to the (k, Δ)-GPM problem. Reversely, let M_p be the matching of the pairs obtained by breaking the hyperedges of $\mathcal{M}_{opt}$, thus forming edges from G_p.

Because algorithm A has approximation ratio r, we have

$$w(M) \geq \frac{w(M_{opt})}{r}.$$

Moreover, by the proposition's initial assumption, we have (using the lower-bound side)

$$\alpha_1(k) \cdot w(M) \leq w(\mathcal{M}_A)$$

and (using the upper-bound side)

$$w(\mathcal{M}_{opt}) \leq \alpha_2(k) \cdot w(M_p).$$

Since $w(\mathcal{M}_{opt}) \geq w(\mathcal{M}_A)$ and $w(M_{opt}) \geq w(M_p)$ by their optimality property, we obtain

$$\begin{aligned} w(\mathcal{M}_A) &\geq \alpha_1(k) \cdot w(M) \\ &\geq \alpha_1(k) \cdot \frac{w(M_{opt})}{r} \\ &\geq \alpha_1(k) \cdot \frac{w(M_p)}{r} \\ &\geq \frac{\alpha_1(k)}{\alpha_2(k) \cdot r} \cdot w(\mathcal{M}_{opt}). \end{aligned}$$

Hence, A provides an $r \cdot \alpha_2(k)/\alpha_1(k)$-approximation for the (k, Δ)-GPM problem. □

D. PAIRWISE WEIGHTS AS APPROXIMATIVE WEIGHTS

Following the construction presented in § IV-C1, one can use algorithms to solve the one-to-one assignment problem to build a hypergraph matching and hence a partition of consumers and prosumers into independent energy communities. However, the partition will not lead in this case to the global maximum in terms of weight as shown in the example of Figure 2(f). Nevertheless, one can bound how far from the optimal hypergraph matching the constructed matching is by building on the following proposition (cf. Proposition 1). In the following, the hyperedge weights correspond to the ones calculated based on cost savings following Definition 1, i.e., $w_t(G) = \mathsf{gain}(G, [t-\tau, t])$ for a certain τ (e.g. one month); t is omitted in the following. Neighborhoods are defined in Definition 2.

Proposition 3: For every prosumer $p \in P$ and group $X \subseteq C$ made only of consumers, with $1 \leq |X| \leq k-1$, and so that all consumers of X are located within distance Δ from p, i.e., $\forall c \in X, c \in N_\Delta(p)$, we have

$$\frac{1}{|X|} \sum_{c\in X} w(\{p, c\}) \leq w(\{p\} \cup X) \leq \sum_{c\in X} w(\{p, c\})$$

where w corresponds to cost-saving weights according to Definition 1, i.e., the financial gain of using P2P energy sharing for the community $G = \{p\} \cup X$.

Proof: For ease of notation, let $w_X = \sum_{c\in X} w(\{p, c\})$. *Lower bound:* In any community $G = \{p\} \cup X$, it is always possible to ignore some of the consumers upon optimizing for the community's cost (i.e., adding more consumers to a community can only produce more benefits and not less). Hence, we have $w(G) \geq w(\{p, c\})$ for any $c \in X$, that is $w(G) \geq \max_{c\in X} w(\{p, c\})$. Since $w_X \leq |X| \cdot \max_{c\in X} w(\{p, c\})$, we get $w(G) \geq w_X/|X|$.

Upper bound: The intuition behind the upper bound comes from the infeasibility to produce more cost saving as a group

than the sum of each consumer individually working with the same prosumer p. If it were possible, one would be able to re-create an individual strategy for one of the consumers c that would then beat the optimal individual strategy that produces a gain of $w(\{p, c\})$. Essentially, each gain obtained by the community as a whole is either due to an efficient usage of p's battery (irrelevantly of the consumers associated with p) or to a price difference between the price paid by using p's local energy (either from the same hour or from p's battery) and the current centralized grid's price.

Suppose one has an allocation A of p's resources over time in order to maximize the cost saving $w(G)$ for the community G, and let's note $w^A(G)$ the cost saving obtained following such allocation, i.e., $w^A(G) \geq w^{A'}(G)$ for any other allocation A' of p's resources. A provides, for each timestep, p's decisions upon the quantity of energy to transfer into/from its battery system, and by consequence how much energy is bought/sold from/to the grid by the group while covering X's aggregated consumptions. Hence, for any $c \in X$, one can deduce from A an "individual strategy" A_c following A and counter-acting the energy consumption ℓ of $X \setminus \{c\}$ by selling ℓ kWh to the rest of the grid. Intuitively, the cost saving $w^{A_c}(\{p, c\})$ associated with the A_c allocation cannot beat the best individual allocation for c cooperating solely with p as a community of size 2, that is $w^{A_c}(\{p, c\}) \leq w(\{p, c\})$. We formally prove the last statement hereafter.

Let's have a more focused look first at how the cost savings are calculated (i.e., base cost minus community cost, over the time period $\mathcal{T}$):

$$w^A(G) = \sum_{x \in G} \mathsf{bill}(x, \mathcal{T}) - \mathsf{bill}_A(G, \mathcal{T}), \tag{2}$$

with $\mathsf{bill}_A(G, \mathcal{T})$ being the cost over $\mathcal{T}$ under A. Following the strategy A, denote $\boldsymbol{bat}_A(p, t)$ the battery of p at time t, $\boldsymbol{el_{in}}^A(t)$ the energy bought from the grid at time t and $\boldsymbol{el_{out}}^A(t)$ the energy sold during the same hour. Since we assume A is a valid strategy, we have that $\mathsf{el}_{cons}(h, t)$ for $h \in G$ is balanced by electricity coming from either p's battery, the grid or p's local production. Hence, noting $\boldsymbol{bat}_A(t) = \boldsymbol{bat}_A(p, t) - \boldsymbol{bat}_A(p, t-1)$ and $\boldsymbol{grid}_A(t) = \boldsymbol{el_{in}}^A(t) - \boldsymbol{el_{out}}^A(t)$, we have:

$$\sum_{h \in G} \mathsf{el}_{cons}(h, t) = \boldsymbol{bat}_A(t) + \mathsf{el}_{gen}(p, t) + \boldsymbol{grid}_A(t).$$

1) INDIVIDUAL ALLOCATION OF THE COMMUNITY'S BENEFITS

We can now make the consumer allocated energy strategies under A explicit as follows. Order arbitrarily first the members of $G = \{p, c_1, \ldots, c_{k-1}\}$ and process the c_i's in the same order (hereafter consider "$p = c_0$"):

1) Use in priority the energy from $\mathsf{el}_{gen}(p, t)$;
2) If $\mathsf{el}_{cons}(c_i, t)$ is not yet covered by 1 and if there is still energy in the battery pool, i.e. $\boldsymbol{bat}_A(t) > 0$, then use it;
3) If $\mathsf{el}_{cons}(c_i, t)$ is not yet covered by 1 & 2, then use energy from $\boldsymbol{grid}_A(t)$.

With such explicit allocations, we can compute "individual cost savings", that is reusing the w-notation we set $w^A(c_i) = \sum_{t \in \mathcal{T}} w_t^A(c_i)$ is the cost saving that we can attribute to c_i and calculated as follows. We set $\boldsymbol{el_{out}}^A(c_i, t) = 0$ if $i > 0$, $\boldsymbol{el_{out}}^A(c_0, t) = \boldsymbol{el_{out}}^A(t)$ and $\boldsymbol{el_{in}}^A(c_i, t) = \boldsymbol{grid}_A(c_i, t)$ where $\boldsymbol{grid}_A(c_i, t)$ is the amount of energy drawn from $\boldsymbol{grid}_A(t)$ by the above allocation (potentially 0 if c_i only needed solar panel and/or battery energy during that timestep). Individual cost saving is then obtained as:

$$w_t^A(c_i) = \mathsf{cost}(c_i, t) - \mathsf{cost}_A(c_i, t)$$

where $\mathsf{cost}(c_i, t)$ is the usual cost (no community, cf. § III-B) and $\mathsf{cost}_A(c_i, t)$ is c_i's cost using $\boldsymbol{el_{in}}^A(c_i, t)$ and $\boldsymbol{el_{out}}^A(c_i, t)$ in the cost calculation (eq. 1). Now, let's observe that we have from developing equation 2:

$$\begin{aligned} w^A(G) &= \left(\sum_{h \in G} \mathsf{bill}(h, \mathcal{T})\right) - \mathsf{bill}_A(G, \mathcal{T}) \\ &= \sum_{0 \leq i \leq k-1} w^A(c_i). \end{aligned}$$

This holds because

$$\mathsf{bill}_A(G, \mathcal{T}) = \sum_{h \in G} \mathsf{cost}_A(h) = \sum_{h \in G} \sum_{t \in \mathcal{T}} \mathsf{cost}_A(h, t),$$

and irreverently on how the energy is locally allocated in each timestep, aggregated together, they account for all G's cost.

Now, let's show that $w^A(p)$ is negative, that is p's contribution under our formulation is decreasing the total cost saving $w^A(G)$. In our allocation of resources, p has priority on its battery and PV panels regardless if it is part of a community or alone. However, if p were left alone, some $\mathsf{el}_{gen}(p, t)$ or $\boldsymbol{bat}_A(t)$ could have been sold to the grid or stored instead of being used by the other peers forming the community G, which could eventually only decrease p's cost (and cannot increase it by any mean).

