A distributed asynchronous algorithm for the two-stage stochastic unit commitment problem

Ignacio Aravena
CORE, UCL
ignacio.aravena@uclouvain.be

Anthony Papavasiliou
CORE, UCL
anthony.papavasiliou@uclouvain.be

***Abstract*—We present a distributed asynchronous algorithm for solving the two-stage stochastic unit commitment problem. The algorithm uses Lagrangian relaxation to decompose the problem by scenarios and applies an incremental method to solve the dual problem. At each incremental dual iteration, the algorithm evaluates the dual function, providing a lower bound, and recovers a feasible commitment for first stage units, which (through a feasibility recovery process) results in an upper bound. Both the incremental dual iterations as well as the feasibility recovery are executed asynchronously, resulting in more efficient utilization of parallel processors. The method is tested on a model of the Central Western European system, for which it achieved convergence three times faster than an equivalent distributed synchronous algorithm.**

***Index Terms*—stochastic unit commitment, incremental subgradient method, asynchronous algorithm, distributed optimization**

I. Introduction

The large-scale integration of renewable energy has motivated applications of stochastic programming, and recently also robust programming, in power systems planning and operations. The advantage of these optimization paradigms is that their application to the unit commitment problem serve as systematic approaches to schedule production and reserves in electric power systems facing the increased operational uncertainty resulting from the integration of renewable energy resources.

Stochastic unit commitment has been examined in the context of wind power integration. Research demonstrates superior performance relative to deterministic ad-hoc scheduling methodologies in the presence of wind power [1], [2], as well as under uncertain conditions resulting from wind power production and contingencies [3].

The stochastic unit commitment problem is commonly solved using dual decomposition [4], [5]. The customary choice is to decompose the problem across scenarios, by taking the Lagrangian dual with respect to the non-anticipativity constraints. Once a dual problem is at hand, several techniques are available to solve it, including subgradient methods, the progressive hedging algorithm [6], proximal point algorithms and bundle methods among others. As the evaluation of the Lagrangian dual is decomposable across scenarios, it can be parallelized at each iteration of the dual problem. The increase in distributed computing capacity (multi-core, multi-processor and multi-node architectures) has been recently exploited for solving large-scale scale models through progressive hedging [7] and subgradient methods [3], [8].

Two main solution approaches have been proposed for the robust unit commitment problem. One approach relies on a decomposition scheme that sequentially iterates among a master and a slave problem, inspired by Bender's decomposition [9]. The other approach [10] is based on a parallelizable column-and-constraint generation algorithm that allows for second stage discrete variables.

All of the aforementioned parallel (or parallelizable) algorithms for solving stochastic and robust unit commitment share an important drawback: they require synchronization points, i.e. at each iteration all the subprocesses run in parallel but the algorithm must wait until all subprocesses are finished before moving to the next iteration. This results in an under-utilization of the parallel computing infrastructure.

One solution to this drawback would be to replace the iteration method (gradient, subgradient, proximal point) by its incremental version. An incremetal method computes the update direction, at each iteration, based on the information of part of the objective function. These methods have a long tradition in differentiable optimization, and have been recently utilized in non-differential optimization problems, such as the ones arising from dual decomposition. Incremental subgradient methods were first studied by Kibardin [11], and have been the subject of a series of contributions since then, see Nedić and Bertsekas [12] and references therein for a review. Nedić et al. [13] propose and prove the convergence of an asynchronous incremental version of the subgradient algorithm. Kiwiel [14] presented a unified convergence framework for approximate subgradient and incremental subgradient methods. More recently, Bertsekas [15] proposed the extension of these incremental methods to proximal point algorithms.

This paper presents a distributed asynchronous algorithm for the two-stage stochastic unit commitment problem, which is based on the dual decomposition presented in [3]. The dual problem is solved by an asynchronous distributed subgradient method. Primal recovery is also handled asynchronously, by gathering a feasible commitment for first stage units at each incremental iteration, which is then feed to a queue for primal recovery.

