GPU Optimisation of an Endogenous Peer-to-Peer Market with Product Differentiation

B. Thomas, A. El Ouardi, Samir Bouaziz, Roman Le Goff Latimier, H. Ben Ahmed

GPU Optimisation of an Endogenous Peer-to-Peer Market with Product Differentiation

B. THOMAS; A. EL OUARDI; S. BOUAZIZ
Université Paris-Saclay
ENS Paris-Saclay, CNRS, SATIE
91190, Gif-sur-Yvette, France
{beatrice.thomas}@ens-paris-saclay.fr

R. LE GOFF LATIMIER; H. BEN AHMED
UniR
SATIE, ENS Rennes, CNRS
35170, Bruz, France
{roman.legoff-latimier; hamid.benahmed}@ens-rennes.fr

***Abstract*—With the increase of renewable energy and the consumer flexibility needed in the energy transition, grid management will become more and more complex. Solutions for a decentralized market and handling of the physical state of the network already exist in the literature. But when simulated on a single machine for simulation purposes, these implementations do not scale up well with the problem size. This makes it impossible for researchers and system operators to tune parameters and check for stability, robustness, and dimension in a large grid. In this contribution, we optimized an endogenous Peer to peer market on a Graphic Processing Unit to allow the scaling of these algorithms. The computation times between the Central and Graphic Processing Units are divided by more than 25 with 600 agents, and the speed-up increases with the problem size. Some algorithmic and implementation issues are highlighted for such extensive problem dimensions.**

***Index Terms*—Peer-to-Peer Market, Congestion management, endogenous market, GPU, heterogeneous preferences**

I. Introduction

The energy transition requires, on the first hand, the multiplication of renewable power plants and, on the other hand, the generalization of consumption flexibility [1]. Moreover, mobility migration towards electric energy is likely to cause a considerable increase in consumption [2]. Therefore, the intermittency and imperfect predictability of production must be compensated for by constantly adjusting the flexibilities available on the consumption side [3]. This exercise is made even more difficult by the diversity of distributed energy resources that must be aggregated to have a significant effect [4]. Electricity networks are thus faced with multiple challenges: they will be used more intensively by a considerable number of actors and in a more complex manner.

This situation gives rise to a rich scientific literature devoted to designing new management rules to operate the network safely and efficiently [5]. Indeed, the current management mechanisms would face difficulty scaling up in terms of the number of actors involved. Many efforts are devoted to distributing control in order to overcome this difficulty [6]. Each actor would then perform a portion of the computation, all these results being coordinated by information exchanges until converging towards the global solution [7].

Such approaches open promising ways for solving operationally and in real-time optimal management problems whose complexity is out of all proportion to the current situation. However, before a possible actual deployment, numerous simulations are required from the research community, network managers, and normative authorities. In a non-exhaustive way, it will be necessary to judiciously parameterize the distributed management mechanisms [8], to verify the behavior during extreme events [9], or to ensure the robustness of the proposed solutions in case of faults on the power or communication network [10]. All these simulations cannot yet rely on the computational power provided in a distributed manner and must therefore be performed centrally on a single machine or computing cluster. The efficient and scalable resolution of the distributed management mechanisms proposed in the literature is, consequently, a lock that must be lifted before effective implementation.

In this context, Graphics Processing Units (GPU) are particularly relevant because of their important physical parallelism. These architectures have shown their efficiency in solving power flow (PF) [11], and Optimal Power Flow (OPF) problems [12]. On the other hand, to our knowledge, the transposition of distributed management mechanisms on GPUs is still the subject of few contributions [13]. In particular, because of the diversity of mechanisms currently proposed in the literature, any contribution to this domain bears the risk of being specific to a particular case. To overcome this point, we choose here the so-called endogenous market problem [10], [14], [15]. This solves a peer-to-peer market (P2P) constrained as an OPF by the electrical network's physical equations and limits. The choice of a P2P market is, first of all, motivated by the numerous recent works of which this paradigm has been the object [16] and by the specific functionalities it allows, such as heterogeneous preferences. In addition to its use as such, a P2P market can be considered as a generalization of other market structures [17]. Thus this formalism can be adapted to other cases to ensure the genericity of the work presented here. Taking into account the constraints of the network is also necessary for this genericity and makes it possible to compare results with an OPF, which is a notable particular case of this endogenous market.