Now, recall $w(\{p, c\})$ is the optimal cost saving for the pair $\{p, c\}$. Consider the strategy A_c for the pair $\{p, c\}$ that follows A except it sells to the grid the energy that was originally intended to cover the other consumers $h \in G \setminus \{p, c\}$. Since A_c is not optimal in regards to the pair $\{p, c\}$, $w^{A_c}(\{p, c\}) \leq w(\{p, c\})$. Obviously thanks to the additional benefits coming from selling extra electricity, the cost under A_c is strictly less than $\mathsf{cost}_A(h)$ entailing $w^A(c) \leq w^{A_c}(\{p, c\})$. However, A_c being a valid strategy for the pair $\{p, c\}$, one also gets that $w^{A_c}(\{p, c\}) \leq w(\{p, c\})$ which entails $w^A(c) \leq w(\{p, c\})$.

We can now conclude our proof by putting together the last two claims:

$$\begin{aligned} w^A(G) &= \sum_{0 \leq i \leq k-1} w^A(c_i) = w^A(p) + \sum_{1 \leq i \leq k-1} w^A(c_i) \\ &\leq \sum_{1 \leq i \leq k-1} w(\{p, c_i\}). \end{aligned}$$

□

In our evaluation, we compare the bounds shown in Proposition 3 with those calculated in practical instances in Figure 4 and conclude that the theoretical bounds shown here are close to those observed in practice. At last, let

us derive from the previous proposition an approximation bound for the GPM problem by using the optimal pairwise weights as approximation for the hyperedge weights. Using in Proposition 2 the optimal assignment over $G_{\mathrm{p}} = (P_k \cup C, E_\Delta)$ obtainable in $\mathcal{O}(kn^3)$ with the Hungarian algorithm, and considering the bounds provided by Proposition 3, i.e., $\alpha_1(k) = 1/(k-1)$ and $\alpha_2(k) = 1$, we obtain the following result.

Corollary 1: For cost-saving weights, it is possible to compute a $(k-1)$-approximate hypergraph matching solution to the $(k, \Delta)-$GPM in $\mathcal{O}(kn^3)$ time.

V. EFFICIENT PEER MATCHING ALGORITHMS

We describe in this section algorithms that produce a solution (i.e., a hypergraph matching) to the GPM problem with the input of the matching problem being a bipartite graph based on end-users location (as defined in § IV-A). In order to balance the trade-off quality of the solution versus computational load in different ways, we present in § V-B three different algorithms: *Round Robin*, *Single Pass* and *Classic Greedy*, then explain in § V-D how to instantiate them using different weight functions.

We analyze here the time-complexity of the introduced matching procedures; we note the weight function is constant in this analysis. Dealing with the latter, for each algorithm and weight function, we provide an asymptotic analysis of the number of weight computations in order to produce the solution in § V-D. Recall that following § III-E, in order to reduce the search space, we have restrained hyperedges to contain exactly one member of the set P (a prosumer in our context). It is worth noting that all the algorithms can be easily adapted if this constraint is lifted by including prosumer-prosumer pairs in E_Δ and changing the line where the possible neighbors are calculated.

A. DEFINITIONS

Let $P \cup C$ be the input of a GPM problem of parameter (k, Δ). We assume that $\mathsf{weight}(p_i, c_j, M_i, t)$ returns a weight associated to node $p_i \in P$ and node $c_j \in C$ and possibly using a partially computed set of nodes $M_i \subset C$ that are already associated with p_i, whereas t indicates the current time-step (recall, weights are time-dependent). This function can be resolved by either:

1) executing a computationally-expensive procedure that relies on solving one or several LP-programs as defined in § III-B and III-C;
2) simply performing a look-up of a previously computed weight.

When a weight is obtained in the first case above, we say it is *computed*; such computation is the bottleneck of the matching procedures and is highlighted in the pseudocodes of the algorithms. The input of the matching algorithms is the bipartite graph $H_\Delta = (P \cup C, E_\Delta)$ where E_Δ captures all neighborhoods at geographic distance Δ from each prosumer, i.e., $E_\Delta = \{(i,j) \in P \times C \mid \mathsf{dist}(p_i, c_j) \leq \Delta\}$; an example of E_Δ restraining possible hyperedges is illustrated in Figure 2(a)-(b). We highlight in the following remark that the presented *Peer Matching* algorithms are more general than the GPM problem.

Algorithm 1 Round Robin Matching Procedure

Input : A bipartite graph $H_\Delta = (P \cup C, E_\Delta)$ and $k \geq 2$
Output: M, a k-bounded bipartite hypergraph matching;

```
// Initialization
foreach i ∈ P do
    M_i ← ∅ ;
foreach j ∈ C do
    S_j ← False ;
Ψ ← order(P) ;
while Ψ ≠ ∅ do
    foreach i ∈ Ψ do
        N ← {j ∈ C | {i,j} ∈ E_Δ ∧ ¬S_j} ;
        if N = ∅ ∨ |M_i| = k − 1 then
            Ψ ← Ψ \ {i} ;
        else
            if |N| > 1 then
                foreach j ∈ N do
                    b_j ← weight_t(p_i, c_j, M_i);
                ℓ ← arg max_{j∈N} b_j ;
            else
                ℓ ← N[1]
            S_ℓ ← True ;
            M_i ← M_i ∪ {ℓ};
return {M_i | i ∈ P};
```

Remark 1: Note that Algorithms 1, 2 and 3 do not rely on the geographical positions of the nodes but on the authorised matching pairs captured by E_Δ, hence any bipartite graph can be given as input and the algorithms can be applied to a wider range of problems.

The function $\mathsf{order}(P)$ sorts the set P according to a predefined ordering, to be provided by the user (cf § VI-A for the ones used in our evaluation). As an example, the matchings obtained by the three algorithms are displayed in Figure 2(d)-(f) when using the pairwise weights of Figure 2(c) with processing order p_1, p_2, p_3.

B. ALGORITHMS FOR EXPLORING THE DESIGN SPECTRUM

1) ROUND ROBIN MATCHING

Algorithm 1 builds a bipartite hypergraph matching by affecting to each prosumer one consumer at a time in a round-robin fashion. If no consumer can be affected to p (either because p has already $k-1$ consumers affected or no unmatched consumer is found within distance Δ of p), p is skipped (and discarded from subsequent matching attempts). Hence, all prosumers get at most 1 consumer each before a second iteration starts in affecting a second consumer to every prosumer. When a prosumer has a choice of which consumer

Algorithm 2 Single-Pass Matching Procedure

Input/Output: as in Algorithm 1.

```
// Same as lines 1-5 in Algorithm 1
foreach i ∈ P do
    N ← {j ∈ C | {i,j} ∈ E_Δ ∧ ¬S_j} ;
    if |N| ≥ k then
        foreach j ∈ N do
            b_j ← weight_t(p_i, c_j, M_i);
        while |M_i| < k − 1 do
            ℓ ← arg max_{j∈N} b_j ;
            M_i ← M_i ∪ {ℓ};
    else
        M_i ← N;
    foreach j ∈ M_i do
        S_j ← True ;
return {M_i | i ∈ P};
```

Algorithm 3 Classic Greedy Matching Procedure

Input/Output: as in Algorithm 1.

```
// Same as lines 1-5 in Algorithm 1
foreach {i,j} ∈ E_Δ do
    b_{i,j} ← weight_t(p_i, c_j, M_i);
// Sort all possible matching pairs
   by decreasing weights
B ← decreasing_sort({b_{i,j} | {i,j} ∈ E_Δ}) ;
foreach b_j ∈ B do
    if |M_i| < k − 1 ∧ ¬S_j then
        S_j ← True ;
        M_i ← M_i ∪ {j};
return {M_i | i ∈ P};
```

to pick, it greedily selects the consumer with highest weight; how the weights are settled is further described in § V-D.

2) SINGLE PASS MATCHING

Contrary to the previous one, Algorithm 2 builds a matching in a single pass over the prosumers. For each prosumer p, the $k-1$ best available consumers (in terms of weight) are matched with p.

3) CLASSIC GREEDY MATCHING

Algorithm 3 is the "classic" greedy procedure for solving the assignment problem (cf. § IV-C1) that, based on pre-computing all pairwise weights, sorts the prosumer-consumer pairs $(p, c) \in E_\Delta$ from highest to lowest, then associates consumers to prosumer whenever possible (c not already matched and p having less than $k-1$ affected consumers). As it is well-known that the greedy algorithm provides a 2-approximation to the assignment problem [48], Propositions 3 and 2 entail an approximation bound of $2 \cdot (k-1)$ on the quality of the output of the algorithm.

Corollary 2: Algorithm 3 provides a $2 \cdot (k-1)$*-approximation for the* $(k, \Delta)-$*GPM problem.*

C. ALGORITHMIC TIME AND SPACE COMPLEXITY

We analyze here the time complexity of the three introduced matching algorithms, assuming a constant time for each weight calculation.