The rest of the paper is structured as follows: Section II briefly introduces the two-stage stochastic unit commitment problem and the dual decomposition algorithm. Section III

978-1-4673-8040-9/15/$31.00

presents the asynchronous algorithm, the dual iterations and the primal recovery method. Section IV presents a numerical comparison of the synchronous and asynchronous method, using the Central Western European system as a case study. Section V presents conclusions and outlines extensions of this work.

II. Two-stage stochastic unit commitment and dual decomposition

Unless specified otherwise, lower case letters denote variables, Greek letters denote dual variables, upper case letters denote parameters or sets and bold typeset denotes vectors. Vectors with partial indexation are used to shorten the notation, e.g. if $\boldsymbol{p} = [p_{g_1,s_1,t_1} \dots p_{g_{|G|},s_{|S|},t_{|T|}}]'$ then $\boldsymbol{p}_s = [p_{g_1,s,t_1} \dots p_{g_{|G|},s,t_{|T|}}]'$ and $\boldsymbol{p}_{gs} = [p_{g,s,t_1} \dots p_{g,s,t_{|T|}}]'$.

Following Papavasiliou et al. [8], the two-stage stochastic unit commitment problem can be formulated in a compact fashion as problem (1)-(5).

$$\min_{\substack{p,u,v,f\\ w,z}} \sum_{s\in S}\sum_{g\in G}\sum_{t\in T} \pi_s\big(C_g(p_{gst}) + K_g u_{gst} + S_g v_{gst}\big) \tag{1}$$

$$\text{s.t.} \sum_{g\in G_n} p_{gst} + \sum_{l\in L_n} f_{lst} = D_{nst},\ \forall n\in N, s\in S, t\in T \tag{2}$$

$$(\boldsymbol{p}_s, \boldsymbol{u}_s, \boldsymbol{v}_s, \boldsymbol{f}_s) \in \mathcal{D}_s, \quad \forall s\in S \tag{3}$$

$$(\boldsymbol{w}, \boldsymbol{z}) \in \mathcal{D}^{wz} \tag{4}$$

$$\boldsymbol{u}_{gs} = \boldsymbol{w}_g, \quad \boldsymbol{v}_{gs} = \boldsymbol{v}_g, \quad \forall g\in G_{\text{SLOW}}, s\in S \tag{5}$$

The first stage represents the day-ahead scheduling of slow thermal generators, and the second stage accounts for real-time operations with different realizations for net demand (demand minus renewable energy supply). The variables include power production $\boldsymbol{p}_g$, commitment $\boldsymbol{u}_g$ and startup $\boldsymbol{v}_g$ for each generator $g\in G$, the flow $\boldsymbol{f}_l$ on each line $l\in L$, and the state variables $\boldsymbol{w}_g$ and $\boldsymbol{z}_g$ for each slow generator $g\in G_{\text{SLOW}}$. All variables, except for $\boldsymbol{w}_g$ and $\boldsymbol{z}_g$, are indexed by scenario.

The objective function (1) minimizes the expected generation cost over the set of scenarios S, where π_s denotes the probability of a given scenario s. The generation cost is composed of the fuel consumption cost $C_g(\cdot)$ (a piece-wise linear function), the fixed on cost K_g and the startup cost S_g. Constraints (2) correspond to the power balance at each node $n\in N$. The uncertain net demand of each node is denoted as D_{nst}. Constraints (3) enforce the variables to be in the domain $\mathcal{D}_s$, which includes integrality constraints (for commitment decisions), minimum and maximum power output, ramp-up and ramp-down constraints, minimum up and down time constraints and DC power flow constraints. Constraints (4) enforce operating constraints on the state variables $\boldsymbol{w}_g$ and $\boldsymbol{z}_g$. These constraints are redundant in the extended form of the model, but necessary for the decomposition algorithm. Non-anticipativity constraints for slow generators are imposed by Eq. (5).