The rest of this paper is organized as follows. First, section II will present the endogenous market problem and the algorithm used for its solution. The next part will be devoted

to the architectural optimizations allowing us to obtain the accelerations and scaling effects presented in section IV. The last part will be devoted to a discussion on the limits of the algorithm for high-dimensional resolutions

II. Problem formulation

A. Centralized endogenous market

Let a grid be composed of N_b Buses, L lines (with a unique couple (i,j) of buses associated with each line l), and N agents. The power flow on a line l is noted $\phi_{ij} = \phi_l$, θ_i is the voltage angle, and $B_{ll} = 1/x_l$ the line susceptance. The market problem with grid constraint thought a DC-PF can be defined as follows:

$$\min_{T,P} \quad \sum_{n\in\Omega}\Big(g_n(p_n) + \sum_{m\in\omega_n}\beta_{nm}t_{nm}\Big) \tag{1a}$$

$$\text{s.t.} \quad T = -{}^tT \qquad (\Lambda) \tag{1b}$$

$$p_n = \sum_{m\in\omega_n} T_{nm} \qquad (\mu) \qquad n\in\Omega \tag{1c}$$

$$\underline{p_n} \le p_n \le \overline{p_n} \qquad n\in\Omega \tag{1d}$$

$$T_{nm} \le 0 \qquad n\in\Omega_c \tag{1e}$$

$$T_{nm} \ge 0 \qquad n\in\Omega_g \tag{1f}$$

$$\underline{p_n} \le T_{nm} \le \overline{p_n} \qquad n\in\Omega_p \tag{1g}$$

$$\phi_{ij} = B_l(\theta_i - \theta_j) \qquad (i,j)\in L \tag{1h}$$

$$\underline{\phi_{ij}} \le \phi_{ij} \le \overline{\phi_{ij}} \qquad (\delta_{1-2}) \qquad (i,j)\in L \tag{1i}$$

The optimization aimed at minimizing the sum of the agent's cost function $g_n = \frac{1}{2}a_n p_n^2 + b_n p_n$ and the heterogeneous preferences β_{nm} for all agents $n\in\Omega$. The constraints are the trade balance (1b), the relation between total power and trades for each agent (1c), the power limits (1d) and the trade limits (1e)-(1g) for respectively the consumers Ω_c, the generators Ω_g and the prosumers Ω_p with $\Omega = \Omega_c \cup \Omega_g \cup \Omega_p$ and $\Omega_c \cap \Omega_g \cap \Omega_p = \emptyset$. Finally, (1h) is the DC-PF, and (1i) is the power flow constraint on each line. Constraints (1b), (1c) and (1i) are associated with dual variables λ, μ, δ_1 and δ_2. With G being the grid sensitivity – *i.e.* the relation between the agent power and the power flow, of size $L\cdot N$ – the constraints (1h)-(1i) become:

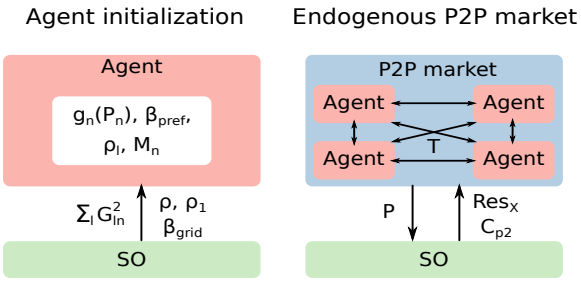
$$|GP| \le \bar{l} \tag{2}$$

$$G = B\cdot C^T\cdot(C\cdot B\cdot C^T)^{-1}\cdot I \tag{3}$$

if $l_{ij} = -\overline{l_{ij}} = \bar{l}$ and where the diagonal matrix B is the susceptance, C and I the incidence matrices respectively between buses and lines and between buses and agents.

According to the study case, some lines may be oversized so that it is known beforehand that their power constraints will never be activated. To save time and memory space, the G matrix can be computed for the whole grid, and the rows corresponding to these lines in G and $\bar{l}$ can be removed. After computing $G\cdot P$ with all rows, this hypothesis can be checked afterward.

Readers can notice that if $\beta_{nm} = 0$ and if only the total power is considered, (1b) becomes that the sum of all powers is null and (1) becomes a DC-OPF with flexible agents. Thus our endogenous market formulation can be considered as a generalization of the DC-OPF.



(a) Initialization

(b) Communication of the endogenous P2P market

Fig. 1: Communication of the Engogenous Peer-to-Peer market

B. Endogenous P2P Market organization

Problem (1) is a quadratic minimization under linear constraint and can be solved as such. Nevertheless, the high complexity of this formulation does not allow it to scale up with the problem size. This is why this resolution would be decentralized in an operational context. In particular, the grid topology, line impedances, and thermal limits must be known to compute the line constraints. Within a microgrid, these could be common knowledge. However, one or several System Operators (SO) are mandatory to handle large-scale networks. The actual implementation of this endogenous P2P market is likely to be decomposed as described in figure 1.