Proposition 4: Algorithms 1, 2 and 3 run in respectively $\mathcal{O}(mk)$, $\mathcal{O}(m)$ *and* $\mathcal{O}(m \log m)$ *time, and use* $\mathcal{O}(|E_\Delta|)$ *memory.*

Proof: Algo. 1: the main while-loop is executed at most k times (after what all prosumers have been removed by line 10) and each inner for-loop goes through $|P|$ prosumers, each time calculating the best local choice using $\mathcal{O}(|N_\Delta(p)|)$ for prosumer p. The inner for-loop thus takes total time $\mathcal{O}(|P| + |E_\Delta|) = \mathcal{O}(\bar{\delta_\Delta}|P|)$.

Algo. 2: line 5 is executed $\mathcal{O}(|E_\Delta|)$ times whereas lines 6-8 also take $\mathcal{O}(m)$ in total when using a selection and partition algorithm to find the $k-1$ highest unsorted weights in each neighborhood.

Algo. 3: this is the time needed to sort all weights, then going through the sorted list takes $\mathcal{O}\left(|P|\bar{\delta_\Delta}\right)$ time.

Memory: *Algo. 1* and *Algo. 2* only require to store one bit per node, the constructed matching and momentary one node's full neighborhood, hence only $\mathcal{O}(|V|)$ space in addition to its input. *Algo. 3* requires storing and sorting all the input edges. □

D. INSTANTIATION OF THE WEIGHT FUNCTION

We define here two ways to calculate the weights: either in a "memoryful" fashion, taking into account previous choices made by the matching algorithm, or in a "memoryless" one with constant weights only depending on the involved prosumer and consumer pair (p_i, c_j) being examined. For each variant, we propose two ways to calculate the weight for a given pair, either based on the energy cost produced by the pair working as a community in the recent past $[t-\tau, t]$ (running a single LP-solver as described in § III-B and III-C) or based on the cost-saving produced by the pair over the same period $[t-\tau, t]$. In the former case, the goal of the GPM problem being to minimize the total cost, we inverse the weights to keep a maximum-weight problem. In all, four weight functions are defined as $\mathsf{WX}_t(i, j)$ to instantiate the call to $\mathsf{weight}_t(p_i, c_j, M_i)$.

1) MEMORYLESS WEIGHTS

Using memoryless weights (WA and WB) means that the weights are constant and independent of the matching procedure being run. This also means that building the matching is nothing more than solving the classic assignment problem (as described in § IV-C1). An example of memoryless weights is given in Figure 2(c) where a weight is given for each pair $(p, c) \in P \times C$. We set the memoryless weights as follows:

$\mathsf{WA}_t(i, j) = -\mathsf{bill}(\{p_i, c_j\}, [t-\tau, t])$;

$\mathsf{WB}_t(i, j) = \sum_{x \in \{p_i, c_j\}} \mathsf{bill}(\{x\}, [t-\tau, t])) + \mathsf{WA}_t(i, j)$.

Proposition 5: Algorithms 1, 2 and 3 need to compute $\mathcal{O}(m)$ *memoryless weights to solve the* $(k, \Delta)-$*GPM problem.*

Proof: Note Algorithms 2 and 3 only call the weight routine once per prosumer-consumer pair, hence $\mathcal{O}(|E_\Delta|)$ weights are ever calculated. By remembering previously computed weights, Algorithm 1 needs also to compute each weight only once since the current matching is not involved when using memoryless weights. □

Regarding the minimum number of weights to compute, the three algorithms differ. Algorithm 3 needs to sort all the weights, so $|E_\Delta|$ weights are always computed regardless how the nodes are distributed in space. However, for Algorithm 1 and Algorithm 2, there exist cases where none of $\Omega(|P|^2)$ weights are computed.[3] It is naturally possible to pre-compute all the $\mathcal{O}(n^2)$ memoryless weights based on some E_Δ. However, we note that the necessary number of weights to compute may be lower when using Algorithm 1 and Algorithm 2.

2) MEMORYFUL WEIGHTS

We define two memoryful weight functions (WC and WD) to take into account the already matched pairs, hence trying to provide the most accurate answer to the bipartite hypergraph matching problem. The weights are set as follows:

$\mathsf{WC}_t(i,j) = -\mathsf{bill}(M_i \cup \{p_i, c_j\}, [t-\tau, t])$;

$\mathsf{WD}_t(i,j) = \sum_{x \in M_i \cup \{p_i,c_j\}} \mathsf{bill}(\{x\}, [t-\tau, t])) + \mathsf{WC}_t(i,j)$.

Proposition 6: Algorithms 2 and 3 need to compute $\mathcal{O}(m)$ weights when using memoryful weights for solving the GPM problem of parameter (k, Δ); Algorithm 1 needs to compute $\mathcal{O}(km)$ weights.

Proof: Variants C and D now depend on previous matching. As with memoryless weights, Algorithms 2 and 3 only call the weight routine at most once per edge in E_Δ, hence $\mathcal{O}(|E_\Delta|)$ weights are ever used. Note that for those algorithms, memoryful weights are identical to memoryless ones. For Algorithm 1, the number of weight calculation is bounded by its time complexity, cf. Proposition 4. □

For Algorithm 1, pre-computation is not needed as each time the weight function is called, a weight calculation must occur (i.e., M_i is different for each iteration).

3) SAMPLING FOR REDUCING THE SEARCH SPACE

One solution to further reduce the number of weights to compute is to sample the neighborhoods, before executing the matching procedure. We propose here two sampling methods for a given sampling size s. For neighborhoods containing less than s consumers, we keep them all, otherwise for neighborhood N:

- **Random**: pick uniformly at random s neighbors in N;
- **Greedy**: pick the s consumers with largest consumptions in N (expected to produce higher cost savings than the others).

Applying such a sampling step as a pre-processing of the neighborhoods reduces the number of weights to compute to only $\mathcal{O}(ns)$.

[3] One can indeed construct the following example (for Algorithm 1): p_i is positioned at $(0, i \cdot \varepsilon)$ and c_j is positioned at $(0, \Delta + j \cdot \varepsilon')$ with $0 < \varepsilon < \frac{\Delta}{|P|}$ and $\varepsilon < \varepsilon' < \frac{\varepsilon(1+|C|)}{|C|}$.

TABLE 3. Summary of our proposed approaches to solve the GPM problem (all relying on pairwise weights as described in § IV-D) compared with a traditional solution based on local search booted by calculating a greedy hypergraph matching as an initialization step.

Proposed Algorithm	Space Bound[#]	Time Bound[#]	Weight Function	No. of Weights[*]	Approx. Ratio[◇]
Greedy Hypergraph	$\mathcal{O}(n^k)$	$\mathcal{O}(kn^k \log(n))$	$\mathbf{weight}_t$	$\mathcal{O}(n^k)$	k
Round Robin	$\mathcal{O}(m)$	$\mathcal{O}(km)$	Memoryful Memoryless	$\mathcal{O}(km)$ $\mathcal{O}(m)$	None
Single Pass	$\mathcal{O}(m)$	$\mathcal{O}(m)$	Memoryless	$\mathcal{O}(m)$	None
Classic Greedy	$\mathcal{O}(m)$	$\mathcal{O}(m \log(m))$	Memoryless	$\mathcal{O}(m)$	$2(k-1)$
Optimal Pairwise	$\mathcal{O}(n^2)$	$\mathcal{O}(kn^3)$	Memoryless	$\mathcal{O}(n^2)$	$k-1$

[#] Assuming for simplicity that $|C| = \mathcal{O}(n)$. Also, recall $m = |E_\Delta|$.
[*] Sampling weights (cf. V-D3) can be applied to all our approaches and reduce weight calculation to $\mathcal{O}(ns)$ for a sampling size of s.
[◇] Using cost-saving weights for the algorithms using pairwise weights.

E. SUMMARY OF OUR ALGORITHMIC APPROACHES

The complexity bounds, approximation ratio and weight functions of all our introduced approaches are summarized in Table 3, along the classic greedy algorithm over the hyperedges as baseline reference. Each algorithm presents interesting perspectives and trade-offs: Optimal Pairwise is likely to achieve a better solution, Classic Greedy reduces space/time complexity while guaranteeing a bounded approximation, Single Pass is the fastest algorithm time-wise, while Round Robin offers the possibility to consider the current state of the matching using memoryful weights.

One can observe that all the proposed algorithms compute significantly fewer weights than the number of hyperedges, that is bounded by $\mathcal{O}\left(|P| \cdot \binom{\delta}{k-1}\right) = \mathcal{O}(n\delta^{k-1})$, with δ being the maximum size of the neighborhoods, i.e., $\delta = \max_{p \in P} |N_\Delta(p)|$, or said otherwise whenever $|C| = \Theta(n)$, the number of hyperegdes can be in $\Theta(n^k)$.

VI. EXPERIMENTAL STUDY

We present in this section an experimental study comparing the performance of the proposed algorithms in terms of number of computed peer matching preferences (i.e., hyperedge weights) versus the amount of cost saving produced by the computed matching (i.e., the quality of the output matching) exploring in details communities of size at most 5 with a billing period of one year. We further investigate the aforementioned trade-off by analyzing the quality of the solution versus the maximum size of the communities, the computational and cost saving efficiency of pre-sampling the neighborhoods and the gains obtained with shorter billing periods. At last, we complement our results with experiments on scaled instances demonstrating the scalability of our proposed approach and further summarize all our results.