The variables $\boldsymbol{p}_s, \boldsymbol{u}_s, \boldsymbol{v}_s, \boldsymbol{f}_s$ are coupled only through the non-anticipativity constraints (5). Therefore, the application of Lagrangian relaxation to these constraints (5) generates a dual problem which is decomposable across scenarios. Let the Lagrangian function be expressed as equation (6):

$$\begin{aligned}\mathcal{L} = \sum_{s\in S}\pi_s\Big(&\sum_{g\in G}\sum_{t\in T}\big(C_g(p_{gst}) + K_g u_{gst} + S_g v_{gst}\big)\\ &+ \sum_{g\in G_{\text{SLOW}}}\sum_{t\in T}\big(\mu_{gst}(u_{gst}-w_{gt}) + \nu_{gst}(v_{gst}-z_{gt})\big)\Big).\end{aligned} \tag{6}$$

Then the dual problem can be expressed as the unconstrained maximization problem (7), with dual variables $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$.

$$\max_{\mu,\nu} \quad \sum_{s\in S} f_s(\boldsymbol{\mu}_s, \boldsymbol{\nu}_s) + g(\boldsymbol{\mu}, \boldsymbol{\nu}), \tag{7}$$

The objective function of the dual problem (7) is composed of the functions $f_s(\cdot)$, defined in Eq. (8) for each scenario s:

$$\begin{aligned} f_s(\boldsymbol{\mu}_s, \boldsymbol{\nu}_s) = \\ \inf_{p,u,v,f}\ &\sum_{t\in T}\Big(\sum_{g\in G}\big(C_g(p_{gst}) + K_g u_{gst} + S_g v_{gst}\big)\\ &+ \sum_{g\in G_{\text{SLOW}}} \mu_{gst}u_{gst} + \nu_{gst}v_{gst}\Big)\\ \text{s.t.}\ &\sum_{g\in G_n} p_{gst} + \sum_{l\in L_n} f_{lst} = D_{nst}, \forall n\in N, t\in T\\ &(\boldsymbol{p}_s, \boldsymbol{u}_s, \boldsymbol{v}_s, \boldsymbol{f}_s)\in\mathcal{D}_s. \end{aligned} \tag{8}$$

The evaluation of these functions requires solving a deterministic unit commitment problem. The other component of the lower bound is the function $g(\cdot)$, defined in equation (9):

$$\begin{aligned} g(\boldsymbol{\mu}, \boldsymbol{\nu}) = \\ \inf_{w,z}\ &\sum_{s\in S}\sum_{g\in G_{\text{SLOW}}}\sum_{t\in T} -\pi_s\big(\mu_{gst}w_{gst} + \nu_{gst}z_{gst}\big)\\ \text{s.t.}\ &(\boldsymbol{w}, \boldsymbol{z})\in\mathcal{D}^{wz} \end{aligned} \tag{9}$$

This evaluation requires solving a scheduling problem that accounts for the minimum up and down times of slow units.

Both functions $f_s(\cdot)$ and $g(\cdot)$ are concave, non-differentiable functions, consequently the optimal solution of the dual problem (7) can be found by applying the subgradient method, proximal point method or other algorithms for convex non-differentiable optimization.

The decomposition algorithm is described in detail in [3], [8]. At each subgradient iteration k, the algorithm evaluates in parallel the dual functions for the current value of the dual variables $\boldsymbol{\mu}^k, \boldsymbol{\nu}^k$, which provides a lower bound (the objective of the dual problem) and a subgradient $\boldsymbol{q}^k, \boldsymbol{r}^k$ for the dual objective, given by relation (10)

$$\begin{aligned} q^k_{gst} &= \pi_s(u^*_{gst} - w^*_{gt}),\ \forall g\in G, s\in S, t\in T\\ r^k_{gst} &= \pi_s(v^*_{gst} - z^*_{gt}),\ \forall g\in G, s\in S, t\in T, \end{aligned} \tag{10}$$

where $\boldsymbol{u}_s^*, \boldsymbol{v}_s^*$ are the optimal values computed during the evaluation of $f_s(\boldsymbol{\mu}_s^k, \boldsymbol{\nu}_s^k)$ and $\boldsymbol{w}^*, \boldsymbol{z}^*$ the optimal values computed during the evaluation of $g(\boldsymbol{\mu}^k, \boldsymbol{\nu}^k)$.