When a new agent joins the market, Fig: 1a, the SO sends him the computation parameters (defined in the following sections). The only agent data that the SO needs is the agent location (the bus of injection).

During the computation, Fig: 1b, each agent must send their trade to the others for the P2P market and the sum of their trades to the SO. The SO computes the grid state and sends to each agent its nodal price (c_{p2}, (14d). This nodal price will change the optimum of the P2P market toward a solution that respects all line constraints. All computations and communications must stop only when the SO confirms that the line constraints are respected (Res_x) and when the P2P Market has converged (Res_s) to a balanced solution (Res_r).

The present contribution focuses on reproducing the behavior of the above management algorithm in an efficient implementation on GPU. Adopting a similar decomposition between the roles of the SO and the P2P market ensures that the behavior of the algorithm is reproduced and thus guarantees its reliability. Moreover, this decomposition is suitable for a parallelized implementation on GPU, as the following sections will describe.

C. Resolution decentralization

The objective of this section is to summarize how (1) can be decomposed as shown in Fig.1. A comprehensive

demonstration can be found in [15]. (1b) and (1i) constraints are relaxed to form their augmented Lagrangian. As the power flow sign in the line is a convention, only one penalty factor ρ_1 can be kept for the two (1i) constraints. To simplify the expressions and reduce the computation's count, some variables are added:

$$\alpha_{ln}^k = G_{ln} p_n^k \tag{4}$$

$$Q_l^{tot} = \sum_{n\in\Omega} \alpha_{ln} = \phi_l \ ; Q_{ln}^{part} = \sum_{j>n} \alpha_{lj} \tag{5}$$

Then an Alternating Direction Method of Multipliers (ADMM), [18] is used to iteratively solve this so-called global problem for each agent n:

$$\begin{aligned}\min_{T_n, p_n} \quad & g_n(p_n) + \sum_{m\in\omega_n} (\beta_{nm} t_{nm}) \\ & + \frac{\rho}{2} \sum_{m\in\omega_n} \left(t_{nm} - \frac{t_{nm}^k - t_{mn}^k}{2} + \frac{\lambda_{nm}^k}{\rho} \right)^2 \\ & + \rho_1 \sum_{l\in L} \left(\alpha_{ln}^2 + (|\kappa_{l1}^k| - |\kappa_{l2}^k| + 2Q_{ln}^{part}) \alpha_{ln} \right)\end{aligned} \tag{6}$$

with the following variable updates and Π^- being the projection to the negative numbers.

$$\begin{aligned}\kappa_{1l}^k &= \bar{l}_l + \delta_{l1}^{k-1} - \sum_{n\in\Omega} G_{ln} p_n^k = \bar{l}_l + \Pi^-(\kappa_{1l}^{k-1}) - Q_l^{tot} \\ \kappa_{2l}^k &= \bar{l}_l + \delta_{l2}^{k-1} + \sum_{n\in\Omega} G_{ln} p_n^k = \bar{l}_l + \Pi^-(\kappa_{2l}^{k-1}) + Q_l^{tot}\end{aligned} \tag{7}$$

$$\lambda_{nm}^k = \lambda_{nm}^{k-1} + \frac{\rho}{2}(t_{nm}^k + t_{mn}^k) \tag{8}$$

Finally, the residuals to check for early convergence can be computed:

$$\begin{aligned}r^{k+1} &= \left\| t_{nm}^{k+1} + t_{mn}^{k+1} \right\| \\ s^{k+1} &= \left\| t_{nm}^{k+1} - t_{nm}^k \right\| \\ x^{k+1} &= \left\| [\Pi^-(\kappa_1^{k+1}) - \Pi^-(\kappa_1^k)]^2 + [\Pi^-(\kappa_2^{k+1}) - \Pi^-(\kappa_2^k)]^2 \right\|\end{aligned} \tag{9}$$