A. EXPERIMENTAL SET-UP

1) ENERGY DATA

Consumption profiles of 2221 real households are used in this study, originating from [67]. Each trace contains electricity

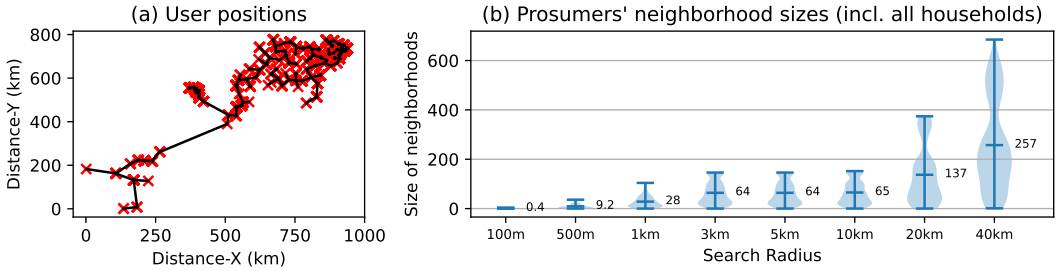

FIGURE 3. Dataset used in our evaluation: (a) Position of the users with an example of grid infrastructure relying them; (b) Size of the neighborhoods.

consumption measures on a hourly basis for a different household for one year.

For our experimental evaluation, following current trends and near-future realistic projections of energy resources [16], [68], [69], we have equipped households willing to participate in the energy-sharing system with energy resources as follows: 10% of them are prosumers with only PV panels, 10% are prosumers with both PV panels and a battery system, and the rest are usual consumers.

Capacities for the energy resources are set for each prosumer relative to its average electricity consumption following the current trends on installed PV capacities (based on the Open PV 19 dataset [68]). An hourly solar profile is used here assuming same roof panel orientations [70]. We use in our study the original city position of the households (obtained through using their respective zipcode), where we added a small random offset of up to 1km, see Fig. 3(a). The hourly electricity price profile is based on the output from a European scale dispatch model [71]. Taxes are set to $\mathsf{tax} = 25\%$, $\mathsf{el}_{tax} = 6.9$ € cents/kWh and $\mathsf{el}_{net} = 0.58$ € cents/kWh.

2) ALGORITHMS

We compare in this study the three procedures defined in § V-B: *Round Robin*, *Single Pass* and *Classic Greedy* algorithms, as well as the hypergraph matching computed based on the optimal pairwise matching as described in § IV-C1. For the first two, the order in which prosumers are processed can take three forms: *Increasing* or *Decreasing* order of their average energy consumptions, or decreasing order of their energy resources (*Resource*); the latter is obtained by adding together both PV capacity (kWp) and battery capacity (kWh). Single Pass and Classic Greedy algorithms can be tuned to use two variants of the weight function (as explained in § V-D): cost-based weights (WA) or saving-based weights (WB). In addition to the above two variants (referred as ''memoryless weights''), the Round Robin algorithm is tested with the two memoryful counterparts: cost-based memoryful weights (WC) or saving-based memoryful weights (WD). We hence tested the 16 different combinations of algorithms, prosumer orders and weight functions of Table 4 (cf. first three columns of Table 5). The

TABLE 4. Tested parametrized procedures for comparing our Peer Matching algorithms (cf. § VI-B).

Algorithm	Processing Order	Weight Function
Round Robin	Increasing	WA, WB, WC, WD
	Decreasing	WA, WB, WC, WD
	Resource	WA, WB
Single Pass	Decreasing	WA, WB
	Resource	WA, WB
Classic Greedy	–	WB
Optimal Pairwise	–	WB

experiments were run on a high-end server (Intel Xeon E5 2650 CPU, 64GB RAM) where computing a single weight takes about 2 seconds (corresponding to the execution of an LP-solver with 26280 input variables for a cost-optimization over one year of data).

3) SEARCH RADIUS

We set for the experiments 8 different search radii: 100m, 500m, 1km, 3km, 5km, 10km, 20km and 40km. For each search radius, Fig. 3(b) presents the distribution of the size of the nodes' neighborhoods (only based on the node positions). Since the search radii of 5km and 10km produce very similar neighborhood sizes as 3km, we omit them in the rest.

4) PAIRWISE VERSUS HYPEREDGE WEIGHTS

All studied memoryless algorithms manipulate only the pairwise weights to infer their matching decisions. Following Proposition 3, we know that when using cost-saving weights, we can upper- and lower-bound all hyperedge weights in relation to the sum of the pairwise weights. Fig. 4 explores what is the distribution of the ratio between the hyperedge weights (the ''real weights'') and the sum of pairwise weights (the ''approximate weights'' as presented in § IV-D), i.e., the parameter $w(\{p\} \cup X)/\sum_{c \in X} w(\{p, c\})$. The figure illustrates how close to the theoretical bounds of $\mathbf{1}/(\mathbf{k}-\mathbf{1})$ (lower bound) and $\mathbf{1}$ (upper bound) the actual values are, when considering a range of 1km (using about 80k weights for possible triplets, 10k pairwise weights and 1000 random samples for 4- and 5-weights).

One may observe that (i) the bounds obtained in Proposition 3 are very close to being tight for $k = 3$, (ii) the

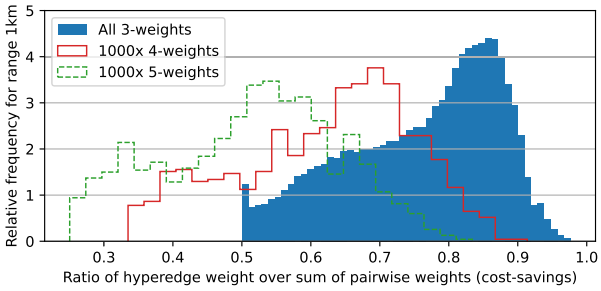

FIGURE 4. Comparison of hyperedge weights with the sum of pairwise weights for memoryless cost-saving weights, search radius of 1km, and $k = 3$ to 5; For $k = 3$, all weights are considered, while for $k = 4$ and $k = 5$, 1000 randomly chosen weights are used.

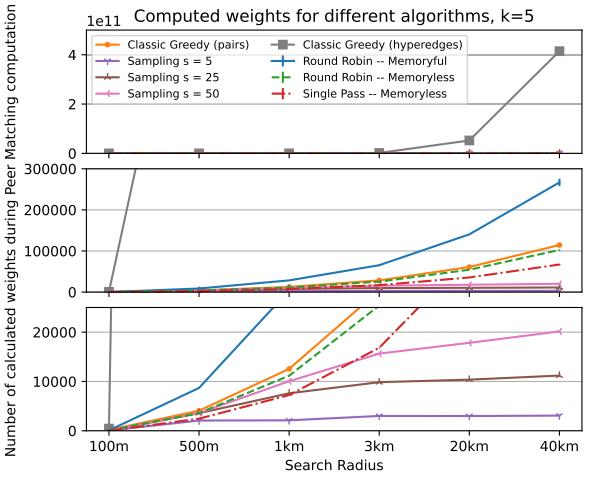

FIGURE 5. Average number of calculated weights for the different matching algorithms and search radii; $k = 5$.

lower-bound is quite close to the lowest found ratios and that, (iii) the ratios are in general between 0.5 and 0.9. This indicates that pairwise weights do provide a suitable approximation for the weights of larger groups.

B. COMPARISON OF PEER MATCHING ALGORITHMS

We shall hereafter compare the performance of all our algorithms based on our two performance measures: *quality of the matching* (in terms of cost saving achieved by the communities) and *number of weight calculations* (main computational bottleneck) for the 16 parametrized procedures of Table 4.

1) QUALITY OF THE MATCHING

The solution obtained using the studied algorithms is summarized in relative terms in Table 5. The results are calculated upon matching together 445 prosumers (half of them having also a battery system) with 1776 consumers into groups of size at most $k = 5$ containing exactly one prosumer each and running the LP-solver (cf. § III-B) over a year of data to obtain each community's yearly cost (used as the hyperedge weight). Recall the hyperedge weight is the basis of the `weight` function in Algorithms 1, 2 and 3, following the four different ways to calculate weights as presented in § V-D.

In our scenario, up to 150k € can be saved when all the nodes cooperate in a single community (equivalent of setting $\Delta = k = \infty$), from which up to 91.6% can be recovered by using $\Delta = 40$km and $k = 5$. In terms of quality of the solution, the Single Pass algorithm using decreasing order and cost-based weights is clearly outperformed by the rest, whereas the three best performing ones are the Optimal Pairwise, the Classic Greedy and the Round Robin algorithm using decreasing order and WD (all with cost-saving weights). Closely behind, we note the high performance of both the Round Robin (decreasing order, WB) and the Single Pass (WB) algorithms, both computing significantly less weights than the algorithms that require the full pairwise weight list (Optimal Pairwise and Classic Greedy).

2) NUMBER OF WEIGHT CALCULATIONS

Fig. 5 presents the number of pairwise or group weights[4] calculated along the computation of the matchings of Table 5. Recall a weight calculation corresponds to running a LP-solver (§ III-B) and may correspond to calculate the cost or cost-saving of a community of size 2 up to k (only for memoryfull weights of the Round Robin matching).