The subgradient is used to update the value of the multipliers and a feasible primal schedule for slow units is recovered from $\boldsymbol{w}^*, \boldsymbol{z}^*$. This schedule is used in order to evaluate the second stage cost of problem (1)-(5), providing an upper bound. The difference between the upper and lower bound, i.e. the duality gap, is used as a stopping criterion.

III. Asynchronous algorithm

The algorithm outlined in the previous section is a synchronous parallel algorithm since all the dual component functions need to be evaluated before moving to the next iteration. In the same sense, primal recovery (second stage cost evaluation) is executed after recovering $\boldsymbol{w}^*, \boldsymbol{z}^*$, i.e. primal recovery is synchronized with dual iterations. These synchronization points result in an under-utilization of parallel processors whenever the time that is required for evaluating the functions $f_s(\cdot)$ varies substantially across scenarios, which is often the case. This section presents an alternative algorithm, that is based on the same decomposition scheme described in the previous section, but avoids synchronization points by using an incremental subgradient method.

A. Incremental subgradient methods

Incremental subgradient methods are of interest when facing convex optimization problems of the form (11)

$$\max_{x \in X} \quad \sum_{i \in I} h_i(\boldsymbol{x}), \tag{11}$$

where X is a nonempty, closed, convex set and the component functions $h_i(\cdot)$ are concave but non-differentiable.

Incremental methods, in general, are based on the idea that it is possible to perform an incremental update based only on the information of some subset of the component functions as long as, in some average sense, the series of incremental updates emulate full updates.

In the case of non-differentiable component functions, Nedić and Bertsekas [12] described and established convergence results for the incremental subgradient method, when the subgradient is evaluated for the current value of the variables $\boldsymbol{x}^k$. An extension of such results is presented in Nedić et al. [13] for the distributed asynchronous subgradient method, where the authors examined convergence of an incremental method with bounded delays in the subgradient computations, assuming that all the component functions are used with the same frequency as the number of iterations tends to infinity. This method does not require any synchronization point and allows for delays among the processes that compute the subgradients of the component functions, which often occurs in distributed implementations. More general convergence results were later provided by Kiwiel [14], whom allowed for errors in the computation of the subgradients.

The update rule of the asynchronous incremental subgradient for problem (11) at the k-th iteration can be described as (12), where $\boldsymbol{q}_i^j$ is a subgradient of the component function $h_i(\cdot)$ at the j-th iteration and $\mathcal{P}_X$ denotes the projection onto X.

$$\boldsymbol{x}^{k+1} := \mathcal{P}_X\left(\boldsymbol{x}^k + \alpha^k \boldsymbol{q}_i^j\right),\ i \in I,\ \boldsymbol{q}_i^j \in \partial_x h_i(\boldsymbol{x}^j),\ j \leq k \tag{12}$$

The incremental step (12) can be used for updating multipliers, however the computation of a lower bound presents a challenge. As different components are evaluated with different values of $\boldsymbol{x}$ due to delays or the existence of a queue (in the case that the number of processors is smaller than $|I|$), there is no lower bound computed from the dual problem. In other words, the dual algorithm will converge but it would require the evaluation of all the component functions for certain $\boldsymbol{x}$'s to produce lower bounds.