D. P2P market resolution

The minimization (6) for one agent, will be called local problem and can be written in this way:

$$\begin{aligned}T_n^{k+1} = \underset{t_{nm}}{\text{argmin}} \quad & g\Big(\sum_{m\in\omega_n} t_{nm}\Big) + \sum_{m\in\omega_n} f_{nm}(t_{nm}) \\ \text{s.t.} \quad & t_{nm} \in C\end{aligned} \tag{10}$$

and the sharing ADMM solution [18] is:

$$t_i^{j+1} = \underset{lb_n < t_i < ub_n}{\text{argmin}} \left(f(t_i) + \frac{\rho_l}{2} \left\| t_i - t_i^j + \bar{t}^j - \tilde{p}^j + \mu^j \right\|_2^2 \right) \tag{11a}$$

$$\tilde{p}^{j+1} = \underset{\underline{p_n} < M_n \tilde{p} < \overline{p_n}}{\text{argmin}} \left(g(M_n \tilde{p}) + \frac{M_n \rho_l}{2} \left\| \tilde{p} - \mu^j - \bar{t}^{j+1} \right\|_2^2 \right) \tag{11b}$$

$$\mu^{j+1} = \mu^j + \bar{t}^{j+1} - \tilde{p}^{j+1} \tag{11c}$$

$$\begin{aligned}f_i(t_i) &= \frac{\rho}{2}(x_i - \frac{t_{ni}^k - t_{in}^k}{2} + \frac{\lambda_{ni}^k}{\rho})^2 + \beta_{ni} \cdot t_i \\ g(M_n \tilde{p}) &= M_n^2 \cdot \left[0.5 a_n + \rho_1 \sum_{l\in L} G_{ln}^2 \right] \cdot \tilde{p}^2 \\ &\quad + M_n \left[b_n + \rho_1 \sum_{l\in L} \left((|\kappa_{l1}^k| - |\kappa_{l2}^k| + 2Q_{ln}^{part}) G_{ln} \right) \right] \tilde{p}\end{aligned} \tag{12}$$

This is the minimization of two scalar quadratic functions that can be noted as follows:

$$\sum_j a_j \cdot (y - b_j)^2 + \sum_j c_j \cdot y \tag{13}$$

Most of the coefficients are constant. By using respectively t and p subscripts for the coefficients of equations (11a) and (11b), the coefficients that will change according to the global (k) and local (j) iterations are the following:

$$b_{t1} = 0.5(t_{ni}^k - t_{in}^k) - \frac{\lambda_{ni}^k}{\rho} \tag{14a}$$

$$b_{t2} = t_i^j - \bar{t}^j + \tilde{p}^j - \mu^j \tag{14b}$$

$$b_{p1} = \mu^j + \bar{t}^{j+1} \tag{14c}$$

$$c_{p2} = \rho_1 M_n \sum_{l\in L} \left((|\kappa_{l1}^k| - |\kappa_{l2}^k| + 2Q_{ln}^{part}) G_{ln} \right) \tag{14d}$$

Residuals can be computed to check the algorithm early convergence:

$$\begin{aligned}r_l^{j+1} &= \left\| \bar{t}^{j+1} - \tilde{p}^{j+1} \right\| \\ s_l^{j+1} &= \left\| t_i^{j+1} - t_i^j \right\|\end{aligned} \tag{15}$$

To take into account (1d)-(1g), the two solutions t_i and $\tilde{p}$ must be respectively be projected into their admissible set $[lb, ub]$ (according to the agent type) and $[\underline{p_n}/M_n, \overline{p_n}/M_n]$. The whole algorithm is summarized in Alg.1.

Algorithm 1 Global algorithm on CPU-GPU

While$((r, s, x) > \epsilon_{g,x}$ and $k < k_{max})$
P2P Market
  While$((r_l, s_l) > \epsilon_l$ and $j < j_{max})$
    $B_{t2} \leftarrow$ (14b)
    $T^{j+1} \leftarrow$ (11a)
    $T^{j+1} \leftarrow max(min(T^{j+1}, Ub), Lb)$
    $\overline{T} \leftarrow mean(T^{j+1})$
    $B_{p1} \leftarrow$ (14c)
    $\tilde{p} \leftarrow$ (11b)
    $\tilde{p} \leftarrow max(min(\tilde{p}, \overline{p_n}/M_n), \underline{p_n}/M_n)$
    $\mu \leftarrow$ (11c)
    IF $(j\%Step_l == 0)$
      $(s_l^{k+1}, r_l^{k+1}) \leftarrow$ (15)
  $\Lambda, B_{t1} \leftarrow$ (8), (14a)
  $p_n^{k+1} \leftarrow \overline{T} \cdot M_n$
SO computation
  $(\alpha_{ln}^{k+1}, Q^{partial}, Q^{tot}) \leftarrow$ ((4),(5))
  $(\kappa_{1l}^k, \kappa_{2l}^k) \leftarrow$ (7)
  $C_{p2} \leftarrow$ (14d)
  IF $(k\%Step_g == 0)$
    $(r^{k+1}, s^{k+1}, x^{k+1}) \leftarrow$ (9)