All algorithms (except the sampling ones) display a quadratic increase in number of weights as the search radius increases, with varying slopes. Confirming the theoretical upper bounds given in Section V, memoryful weights entail a large computational overhead. On the contrary, memoryless weights reduce the burden on calculating weights with the Single Pass algorithm only calculating about 57% of all prosumer-consumer pairs, and less than 25% of the memoryful weights. As expected, sampling s values among the neighborhoods further reduces the weight computations to a bit less than ns in total. One can observe that all tested algorithms compute several orders of magnitude fewer weights than the number of possible communities, at about 414 billion possible communities for $\Delta = 40$km, that would be required to compute in order to run a greedy algorithm on the input.

3) NEIGHBORHOOD SIZE

The neighborhood size providing the best trade-off depends on the search radius, as shown in Fig. 6(a) with the cost saving produced using different bounds on the community size (k).

In our evaluation scenario, using $k = 5$ provides a cost only 0.01% inferior than using $k = 10$ for the Round Robin algorithm. For the Single Pass algorithm, using neighborhood sizes above 5 is even detrimental as the first prosumers get

[4]Variability is negligible in our evaluation: the average value is displayed with error bars, if any, indicating min-max values accounting for the different tested weight functions and prosumer orders. Both Optimal Pairwise and Classic Greedy require the same number of weights, $|E_\Delta|$.

TABLE 5. Comparison of the different algorithms, processing orders and weight functions ⋆.

Matching Algorithm (§ V-A)	Processing Order (§ VI-A)	Weight Function (§ V-B)	100m	500m	1km	3km	20km	40km
Round Robin (Alg. 1)	Increasing	WA	9.5%	60.9%	74.2%	78.8%	81.6%	83.8%
		WB	9.3%	61.7%	73.7%	78.5%	80.3%	82.4%
		WC	9.5%	60.9%	74.3%	78.9%	81.6%	83.9%
		WD	9.3%	61.7%	74.1%	79.0%	81.0%	83.2%
	Decreasing	WA	9.5%	61.7%	74.8%	79.0%	82.2%	85.0%
		WB	**9.6%**	**64.4%**	77.9%	82.9%	85.0%	87.4%
		WC	9.5%	61.7%	74.7%	79.0%	82.1%	84.8%
		WD	**9.6%**	**64.4%**	78.1%	83.4%	85.6%	88.3%
	Resource	WA	**9.6%**	61.8%	75.4%	79.3%	82.4%	85.0%
		WB	**9.6%**	58.7%	77.9%	82.8%	84.8%	87.3%
Single Pass (Alg. 2)		WA	9.5%	58.7%	71.5%	73.4%	76.1%	79.2%
		WB	9.5%	61.1%	75.0%	79.7%	83.6%	86.9%
	Decreasing	WA	9.5%	58.7%	68.8%	71.1%	74.7%	77.6%
		WB	9.5%	61.1%	73.6%	78.7%	82.5%	86.0%
Classic Greedy (Alg. 3)		WB	**9.6%**	62.7%	77.5%	83.2%	86.7%	90.5%
Optimal Pairwise (cf. § III.C)		WB	**9.6%**	64.0%	**79.0%**	**83.7%**	**87.6%**	**91.6%**

⋆ Tested weight functions: cost-based memoryless (WA) or memoryful (WC) and saving-based memoryless (WB) or memoryful (WD). The presented percentages are the fraction of the gain obtained by the matchings when compared with a single unrealistic 2221-households community. Best results for each search radius are displayed in bold, and for indication only and to ease the reading of the table, performance (compared with the best procedure for a given radius) is indicated by a background with a darker shade of green.

matched with a large amount of peers which removes the possibility for later processed prosumers to get consumers at all. We note choosing k should follow the distribution of prosumers in the pool.

4) WEIGHT SAMPLING

We analyze the effect of weight sampling in Fig. 6(b). The two strategies to sample a fixed number of potential partners in each neighborhood are compared: either perform a random or a greedy selection. The result clearly favors a random selection which is very efficient even for small sampling size (here, $2k = 10$ to $4k = 20$ neighbors), whereas the greedy solution only catches up for large sampling size (both converges to the non-sampling algorithm when s reaches δ, the maximum size of the neighborhoods).

5) CONTINUOUS MATCHING

In previous experiments, we have used historical data for one year as input to the aggregate model (LP-optimization) to calculate the peers' matching preferences. This is based on the assumption that the peers participating in long-term energy communities will likely reproduce overall the same or similar consumption patterns over time (and entailing similar matching preferences in the future), cf. [9], [72].

The impact of performing the peer matching on smaller intervals than one year is studied in Fig. 6(c). The figure displays in relative term the cost saving achieved when performing the peer matching every week, second week, month, and second month, with all compared to the one year matching used so far (set to 1). The cost savings obtained with all smaller periods decrease compare to using a year of data, but only to a very small extend, all reaching at least 99% of the yearly saving. Even though shorter interval communities could potentially gain from seasonable changes in finding ideal trading partners among fellow peers, this experiment confirms that long-term communities are very robust in terms of cost-saving.

C. SCALING EVALUATION AND VALIDATION

We provide here an evaluation of our approach, methods, and introduced algorithms on larger problem instances with the largest instances containing more than 100k end-users and 180M edges (allowed prosumer-consumer pairs), simulating a possible future large-scale adoption of P2P energy sharing at e.g. the city- or region-level.

1) DATA OF LARGE INSTANCES

To build larger graph instances, we scale our original real-world dataset of 2221 end-users. Let **I1** be the household

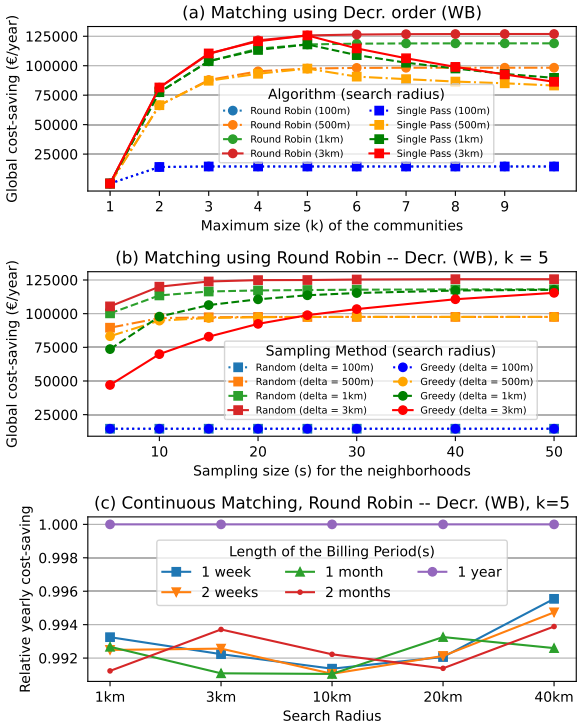

FIGURE 6. Sensitivity analysis showing the impact of (a) different maximum neighborhood sizes, (b) pre-sampling the neighborhoods and (c) shortening the billing period.

TABLE 6. Scaled instances with number of prosumers, consumers, edges, quantity of data (billing period) and average execution time of the LP-solver. NB: one edge's weight calculation requires at most 2 runs of the LP-solver.

I#	|P|	|C|	$\|E_\Delta\|$	Qty. of data	Time/pb (s)[b]
1	445	1776	66979	1 year	2.08± 0.172
2	890	3552	269156	6 months	1.08 ± 0.116
3	1335	5328	601965	4 months	0.76 ± 0.075
4	2670	10656	2412378	2 months	0.435 ± 0.056
5	5785	23088	11383965	1 month	0.266 ± 0.036
6	11570	46176	45657204	2 weeks	0.172 ± 0.028
7	23140	92352	182551323	1 week	0.125 ± 0.02

[b] Over 100-1000 sampled executions, ± standard deviation.

pool instance used in the experiments of § VI-B. Instances **I2** to **I7** are obtained by scaling up the dataset as follows (see Table 6 for the parameters of all scaled instances). We divide the year into smaller portions (from 1 week to 6 months) and the consumption data associated to each portion is used as stand-in for one user in the scaled instance. In this process, we keep the initial PV and battery capacities of the initial household, hence the proportion and distribution of resources among prosumer vs. consumer stays the same in the scaled instances.

The solar profile and price time series are identical for all users and correspond to a portion of the original data centered around the mid-year hour. Neighborhoods are randomly generated as follows: all end-users receive a random position within a square area of size 1×1 and we set $\Delta = \sqrt{\frac{1}{10\pi}}$ leading to neighborhoods containing around 10% of all users

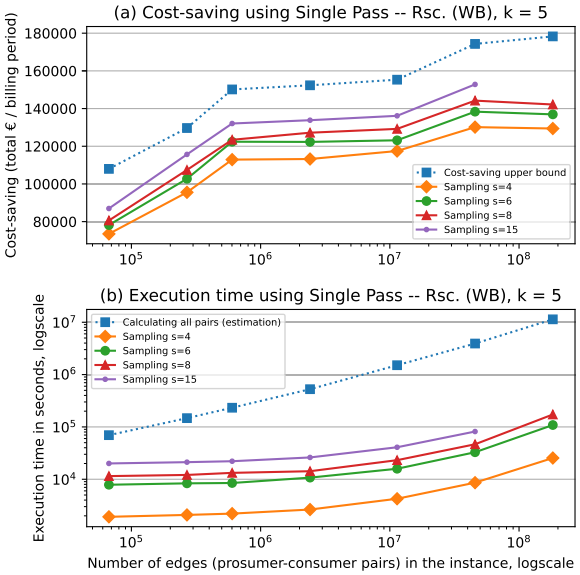

FIGURE 7. Experiments on scaled instances: cost efficiency (€ per billing period) and time efficiency (execution time in seconds) for different approaches as a function of the number of edges in the input instance.