B. Dual algorithm and lower bound computation

Suppose that the problem (7) is solved using an incremental update rule, and let k be the index of the current iteration and θ the index of the most recently evaluated component function $f_\theta(\cdot)$. At the k-th iteration, all other component functions $f_t(\cdot),\ t \neq \theta$ have already been evaluated for some $\boldsymbol{\mu}_t^{j(t)}, \boldsymbol{\nu}_t^{j(t)}$, with $j(t) < k$. Therefore, after evaluating $f_\theta(\boldsymbol{\mu}_\theta^k, \boldsymbol{\nu}_\theta^k)$ a lower bound can be obtained by evaluating the dual objective (7) for the dual variables $\boldsymbol{\mu}^j, \boldsymbol{\nu}^j$, defined by relation (13), at the cost of evaluating $g(\boldsymbol{\mu}^j, \boldsymbol{\nu}^j)$.

$$\begin{aligned} \boldsymbol{\mu}^j &:= \left(\boldsymbol{\mu}_{t_1}^{j(t_1)}, \ldots, \boldsymbol{\mu}_\theta^k, \ldots, \boldsymbol{\mu}_{t_n}^{j(t_n)}\right),\ t_l \neq s \\ \boldsymbol{\nu}^j &:= \left(\boldsymbol{\nu}_{t_1}^{j(t_1)}, \ldots, \boldsymbol{\nu}_\theta^k, \ldots, \boldsymbol{\nu}_{t_n}^{j(t_n)}\right),\ t_l \neq s \end{aligned} \tag{13}$$

Note that $\boldsymbol{\mu}^j, \boldsymbol{\nu}^j$ corresponds to the concatenation of all the dual multipliers for which the functions $f_s(\cdot)$ were lastly evaluated. This lower bound can be computed at each incremental iteration using equation (14)[1].

$$LB^k = f_\theta\left(\boldsymbol{\mu}_\theta^k, \boldsymbol{\nu}_\theta^k\right) + \sum_{\substack{t \in S \\ t \neq \theta}} f_t\left(\boldsymbol{\mu}_t^{j(t)}, \boldsymbol{\nu}_t^{j(t)}\right) + g(\boldsymbol{\mu}^j, \boldsymbol{\nu}^j) \tag{14}$$

Provided that the evaluation of the function $g(\cdot)$ is much faster than the evaluation of $f_s(\cdot)$ (which is the case in our decomposition approach), the additional computational burden required in order to obtain a lower bound at each iteration is negligible. Moreover, since the subgradients of $f_\theta(\cdot)$ and $g(\cdot)$ at $\boldsymbol{\mu}^j, \boldsymbol{\nu}^j$ are available after computing the lower bound, an incremental subgradient update can be made on $\boldsymbol{\mu}, \boldsymbol{\nu}$ according to (15)

$$\begin{aligned} \boldsymbol{\mu}_\theta^{k+1} &:= \boldsymbol{\mu}_\theta^k + \alpha^k \pi_\theta(\boldsymbol{u}_\theta^* - \boldsymbol{w}^*) \\ \boldsymbol{\mu}_t^{k+1} &:= \boldsymbol{\mu}_t^k,\ \forall t \in S, t \neq \theta \\ \boldsymbol{\nu}_\theta^{k+1} &:= \boldsymbol{\nu}_\theta^k + \alpha^k \pi_\theta(\boldsymbol{v}_\theta^* - \boldsymbol{z}^*) \\ \boldsymbol{\nu}_t^{k+1} &:= \boldsymbol{\nu}_t^k,\ \forall t \in S, t \neq \theta, \end{aligned} \tag{15}$$

[1]The evaluation of $f_s(\cdot)$ and $g(\cdot)$ requires solving MILP problems. Consequently, the computation of the lower bound of (14) requires using the lower bound of the branch and bound algorithm.

where $\boldsymbol{u}_\theta^*, \boldsymbol{v}_\theta^*$ are the optimal values from the evaluation of $f_\theta(\boldsymbol{\mu}_\theta^k, \boldsymbol{\nu}_\theta^k)$ and $\boldsymbol{w}^*, \boldsymbol{z}^*$ the optimal values from the evaluation of $g(\boldsymbol{\mu}^j, \boldsymbol{\nu}^j)$.