III. HARDWARE OPTIMIZATION

A. Test objectives

This section will focus on the implementation of the SO resolution (4), (5), (7) and (14d). Indeed the P2P market has

already been optimized in previous work [13]. Furthermore, with the problem size augmentation, the SO computation becomes the dominant part of the algorithm once switched to a GPU. Indeed, the following Tab. I can be obtained by studying the serial and parallel complexity of the different parts. Representative functions of the complexity of each part have been selected. It can be seen that the complexity of the P2P market only depends on the number of agents N, respectively, in a quadratic or a logarithmic way on the CPU and for the GPU. On the SO part, representative operations depend on both the number of agents N and the number of lines L. On a CPU, complexity is linear towards each of these factors. However, these operations being reductions, they can not be fully parallelized on GPU and depend on one parameter each.

TABLE I: Complexity of the Endogenous Market

Location	Representative function	serial complexity	parallel complexity
P2P	$\overline{T_n} = mean(T_n)\ n \in \Omega$	$O(N^2)$	$O(log(N))$
SO	Q^{part} (5)	$O(N \cdot L)$	$O(N)$
SO	C_{p2} (14d)	$O(N \cdot L)$	$O(log(L))$

To parallelize on GPU using the maximum of its capacities, the following rules must be observed:

- using the minimum number of kernel calls;
- using the minimum number of global memory accesses by using local and shared memory when possible;
- using coalescent accesses to the global memory (adjacent treads must read adjacent memory).

In the Cuda language, kernel calls are applied to a grid. The grid comprises several blocs of threads that share memory and can be synchronized only within a bloc. In our program, all data are stored in row-major order. Thus several cases are possible, Fig. 2:

- each bloc computes on one matrix row, *i.e.* one grid line ($Rdim$);
- each bloc computes on one matrix column, *i.e.* one grid agent ($Cdim$);
- each bloc computes a fixed number of coefficients independently of the rows and columns of the matrix ($1D$).

Thus, to have coalescent accesses, blocs must not compute on the matrix column ($Cdim$). To use shared memory, threads that access the same memory must be in the same bloc. The same applies if the matrix is transposed before being stored ($\cdot \cdot T$), but the data in each bloc will not be the same.

B. Methodology and results

The black box compilation step and the GPU features can have unexpected results on performances. Thus to find the best configuration for each computation, the methodology of [19] has been followed. Data has been randomly generated, and the computation time of each kernel call has been measured for several problem sizes. Several computations to average the results have been made, and data are copied between each computation to prevent any GPU optimization. In our case, the matrices that can be transposed are G, α, and Q^{part}, and this will change all the SO computations.

As the performance can depend on the problem size, Fig. 3 shows the SO computation time according to the problem size comparing the use of the transpose. The best configuration for each computation has been chosen. It can be seen that at a small dimension, the two ways of computing are equivalent. Nevertheless, at a large size, using the transposed matrices is faster. For example, for the case $N = 5000$ and $L = 10000$, the computation time is divided by 2.

IV. Problem Size Impact Analysis

A. Presentation

All C++/Cuda codes, benchmarks, and results are in open access on Gitlab[1]. Simulation has been done on a personal computer with an ADM RYZEN CPU operating at 3,3GHz and an Nvidia GPU GeForce RTX 3060 for a laptop. This GPU has 30 Streaming Multiprocessors and 3840 Cuda cores.

To study the impact of the problem size on the P2P and SO computation time, simulations on random cases of different sizes have been done. To do so, the grid has been fixed from the one in the European open source data [20]. A homogeneous random law has been used to select the constraint lines and their limits, the agent powers, preferences, cost functions, and positions on the grid. The random and simulation parameters are respectively resumed in Tab. II and Tab. III. As the penalty factors are arbitrarily fixed, they can be poorly chosen in some cases. Furthermore, some cases may not be feasible due to the constraint lines. Thus, only the results of cases that achieve the precision asked within the fixed maximum number of iterations will be presented. Computations were made on the CPU and GPU to compare the results.

B. Results

Fig. 4 displays the ratio between the SO and P2P computation time ($Ratio = t_{SO}/t_{P2P}$) according to the problem size.