(matching our settings used in § VI-B when Δ was set to 40km). It is important to note that no household consumption data is ever duplicated neither simulated in the generation of the scaled experiments.

Time measurements are obtained with the Python pyomo[5]-based implementation relying on the coin-or/Cbc[6] open-source LP solver on an openSUSE-Tumbleweed server equipped with an Intel Xeon CPU E5-2695 2.1GHz and 64 GB RAM.

2) EXPERIMENTS ON SCALED INSTANCES

We present the results of the experiments on large instances in Figure 7. For the cost saving, we use the single pass algorithm (Algorithm 2) with $k = 5$ and cost saving weights combined a random pre-sampling of the neighborhoods. The upper bound of cost savings in Figure 7 corresponds to the benefit obtained without using small groups but instead a single community that encompasses the entire pool of households. Using a small sampling size of 6 to 15, our solution already reaches up to 88-89% of the theoretical highest possible saving (with groups of arbitrary size, not only limited to size up to 5) while requiring about two orders of magnitude less computation time compared to calculating the cost of all prosumer-consumer pairs (for example, Algorithm 3 or using the optimal matching over the pairs).

Note that we only considered here algorithms based on our modeling of using pairwise weights to build the partition. When considering existing state-of-the-art algorithms for the hypergraph matching problem [37], [38], [39] with hyperedges of size 5, we estimate to 31 years of our CPU time

[5] http://www.pyomo.org/.
[6] https://github.com/coin-or/Cbc.

for the computation of all weights for the smallest instance **I1** (prerequisite for computing the initial greedy solution), and up to 10^{17} seconds for the largest instance. Our proposed solutions hence produce high quality solutions, likely within a couple of percentage points from the optimal for the set maximum size k for the groups, while being able to scale to very large instances.

D. SUMMARY OF THE RESULTS

In short, our experimental evaluation leads to the following observations on our dataset:

1) to maximize cost-efficiency, use Round Robin (decreasing order, WD) for a small search radius and Classic Greedy or Optimal Pairwise for larger search spaces;
2) memoryful weights do not provide a significant advantage in terms of cost-saving that justifies the induced overhead, with Round Robin (Decreasing or Resource order, WB) algorithm providing one of the best trade-offs overall in our experimental scenario;
3) applying a random pre-sampling of each neighborhood of e.g. $s = 20$ weights for $k = 5$ reduces drastically the computational overhead with little impact on the cost-efficiency, with further reductions obtained using the Single Pass algorithm. For a large search radius, the latter only requires a very small fraction of the weight computations of a brute-force approach enumerating all possible communities (about e.g. $2 \cdot 10^{-8}$ for $\Delta = 40$km).
4) our methods can scale to systems with up to a hundred thousand end-users and produce a cost-competitive and computationally efficient solution.

VII. CONCLUSION

This paper studies the peer matching problem to participate in P2P energy sharing. We introduce the Geographical Peer Matching problem within a well-defined mathematical framework setting the problem as a hypergraph matching problem with a bounded search radius Δ and output partitions of size up to k. This allows known approximation algorithms for the weighted k-set problem to be ported to our energy sharing setting. To provide an efficient solution to the problem, we introduce and analyze three different matching algorithms that do not require to compute all $\mathcal{O}(n^k)$ possible weights (each requiring the run of a computationally expensive LP-solver) but only at most $\mathcal{O}(kn^2)$. We also provide optimizations that, even though they do not change asymptotic behaviors of the proposed algorithms, are shown in this work to yield a practical computational advantage without sacrificing the quality of the produced solution. Pre-sampling the weights can further reduce the number of weight calculations to $\mathcal{O}(sn)$. For instance, this means that over the more than 10^{14} possible communities over our 2221 household pool, the cost savings of only 10,000 communities need to be calculated to reach a close to optimal partitioning. Our extensive experimental study shows that up to 91.6% of the benefit of a very large community (i.e., of size $k = n$ and unbounded geographical diameter $\Delta = \infty$) can be obtained by limiting communities to 5 nodes only using a small bounded geographical search radius. We demonstrate that the introduced algorithms are both scalable in terms of required computation, handling up to hundred of thousands of users, and efficient in terms of the quality of the computed solution, achieving performance close to an optimal solution. We further expect that as the introduced algorithms are more general than our specific problem, they can also be useful in other contexts where it is challenging to compute a solution to the hypergraph matching problem.

Some practical aspects for further research are (i) how to push parts of the matching computation towards the end-users for a more edge-friendly solution, for saving data transfers and caring about the privacy perspective [73], [74], while transitioning from batch-based to the online analysis required by today's smart metering infrastructure [75], [76], and (ii) how to update the matching dynamically and maintain a stable network through the arrivals and departures of peers [77], [78], possibly building on and adapting previous distributed and adaptive algorithms for matching with preferences [79], [80].

ACKNOWLEDGMENT

An earlier version of this paper was presented in part at the 37th ACM/SIGAPP Symposium on Applied Computing in [DOI: 10.1145/3477314.3507203].

REFERENCES

[1] R. Duvignau, V. Gulisano, and M. Papatriantafilou, "Efficient and scalable geographical peer matching for P2P energy sharing communities," in *Proc. 37th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2022, pp. 187–190.
[2] Y. Zhou and S. C.-K. Chau, "Multi-user coalition formation for peer-to-peer energy sharing," in *Proc. 11th ACM Int. Conf. Future Energy Syst.*, Jun. 2020, pp. 386–387.
[3] S. C.-K. Chau, K. Elbassioni, and Y. Zhou, "Approximately socially-optimal decentralized coalition formation with application to P2P energy sharing," *ACM SIGENERGY Energy Informat. Rev.*, vol. 2, no. 4, pp. 3–17, Dec. 2022.
[4] M. I. Azim, W. Tushar, and T. K. Saha, "Coalition graph game-based P2P energy trading with local voltage management," *IEEE Trans. Smart Grid*, vol. 12, no. 5, pp. 4389–4402, Sep. 2021.
[5] R. Duvignau, V. Heinisch, L. Göransson, V. Gulisano, and M. Papatriantafilou, "Benefits of small-size communities for continuous cost-optimization in peer-to-peer energy sharing," *Appl. Energy*, vol. 301, Nov. 2021, Art. no. 117402.
[6] W. Tushar, T. K. Saha, C. Yuen, M. I. Azim, T. Morstyn, H. V. Poor, D. Niyato, and R. Bean, "A coalition formation game framework for peer-to-peer energy trading," *Appl. Energy*, vol. 261, Mar. 2020, Art. no. 114436.
[7] R. Duvignau, V. Heinisch, L. Göransson, V. Gulisano, and M. Papatriantafilou, "Small-scale communities are sufficient for cost- and data-efficient peer-to-peer energy sharing," in *Proc. 11th ACM Int. Conf. Future Energy Syst.*, Jun. 2020, pp. 35–46.
[8] W. Tushar, T. K. Saha, C. Yuen, T. Morstyn, H. V. Poor, and R. Bean, "Grid influenced peer-to-peer energy trading," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1407–1418, Mar. 2020.
[9] V. Heinisch, M. Odenberger, L. Göransson, and F. Johnsson, "Organizing prosumers into electricity trading communities: Costs to attain electricity transfer limitations and self-sufficiency goals," *Int. J. Energy Res.*, vol. 43, no. 13, pp. 7021–7039, Jul. 2019.
[10] M. Khorasany, Y. Mishra, and G. Ledwich, "A decentralized bilateral energy trading system for peer-to-peer electricity markets," *IEEE Trans. Ind. Electron.*, vol. 67, no. 6, pp. 4646–4657, Jun. 2020.

[11] S. C.-K. Chau, J. Xu, W. Bow, and K. Elbassioni, "Peer-to-peer energy sharing: Effective cost-sharing mechanisms and social efficiency," in *Proc. 10th ACM Int. Conf. Future Energy Syst.*, Jun. 2019, pp. 215–225.

[12] A. Lüth, J. M. Zepter, P. Crespo del Granado, and R. Egging, "Local electricity market designs for peer-to-peer trading: The role of battery flexibility," *Appl. Energy*, vol. 229, pp. 1233–1243, Nov. 2018.

[13] C. Long, J. Wu, Y. Zhou, and N. Jenkins, "Peer-to-peer energy sharing through a two-stage aggregated battery control in a community microgrid," *Appl. Energy*, vol. 226, pp. 261–276, Sep. 2018.