The incremental update rule (15) already accounts for the incremental subgradient iterations of $g(\cdot)$, thus there is no need to consider a parallel process that generates incremental updates based on $g(\cdot)$, in fact doing it would violate the assumption that updates from the different component functions are used with the same frequency [13]. In a broader sense the update rule (15) can be interpreted as an approximate subgradient update[2], for which convergence results are provided in [14]. Consequently with [14, Theorem 3.4], the chosen step size for the incremental iterations is the diminishing (non-summable) step size of equation (16):

$$\alpha^k = \frac{a}{k+b} \tag{16}$$

In summary, each incremental subgradient iteration consists of four steps: (*i*) evaluate $f_s(\boldsymbol{\mu}_s^k, \boldsymbol{\nu}_s^k)$, (*ii*) evaluate $g(\boldsymbol{\mu}^j, \boldsymbol{\nu}^j)$, (*iii*) compute a lower bound according to (14) and (*iv*) update the dual multipliers of scenario s using Eq. (15). These steps can be executed in parallel for each scenario, without any synchronization point.

C. Primal recovery

As mentioned previously, a feasible commitment for slow units can be obtained at each iteration from the evaluation of the function $g(\cdot)$. For the system studied in this paper, these primal candidates exhibit poor performance during the early subgradient iterations. Another source of primal candidates are the optimal solutions from the function evaluations $f_s(\cdot)$. These additional candidates accelerate the convergence of the upper bound substantially, possibly due to the fact that their computation involves the solution of an entire unit commitment problem.

Consequently, the present algorithm obtains a primal candidate c, i.e. a unit commitment schedule $\boldsymbol{u}^c, \boldsymbol{v}^c$ that respect the minimum up and down times of slow units, from the evaluation of $f_s(\cdot)$ at each incremental subgradient iteration. To produce complete primal solutions, these primal candidates need to be evaluated for their second stage cost for each scenario. As in the case of the dual function evaluation, the computations of the second stage costs can be parallelized.

D. Algorithm implementation

Even though the iterations described in the previous subsections do not require synchronization, they do require communication among processors. In the case of dual incremental iterations, the lower bound evaluation and the update rule requires knowledge of the value of the last dual multipliers evaluated for the other scenarios, while the primal recovery procedure requires knowledge of the slow generator commitments from the dual iterations. Moreover, following the update of the upper and lower bound it is necessary to compute the current optimality gap and decide whether or not to terminate the algorithm. All these tasks can be executed by a coordinator, using the implementation scheme depicted in Fig. 1, where $\tilde{f}_s^c$ stands for the second stage cost of primal candidate c in scenario s.

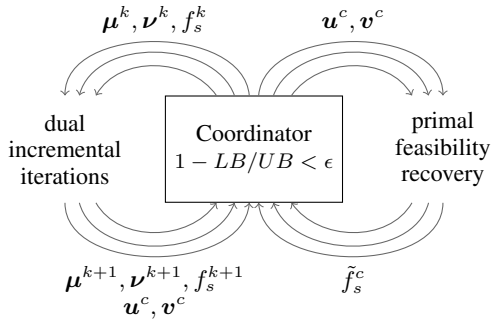

Fig. 1. Asynchronous algorithm implementation scheme.

In practice, the time between two incremental iterations is smaller than the time that takes to evaluate the second stage costs for any primal candidate, hence new primal candidates must enter a queue prior to be evaluated for the second stage costs. Jobs are also queued during the computation of the dual function if the number of processors available is lower than the number of scenarios. These queues are managed by the coordinator, and in the implementation discussed in the case study they are executed following the first-in-first-out principle.