[1]https://gitlab.com/satie.sete/gpu_endogenous_p2p_market

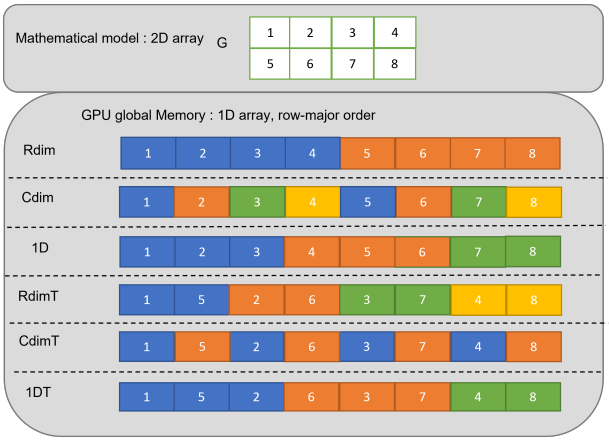


Fig. 2: Memory accesses according to the bloc configurations

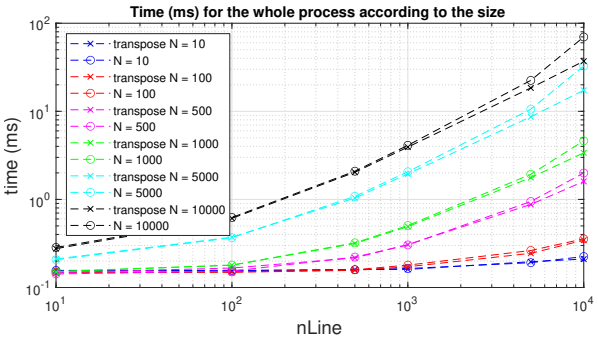


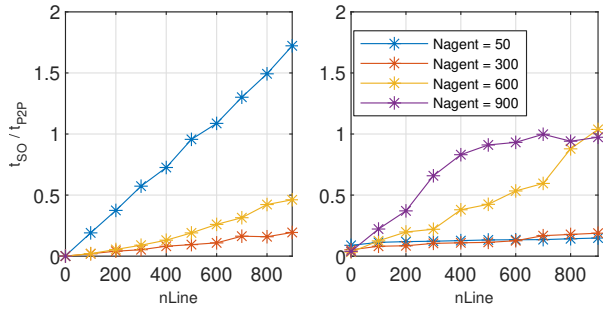
Fig. 3: SO computation time according to the problem size, with and without the transpose

TABLE II: Random characteristics for the study case generation

Features	Consumer		Generator	
	average	variation	average	variation
Proportion	0.6	0	0.4	0
P0 (MW)	60	50	300	250
$\overline{p_n}$ (MW)	-0.9 P0	0	P0	0
$\underline{p_n}$ (MW)	-1.1 P0	0	0	0
a_n (MW^{-2})	1	0	0.01	0
b_n (MW^{-1})	P0	0	20	18
$\lvert\beta\rvert$ (MW^{-1})	4	2	4	2
$\overline{l}$ (MW)	1000	300	1000	300

TABLE III: Parameters for the simulation

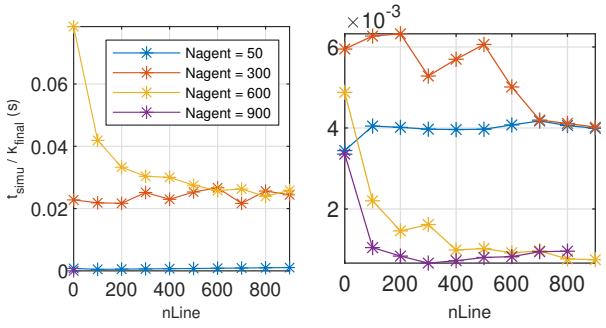
Features	value	Features	value
k_{max}	50000	j_{max}	5000
$step_g$	10	$step_L$	1
ϵ_g	0.01	ϵ_l	0.001
ρ_g	10	ρ_l	ρ_g
ρ_1	0.004	ϵ_x	1
nSimu	20	$offset$	2



(a) CPU version (b) GPU version

Fig. 4: Ratio between SO and P2P computation time

For the CPU, Fig. 4a, the larger the number of agents, the longer it takes to solve the P2P market compared to the SO computation. The opposite happens on the GPU, Fig. 4b. This is relevant to what we demonstrated in the previous section Tab. I.



(a) CPU version (b) GPU version

Fig. 5: Computational time divide by the number of global iteration

Fig. 5 shows the computation time divided by the number of global iterations (k) according to the problem size. For the CPU, Fig. 5a, the computation time increases fast with the number of agents. This is not the case on the GPU, Fig. 5b, thanks to the parallelization. In this study, the more lines there are, the more constrained the problem is when there are many agents. When the line constraints are activated, the problem needs more iterations to be solved, and these iterations are faster as the P2P market only needs to change the agents' trades which activates the constraints.