[14] R. Schleicher-Tappeser, "How renewables will change electricity markets in the next five years," *Energy Policy*, vol. 48, pp. 64–75, Sep. 2012.

[15] C. Zhang, J. Wu, C. Long, and M. Cheng, "Review of existing peer-to-peer energy trading projects," *Energy Proc.*, vol. 105, pp. 2563–2568, May 2017.

[16] U. J. J. Hahnel, M. Herberz, A. Pena-Bello, D. Parra, and T. Brosch, "Becoming prosumer: Revealing trading preferences and decision-making strategies in peer-to-peer energy communities," *Energy Policy*, vol. 137, Feb. 2020, Art. no. 111098.

[17] W. Tushar, C. Yuen, T. K. Saha, T. Morstyn, A. C. Chapman, M. J. E. Alam, S. Hanif, and H. V. Poor, "Peer-to-peer energy systems for connected communities: A review of recent advances and emerging challenges," *Appl. Energy*, vol. 282, Jan. 2021, Art. no. 116131.

[18] W. Tushar, S. Nizami, M. I. Azim, C. Yuen, D. B. Smith, T. Saha, and H. V. Poor, "Peer-to-peer energy sharing: A comprehensive review," *Found. Trends Electric Energy Syst.*, vol. 6, no. 1, pp. 1–82, 2023.

[19] S. Henni, P. Staudt, and C. Weinhardt, "A sharing economy for residential communities with PV-coupled battery storage: Benefits, pricing and participant matching," *Appl. Energy*, vol. 301, Nov. 2021, Art. no. 117351.

[20] P. Huang, M. Han, X. Zhang, S. A. Hussain, R. J. Bhagat, and D. H. Kumar, "Characterization and optimization of energy sharing performances in energy-sharing communities in Sweden, Canada and Germany," *Appl. Energy*, vol. 326, Nov. 2022, Art. no. 120044.

[21] A. L. Bukar, M. F. Hamza, S. Ayub, A. K. Abobaker, B. Modu, S. Mohseni, A. C. Brent, C. Ogbonnaya, K. Mustapha, and H. O. Idakwo, "Peer-to-peer electricity trading: A systematic review on current developments and perspectives," *Renew. Energy Focus*, vol. 44, pp. 317–333, Mar. 2023.

[22] E. Barabino, D. Fioriti, E. Guerrazzi, I. Mariuzzo, D. Poli, M. Raugi, E. Razaei, E. Schito, and D. Thomopulos, "Energy communities: A review on trends, energy system modelling, business models, and optimisation objectives," *Sustain. Energy, Grids Netw.*, vol. 36, Dec. 2023, Art. no. 101187.

[23] F. V. Cerna, M. Pourakbari-Kasmaei, R. G. Barros, E. Naderi, M. Lehtonen, and J. Contreras, "Optimal operating scheme of neighborhood energy storage communities to improve power grid performance in smart cities," *Appl. Energy*, vol. 331, Feb. 2023, Art. no. 120411.

[24] V. Z. Gjorgievski, S. Cundeva, and G. E. Georghiou, "Social arrangements, technical designs and impacts of energy communities: A review," *Renew. Energy*, vol. 169, pp. 1138–1156, May 2021.

[25] E. A. Soto, L. B. Bosman, E. Wollega, and W. D. Leon-Salas, "Peer-to-peer energy trading: A review of the literature," *Appl. Energy*, vol. 283, Feb. 2021, Art. no. 116268.

[26] S. Aggarwal, N. Kumar, S. Tanwar, and M. Alazab, "A survey on energy trading in the smart grid: Taxonomy, research challenges and solutions," *IEEE Access*, vol. 9, pp. 116231–116253, 2021.

[27] M. I. Azim, W. Tushar, T. K. Saha, C. Yuen, and D. Smith, "Peer-to-peer kilowatt and negawatt trading: A review of challenges and recent advances in distribution networks," *Renew. Sustain. Energy Rev.*, vol. 169, Nov. 2022, Art. no. 112908.

[28] J. Liu, Q. Long, R.-P. Liu, W. Liu, X. Cui, and Y. Hou, "Privacy-preserving peer-to-peer energy trading via hybrid secure computations," *IEEE Trans. Smart Grid*, vol. 15, no. 2, pp. 1951–1964, Mar. 2024.

[29] A. Chiarini and L. Compagnucci, "Blockchain, data protection and P2P energy trading: A review on legal and economic challenges," *Sustainability*, vol. 14, no. 23, p. 16305, Dec. 2022.

[30] S. C.-K. Chau, N. Wang, and S. Karumba, "EnergiPay: Off-chain payment channel for blockchain-enabled peer-to-peer energy trading," in *Proc. 15th ACM Int. Conf. Future Sustain. Energy Syst.*, Jun. 2024, pp. 625–632.

[31] L. Wang, Q. Zhou, Z. Xiong, Z. Zhu, C. Jiang, R. Xu, and Z. Li, "Security constrained decentralized peer-to-peer transactive energy trading in distribution systems," *CSEE J. Power Energy Syst.*, vol. 8, no. 1, pp. 188–197, Jan. 2022.

[32] F. Manne, M. Naim, H. H. Lerring, and M. Halappanavar, "On stable marriages and greedy matchings," in *Proc. 7th SIAM Workshop Combinat. Scientific Comput.*, Jan. 2016, pp. 92–101.

[33] Y. Zhou and P. D. Lund, "Peer-to-peer energy sharing and trading of renewable energy in smart communities—Trading pricing models, decision-making and agent-based collaboration," *Renew. Energy*, vol. 207, pp. 177–193, May 2023.

[34] A. Khodabakhsh, J. Horn, E. Nikolova, and E. Pountourakis, "Prosumer pricing, incentives and fairness," in *Proc. 10th ACM Int. Conf. Future Energy Syst.*, Jun. 2019, pp. 116–120.

[35] E. Hazan, S. Safra, and O. Schwartz, "On the complexity of approximating k-set packing," *Comput. Complex.*, vol. 15, no. 1, pp. 20–39, May 2006.

[36] E. M. Arkin and R. Hassin, "On local search for weighted K-set packing," *Math. Oper. Res.*, vol. 23, no. 3, pp. 640–648, Aug. 1998.

[37] B. Chandra and M. M. Halldórsson, "Greedy local improvement and weighted set packing approximation," *J. Algorithms*, vol. 39, no. 2, pp. 223–240, May 2001.

[38] P. Berman, "A D/2 approximation for maximum weight independent set in D-claw free graphs," in *Proc. Scandin. Workshop Algorithm Theory*. Berlin, Germany: Springer, 2000, pp. 214–219.

[39] L. Zhang, H. Zhang, L. Yu, H. Xu, L. Song, and Z. Han, "Virtual resource allocation for mobile edge computing: A hypergraph matching approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[40] L. Wang, H. Wu, Y. Ding, W. Chen, and H. V. Poor, "Hypergraph-based wireless distributed storage optimization for cellular D2D underlays," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 10, pp. 2650–2666, Oct. 2016.

[41] P. Zhao, W. Guo, D. Xu, Z. Jiang, J. Chai, L. Sun, H. Li, and W. Han, "Hypergraph-based resource allocation for device-to-device underlay H-CRAN network," *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 8, Aug. 2020, Art. no. 155014772095133.

[42] M. Neuwohner, "An improved approximation algorithm for the maximum weight independent set problem in D-claw free graphs," in *Proc. 38th Int. Symp. Theoretical Aspects Comput. Sci.*, M. Bläser and B. Monmege, Eds., Saarbrücken, Germany, Mar. 2021, pp. 1–20.

[43] M. Neuwohner, "The limits of local search for weighted k-set packing," in *Proc. 23rd Int. Conf.*, in Lecture Notes in Computer Science, vol. 13265, K. Aardal and L. Sanità, Eds. Eindhoven, The Netherlands: Springer, 2022, pp. 415–428.

[44] M. Neuwohner, "Passing the limits of pure local search for weighted K-set packing," in *Proc. Symp. Discrete Algorithms*, N. Bansal and V. Nagarajan, Eds., Florence, Italy, Jan. 2023, pp. 1090–1137.

[45] T. Cui, L. Chen, and T. Ho, "Distributed optimization in wireless networks using broadcast advantage," in *Proc. 46th IEEE Conf. Decis. Control*, May 2007, pp. 5839–5844.

[46] O. Parekh and D. Pritchard, "Generalized hypergraph matching via iterated packing and local ratio," in *Proc. Int. Workshop Approximation Online Algorithms (WAOA)*. Springer, 2014, pp. 207–223.

[47] X. Ma, Q. Gao, V. Marojevic, and J. H. Reed, "Hypergraph matching for MU-MIMO user grouping in wireless LANs," *Ad Hoc Netw.*, vol. 48, pp. 29–37, Sep. 2016.

[48] R. Duvignau and R. Klasing, "Greediness is not always a vice: Efficient discovery algorithms for assignment problems," in *Proc. 12th Latin-American Algorithms, Graphs Optim. Symp. (LAGOS)*, 2023, pp. 43–52.

[49] R. Duvignau, N. Gillet, and R. Klasing, "Greediness is not always a vice: Efficient discovery algorithms for assignment problems," 2024, *arXiv:2410.09434*.