IV. Numerical results

The asynchronous algorithm was tested on a typical autumn day in the Central Western European system, which includes Austria, Belgium, France, Germany, Luxembourg, the Netherlands and Switzerland. The transmission network corresponds to the model of Hutcheon and Bialek [16], with 679 nodes and 1037 lines. Generator data were obtained from GDF Suez. The system consists of 639 thermal units. 20 scenarios for renewable energy supply were selected from the historical records of autumn 2013.

The algorithm was implemented in Mosel and XPress [17] and it was executed on a Macbook Pro with 12 cores (2 $\times$ Intel Xeon X5650) and 32GB of RAM. A 0.5% MIP gap was used for the evaluation of $f_s(\cdot)$ and $\tilde{f}_s^c$, and a 0.1% for the evaluation of $g(\cdot)$.

The problem was solved using the asynchronous algorithm and a synchronous algorithm based on [3] that also recovers primal candidates from the evaluation of $f_s(\cdot)$. For the asynchronous version, half of the processors where dedicated to dual incremental iterations and the other half to primal recovery. Fig. 2 shows the performance of both algorithms in terms of iterations. The step size parameters are set such that both the asynchronous and synchronous algorithm start with the same step size, however the step size of the synchronous algorithm is set to decrease $|S|$ times faster per iteration, so that (roughly

[2]In practice, the subgradient method is also an approximate subgradient method since the MILP problems involved evaluation of the functions $f_s(\cdot)$ and $g(\cdot)$ are only solved up to a certain accuracy.

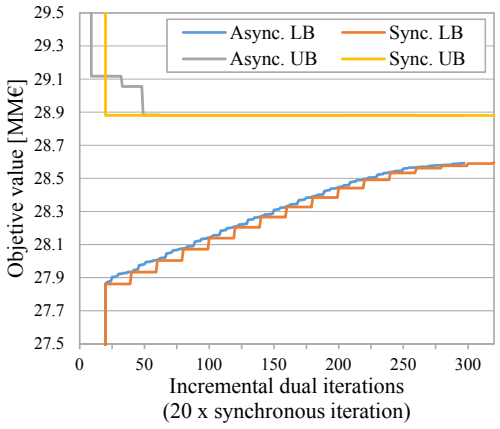

Fig. 2. Convergence of the asynchronous and synchronous versions of the algorithm.

speaking) $|S|$ incremental subgradient iterations are equivalent to one subgradient iteration (in Fig. 2, at any iteration of the synchronous algorithm, the asynchronous algorithm uses the same step size). Under these conditions, the dual incremental subgradient method is a bit faster than the subgradient method, in terms of equivalent iterations. In terms of run time, the asynchronous algorithm required 7 hours and 35 minutes to achieve a 1% optimality gap, while the synchronous algorithm required 21 hours and 32 minutes. This difference is mainly due to the synchronization points between dual iterations and feasibility recovery; the asynchronous algorithm continuously iterates over dual and primal, therefore even if the number of queued primal candidates is huge, they just wait on the queue while the dual iterations continue, as opposed, the synchronous method first iterates on dual and then on primal, evaluating exhaustively all the new primal candidates found on each iteration.

If the synchronous algorithm were limited to primal feasibility recovery using the optimal solution of $g(\cdot)$, the best upper bound found up to the 16th iteration would be 43.9MM€, more than 53% higher than the best objective function that we were able to compute. In that case, the algorithm would not converge until $g(\cdot)$ would start producing good candidates, even if the dual function is already close to its optimum. This highlights the importance of primal feasible solution recovery in dual decomposition schemes.

V. Conclusions

This paper presents an asynchronous algorithm for the two-stage stochastic unit commitment problem based on Lagrangian relaxation and the asynchronous incremental subgradient method. The observed convergence of the incremental subgradient method is very similar to the one of the subgradient method, on the other hand the lack of synchronization points results in substantial speedup of the asynchronous method compared to the subgradient method.

The implementation of the method in a high performance computing environment and the extension of the presented framework to incremental proximal point algorithms [15] will be explored in future research.