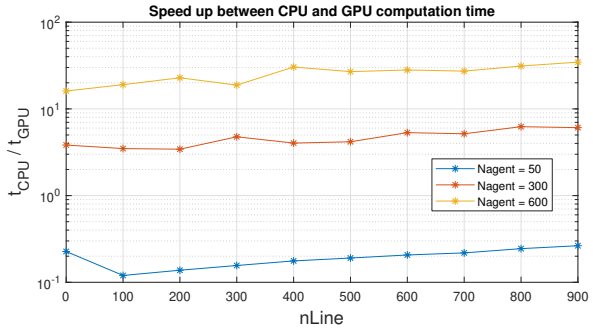


Fig. 6: Speed-up between CPU and GPU

This graph and Fig. 6 also shows the speed-up between the CPU and GPU versions, as they need the same number of iterations to converge. Thus the CPU is faster for small dimensions, but the GPU divides by **5** the computation time at 300 agents and **25** at 600 agents. This speed-up continues to increase with the problem size. To compare with the literature, in [15], the 39-bus IEEE case is solved in 1min with Matlab (CPU). This same case is solved with C++ in about 2s on the CPU and 6s on GPU.

V. Discussion and Limits

The speed-up allowed by the GPU use makes it possible to simulate significant cases and highlights some algorithm problems at these sizes. These issues may prevent any implementation of this algorithm (independently of the hardware used) by imposing a prohibitive number of global iterations (and thus of messages [21]) in a concrete application.

First, this algorithm's performances depend on the penalty factors ρ and ρ_1. According to [18], the penalty factor can be tuned during the computation according to the residuals. Nevertheless, applying that in our case with two penalty factors can no longer ensure convergence. Thus the penalty factors are fixed during the simulation and must be tuned beforehand, which is not always possible.

Furthermore, the increasing size of the study case limits the precision asked for the power flow in the lines. Indeed as $\Phi = G \cdot P$, with G a dense matrix in a case of a meshed grid. A small error on each agent ϵ_{pn} will provoke a noticeable inaccuracy in the power flow $\epsilon_\phi \approx |G| \cdot \epsilon_{pn} \approx N \cdot g \cdot \epsilon_{pn}$ with g the order of magnitude of the G matrix coefficients. Thus it can happen that for a feasible case and a not-bad combination of penalty factors, reaching the asked precision was almost impossible and asked a consequent number of iterations. To solve this issue in our work, the accuracy asked for the power flow has been reduced while allowing the user to add an offset on the line constraints ($\Delta\bar{l} = offset \cdot \epsilon_x$). This offset ensures that the line constraint will be strictly respected even with poor accuracy. The value of $offset$ must be tuned according to the study case; too small, the power flow may exceed the line constraint; too big, it will change the optimum.

VI. Conclusion

To summarize, the main contributions of this work are the following:

- a hardware and software optimization of an endogenous P2P market to reduce the computational time;
- a scaling-up study of the computational time, with the distribution between the SO and the P2P market;
- a highlight of the algorithm limitation for large cases.

Thus this algorithm's implementation on GPU allows for solving significant cases during long periods faster than the CPU solutions. The speed-up achieved at 600 agents is about 25 and increases with the problem size.

Nevertheless, at the moment, many parameters must be tuned to limit the number of global iterations. The algorithm may be improved by using a FAST-ADMM to update the dual variables or by preconditioning the data. Besides, as the SO takes more time with the problem size increase, the computational time can be reduced by decentralizing its computation, making it more suitable for GPU implementation. Finally, this algorithm only considers the line power flows in the case of a DC-PF. Further works on the algorithm must be done to assess voltages, losses, and voltage angles on the grid via an AC-PF.

References

[1] RTE. Conditions and Requirements for the Technical Feasibility of a Power System with a High Share of Renewables in France Towards 2050

[2] Richardson D B, Electric vehicles and the electric grid: A review of modeling approaches, Impacts, and renewable energy integration, Renewable and Sustainable Energy Reviews, Volume 19, 2013, Pages 247-254, ISSN 1364-0321, https://doi.org/10.1016/j.rser.2012.11.042.

[3] Bussar C, Stöcker P, Cai Z, Moraes Jr. L, Magnor D, Wiernes P, Van Bracht N, Moser A, Uwe Sauer D. Large-scale integration of renewable energies and impact on storage demand in a European renewable power system of 2050—Sensitivity study, Journal of Energy Storage, Volume 6, 2016, Pages 1-10, ISSN 2352-152X, https://doi.org/10.1016/j.est.2016.02.004.