[50] A. Paudel, K. Chaudhari, C. Long, and H. B. Gooi, "Peer-to-peer energy trading in a prosumer-based community microgrid: A game-theoretic model," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6087–6097, Aug. 2019.

[51] Z. Zhang, H. Tang, Q. Huang, and W.-J. Lee, "Two-stages bidding strategies for residential microgrids based peer-to-peer energy trading," in *Proc. IEEE/IAS 55th Ind. Commercial Power Syst. Tech. Conf. (ICPS)*, May 2019, pp. 1–9.

[52] Y. Jia, C. Wan, P. Yu, Y. Song, and P. Ju, "Security constrained P2P energy trading in distribution network: An integrated transaction and operation model," *IEEE Trans. Smart Grid*, vol. 13, no. 6, pp. 4773–4786, Nov. 2022.

[53] Y. Zheng, X. Xue, S. Xi, and W. Xin, "Balancing possibilist-probabilistic risk assessment for smart energy hubs: Enabling secure peer-to-peer energy sharing with CCUS technology and cyber-security," *Energy*, vol. 304, Sep. 2024, Art. no. 132102.

[54] Y. Hong, S. Goel, and W. M. Liu, "An efficient and privacy-preserving scheme for P2P energy exchange among smart microgrids," *Int. J. Energy Res.*, vol. 40, no. 3, pp. 313–331, Oct. 2016.
[55] K. Umer, Q. Huang, M. Khorasany, M. Afzal, and W. Amin, "A novel communication efficient peer-to-peer energy trading scheme for enhanced privacy in microgrids," *Appl. Energy*, vol. 296, Aug. 2021, Art. no. 117075.
[56] Y. Ye, Y. Tang, H. Wang, X.-P. Zhang, and G. Strbac, "A scalable privacy-preserving multi-agent deep reinforcement learning approach for large-scale peer-to-peer transactive energy trading," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 5185–5200, Nov. 2021.
[57] Y. Xia, Q. Xu, S. Tao, P. Du, Y. Ding, and J. Fang, "Preserving operation privacy of peer-to-peer energy transaction based on enhanced benders decomposition considering uncertainty of renewable energy generations," *Energy*, vol. 250, Jul. 2022, Art. no. 123567.
[58] V. Tudor, M. Almgren, and M. Papatriantafilou, "Employing private data in AMI applications: Short term load forecasting using differentially private aggregated data," in *Proc. Int. IEEE Conf. Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People, Smart World Congr. (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, Jul. 2016, pp. 404–413.
[59] V. Tudor, V. Gulisano, M. Almgren, and M. Papatriantafilou, "BES: Differentially private event aggregation for large-scale IoT-based systems," *Future Gener. Comput. Syst.*, vol. 108, pp. 1241–1257, Jul. 2020.
[60] S. A. Ahmed, Q. Huang, Z. Zhang, J. Li, W. Amin, M. Afzal, J. Hussain, and F. Hussain, "Optimization of social welfare and mitigating privacy risks in P2P energy trading: Differential privacy for secure data reporting," *Appl. Energy*, vol. 356, Feb. 2024, Art. no. 122403.
[61] M. R. Garey and D. S. Johnson, *Computers and Intractability*, vol. 174. San Francisco, CA, USA: Freeman, 1979.
[62] J. E. Hopcroft, R. Motwani, and J. D. Ullman, "Introduction to automata theory, languages, and computation," *ACM SIGACT News*, vol. 32, no. 1, pp. 60–65, 2001.
[63] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, nos. 1–2, pp. 83–97, 1955.
[64] L. Ramshaw and R. E. Tarjan, "On minimum-cost assignments in unbalanced bipartite graphs," HP Labs., Palo Alto, CA, USA, Tech. Rep., HPL-2012-40R1, 2012.
[65] Z. Lotker, B. Patt-Shamir, and S. Pettie, "Improved distributed approximate matching," *J. ACM*, vol. 62, no. 5, pp. 1–17, Nov. 2015.
[66] H. Cui, J. Zhang, C. Cui, and Q. Chen, "Solving large-scale assignment problems by Kuhn–Munkres algorithm," in *Proc. 2nd Int. Conf. Adv. Mech. Eng. Ind. Informat. (AMEII)*, 2016, pp. 822–827.
[67] E. Nyholm, J. Goop, M. Odenberger, and F. Johnsson, "Solar photovoltaic-battery systems in Swedish households – self-consumption and self-sufficiency," *Appl. Energy*, vol. 183, pp. 148–159, Dec. 2016.
[68] NREL. (2019). *The Open PV Project*. National Renewable Energy Laboratory. [Online]. Available: https://www.nrel.gov/pv/open-pv-project.html
[69] A. Oller-Westerberg and J. Lindahl, "National survey report of PV power applications in Sweden," *Int. Energy Agency Swedish Energy Agency*, vol. 9, 2014.
[70] Z. Norwood, E. Nyholm, T. Otanicar, and F. Johnsson, "A geospatial comparison of distributed solar heat and power in Europe and the US," *PLoS ONE*, vol. 9, no. 12, Dec. 2014, Art. no. e112442.
[71] L. Göransson, J. Goop, T. Unger, M. Odenberger, and F. Johnsson, "Linkages between demand-side management and congestion in the European electricity transmission system," *Energy*, vol. 69, pp. 860–872, May 2014.
[72] V. Heinisch, M. Odenberger, L. Göransson, and F. Johnsson, "Prosumers in the electricity system—Household vs. system optimization of the operation of residential photovoltaic battery systems," *Frontiers Energy Res.*, vol. 6, p. 145, Jan. 2019.
[73] R. Duvignau, V. Gulisano, M. Papatriantafilou, and V. Savic, "Streaming piecewise linear approximation for efficient data management in edge computing," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2019, pp. 593–596.
[74] B. Havers, R. Duvignau, H. Najdataei, V. Gulisano, A. C. Koppisetty, and M. Papatriantafilou, "DRIVEN: A framework for efficient data retrieval and clustering in vehicular networks," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 1850–1861.
[75] J. van Rooij, J. Swetzén, V. Gulisano, M. Almgren, and M. Papatriantafilou, "EChIDNA: Continuous data validation in advanced metering infrastructures," in *Proc. IEEE Int. Energy Conf. (ENERGYCON)*, Jun. 2018, pp. 1–6.
[76] J. van Rooij, V. Gulisano, and M. Papatriantafilou, "LoCoVolt: Distributed detection of broken meters in smart grids through stream processing," in *Proc. 12th ACM Int. Conf. Distrib. Event-Based Syst.*, Jun. 2018, pp. 171–182.
[77] P. Duchon and R. Duvignau, "Local update algorithms for random graphs," in *Proc. Latin Amer. Symp. Theoretical Inform. (LATIN)*. Springer, 2014, pp. 367–378.
[78] G. Georgiadis and M. Papatriantafilou, "Overlays with preferences: Distributed, adaptive approximation algorithms for matching with preference lists," *Algorithms*, vol. 6, no. 4, pp. 824–856, Nov. 2013.
[79] G. Georgiadis and M. Papatriantafilou, "Adaptive distributed B-matching in overlays with preferences," in *Proc. Int. Symp. Exp. Algorithms (SEA)*. Springer, 2012, pp. 208–223.
[80] A. Khan, A. Pothen, M. M. A. Patwary, N. R. Satish, N. Sundaram, F. Manne, M. Halappanavar, and P. Dubey, "Efficient approximation algorithms for weighted B-matching," *SIAM J. Sci. Comput.*, vol. 38, no. 5, pp. 593–619, 2016.

ROMARIC DUVIGNAU (Member, IEEE) received the Ph.D. degree in computer science from the University of Bordeaux (LaBRI), France, in 2015. He is currently an Associate Professor with the Computer Network and Systems Division, Chalmers University of Technology. He was previously affiliated with Aix-Marseille University (LIF). His research interests include data stream processing, edge computing, p2p energy trading, continuous distributed monitoring, and discovery algorithms for assignment problems.

VINCENZO GULISANO received the Ph.D. degree in computer science from the Polytechnic University of Madrid, Spain. He is currently a Professor with the Computer Network and Systems Division, Chalmers University of Technology. His research interests include data processing and distributed/parallel/elastic and fault-tolerant data streaming.

MARINA PAPATRIANTAFILOU received the Ph.D. degree from the Computer Science and Informatics Department, Patras University. She is a member of the Network of National Contacts ACM-WE NeNaC. She is currently an Associate Professor with the Computer Network and Systems Division, Chalmers University of Technology. Earlier, she was with the Max-Planck Institute for Computer Science, Saarbrücken, and CWI, Amsterdam. Her research interests include efficient and robust parallel, distributed, stream processing and applications in multiprocessor, multicore and distributed, cyber physical systems, synchronization, consistency, and fault-tolerance.

RALF KLASING received the Ph.D. degree from the Department of Mathematics and Computer Science, University of Paderborn, Germany, in 1995. He is currently a Senior Researcher (DR) with CNRS, LaBRI, Bordeaux. From 2010 to 2015, he was the Head of the Combinatorics and Algorithms Team, LaBRI. His research interests include distributed algorithms, approximation algorithms for combinatorially hard problems, algorithmic methods for telecommunication, and communication algorithms in networks.