Acknowledgment

The authors would like to thank GDF Suez for providing the generators database used in this study and to the Fair Isaac Corporation FICO for providing licenses for XPress. This research was funded by the Univeristé catholique de Louvain through an FSR grant.

References

[1] P. Ruiz, C. Philbrick, E. Zak, K. Cheung, and P. Sauer, "Uncertainty management in the unit commitment problem," *Power Systems, IEEE Transactions on*, vol. 24, pp. 642–651, May 2009.

[2] J. Morales, A. Conejo, and J. Perez-Ruiz, "Economic valuation of reserves in power systems with high penetration of wind power," *Power Systems, IEEE Transactions on*, vol. 24, pp. 900–910, May 2009.

[3] A. Papavasiliou and S. S. Oren, "Multiarea stochastic unit commitment for high wind penetration in a transmission constrained network," *Operations Research*, vol. 61, no. 3, pp. 578–592, 2013.

[4] S. Takriti, J. Birge, and E. Long, "A stochastic model for the unit commitment problem," *IEEE Transactions on Power Systems*, vol. 11, pp. 1497–1508, Aug 1996.

[5] P. Carpentier, G. Gohen, J.-C. Culioli, and A. Renaud, "Stochastic optimization of unit commitment: a new decomposition framework," *IEEE Transactions on Power Systems*, vol. 11, pp. 1067–1073, May 1996.

[6] R. T. Rockafellar and R. J.-B. Wets, "Scenarios and policy aggregation in optimization under uncertainty," *Mathematics of Operations Research*, vol. 16, no. 1, pp. 119–147, 1991.

[7] S. Ryan, R.-B. Wets, D. Woodruff, C. Silva-Monroy, and J.-P. Watson, "Toward scalable, parallel progressive hedging for stochastic unit commitment," in *2013 IEEE Power and Energy Society General Meeting (PES)*, pp. 1–5, July 2013.

[8] A. Papavasiliou, S. Oren, and B. Rountree, "Applying high performance computing to transmission-constrained stochastic unit commitment for renewable energy integration," *IEEE Transactions on Power Systems*, vol. PP, no. 99, pp. 1–12, 2014.

[9] D. Bertsimas, E. Litvinov, X. Sun, J. Zhao, and T. Zheng, "Adaptive robust optimization for the security constrained unit commitment problem," *IEEE Transactions on Power Systems*, vol. 28, pp. 52–63, Feb 2013.

[10] B. Zeng and L. Zhao, "Solving two-stage robust optimization problems using a column-and-constraint generation method," *Operations Research Letters*, vol. 41, no. 5, pp. 457 – 461, 2013.

[11] V. Kibardin, "Decomposition into functions in the minimization problem," *Avtomat. i Telemekh.*, vol. 9, pp. 66–79, 1979. (In Russian).

[12] A. Nedić and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[13] A. Nedić, D. Bertsekas, and V. Borkar, "Distributed asynchronous incremental subgradient methods," in *Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications* (Y. Butnariu, Dan amd Censor and S. Reich, eds.), vol. 8 of *Studies in Computational Mathematics*, pp. 381–407, Amsterdam: Elsevier, 2001.

[14] K. Kiwiel, "Convergence of approximate and incremental subgradient methods for convex optimization," *SIAM Journal on Optimization*, vol. 14, no. 3, pp. 807–840, 2004.

[15] D. Bertsekas, "Incremental proximal methods for large scale convex optimization," *Mathematical Programming*, vol. 129, no. 2, pp. 163–195, 2011.

[16] N. Hutcheon and J. Bialek, "Updated and validated power flow model of the main continental European transmission network," in *2013 IEEE PowerTech Grenoble*, pp. 1–5, June 2013.

[17] Y. Colombani and S. Heipcke, "Multiple models and parallel solving with Mosel," February 2014. Avalaible at: http://community.fico.com/docs/DOC-1141. Accessed: 2014-11-20.