[4] Evangelopoulos V A, Kontopoulos T P, Georgilakis P S. Heterogeneous aggregators competing in a local flexibility market for active distribution system management: A bi-level programming approach, International Journal of Electrical Power & Energy Systems, Volume 136, 2022, 107639, ISSN 0142-0615 https://doi.org/10.1016/j.ijepes.2021.107639.,

[5] Capitanescu F, Ramos J M, Panciatici P, Kirschen D, Marcolini A M, Platbrood L, Wehenkel L. State-of-the-art, challenges, and future trends in security constrained optimal power flow. Electric power systems research. (2011). 81(8), 1731-1741.

[6] Biskas P N, Bakirtzis A G . "A decentralized solution to the security constrained DC-OPF problem of multi-area power systems." 2005 IEEE Russia Power Tech. IEEE, 2005.

[7] Kargarian A, Mohammadi J, Guo J, Chakrabarti S, Barati M, Hug G, Baldick R. Toward distributed/decentralized DC optimal power flow implementation in future electric power systems. IEEE Transactions on Smart Grid, 9(4), 2574-2594. 2016

[8] Baroche T, Le Goff Latimier R, Pinson P, Ben Ahmed H. Exogenous Cost Allocation in Peer-to-Peer Electricity Markets. IEEE Transactions on Power Systems, Institute of Electrical and Electronics Engineers, 2019, 34 (4), pp.2553 - 2564. ffhal-01964190f

[9] Ryan H, Marqusee J. "Designing resilient decentralized energy systems: The importance of modeling extreme events and long-duration power outages." Iscience (2021): 103630.

[10] Dong A, Baroche T, Le Goff Latimier R, Ben Ahmed H. Asynchronous algorithm of an endogenous peer-to-peer electricity market. 2021 IEEE Madrid PowerTech. IEEE, 2021

[11] Araújo I, Tadaiesky V, Cardoso D, Fukuyama Y, Santana Á . Simultaneous parallel power flow calculations using hybrid CPU-GPU approach. International Journal of Electrical Power & Energy Systems, 105, 229-236. 2019

[12] Roberge V, Tarbouchi M, Okou F. Optimal power flow based on parallel metaheuristics for graphics processing units. Electric Power Systems Research, Volume 140, 2016, Pages 344-353, ISSN 0378-7796,

[13] Thomas B, Le Goff Latimier R, El Ouardi A, Ben Ahmed H, Bouaziz Samir. Optimization of a peer-to-peer electricity market resolution on GPU. 2022 International Conference on Electrical Sciences and Technologies in Maghreb (CISTEM), 2022

[14] Khorasany M, Mishra Y and Ledwich G . "A Decentralized Bilateral Energy Trading System for Peer-to-Peer Electricity Markets," in IEEE Transactions on Industrial Electronics, vol. 67, no. 6, pp. 4646-4657, June 2020, doi: 10.1109/TIE.2019.2931229.

[15] Chernova T, Gryazina E. Peer-to-peer market with network constraints, user preferences and network charges. International Journal of Electrical Power & Energy Systems, Volume 131, 2021, 106981, ISSN 0142-0615, https://doi.org/10.1016/j.ijepes.2021.106981.

[16] Sousa T, Soares T, Pinson P, Moret F, Baroche T, Sorin E. Peer-to-peer and community-based markets: A comprehensive review, Renewable and Sustainable Energy Reviews, Volume 104, 2019, Pages 367-378,ISSN 1364-0321 https://www.sciencedirect.com/science/article/pii/S1364032119300462

[17] Baroche T, Moret F, Pinson P. (2019, June). Prosumer markets: A unified formulation. In 2019 IEEE Milan PowerTech (pp. 1-6). IEEE.

[18] Boyd S, Parikh N, Chu Borja Peleato E, Eckstein J, Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers

[19] Ofenbeck G, Steinmann R, Caparros V, Spampinato D G, Püschel M. Applying the roofline model. 2014 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2014, pp. 76-85, doi: https://doi.org/10.1109/ISPASS.2014.6844463

[20] Jensen T, Pinson P. RE-Europe, a large-scale dataset for modeling a highly renewable European electricity system. Sci Data 4, 170175 .2017. https://doi.org/10.1038/sdata.2017.175

[21] Le Goff Latimier R, Baroche T, Ben Ahmed H., "Mitigation of Communication Costs in Peer-to-peer Electricity Markets," 2019 IEEE Milan PowerTech, 2019, pp. 1-6, doi: 10.1109/PTC.2019.8810716.